
Installing, Configuring, and Running the Open edX Platform: Eucalyptus Release

Release

September 30, 2016

1	General Information	3
1.1	Read Me	3
1.2	Other edX Resources	3
1.3	edX Browser Support	9
1.4	Change Log	9
2	Open edX Platform Releases	11
2.1	Open edX Eucalyptus Release	11
2.2	Open edX Dogwood Release	13
2.3	Open edX Cypress Release	17
2.4	Open edX Birch Release	19
3	Installing and Starting the Open edX Platform	23
3.1	Open edX Platform Installation Options	23
3.2	Installation Prerequisites	26
3.3	Getting Help	26
3.4	Installing and Starting Devstack	27
3.5	Installing and Starting Fullstack	31
3.6	Installing and Starting Analytics Devstack	33
4	Configuring the Open edX Platform	39
4.1	Guidelines for Updating the Open edX Platform	39
4.2	Configuring Open edX Sites	39
4.3	Changing the Appearance of Open edX Sites	41
4.4	Adding Custom Fields to the Registration Page	48
4.5	Specifying Allowed Registration Email Patterns	49
4.6	Adding the CourseTalk Widget	50
4.7	Enabling Open edX Search	52
4.8	Enabling Badging	55
4.9	Enabling Certificates	65
4.10	Enabling Custom Courses	68
4.11	Enabling Entrance Exams	69
4.12	Configuring Open Response Assessments	70
4.13	Enabling Course Prerequisites	71
4.14	Enabling Course and Video Licensing	72
4.15	Configuring an edX Instance as an LTI Tool Provider	72
4.16	Enabling Social Sharing of Courses and Certificates	78
4.17	Enabling Third Party Authentication	80

4.18	Enabling Timed Exams	91
4.19	Setting Up the YouTube API Key	91
4.20	Installing an XBlock	93
4.21	Enabling a CDN for Course Assets	94
5	Adding edX Insights for Course Teams	95
5.1	Options for Installing edX Insights	95
5.2	Using Elastic MapReduce on AWS	100
6	Adding E-Commerce to the Open edX Platform	107
6.1	Install and Start the E-Commerce Service	107
6.2	Manage Static Assets	111
6.3	Create E-Commerce Products	111
6.4	Manage Orders	121
6.5	Test Features	122
6.6	Test Your E-Commerce Application	122
6.7	Additional E-Commerce Features	126
7	Setting Up the Open edX Mobile Applications	129
7.1	Accessing the Source Code	129
7.2	Configuring Mobile Application Features	129
7.3	Configuring Video Modules for Mobile	131
7.4	Enabling Push Notifications	131
8	Index of EdX Feature Flags	133
9	Glossary	135
9.1	A	135
9.2	C	136
9.3	D	138
9.4	E	138
9.5	F	139
9.6	G	139
9.7	H	139
9.8	I	140
9.9	K	140
9.10	L	140
9.11	M	141
9.12	N	141
9.13	O	141
9.14	P	142
9.15	Q	142
9.16	R	143
9.17	S	143
9.18	T	143
9.19	U	144
9.20	V	144
9.21	W	144
9.22	XYZ	145

This guide provides instructions for using your own instance of the Open edX platform and associated applications. This document also contains instructions for installing Open edX releases.

General Information

1.1 Read Me

The *Installing and Configuring the Open edX Platform* documentation is created using [RST](#) files and [Sphinx](#). As a member of the community, you can help update and revise this documentation project on GitHub:

```
https://github.com/edx/edx-documentation/tree/master/en_us/install_operations/source
```

To suggest a revision, follow the [GitHub Flow](#): fork the project, make changes in your fork, and submit a pull request back to the original project.

1.2 Other edX Resources

Course teams, researchers, developers, learners: the edX community includes groups with a range of reasons for using the platform and objectives to accomplish. To help members of each group learn about what edX offers, reach goals, and solve problems, edX provides a variety of information resources.

To help you find what you need, browse the edX offerings in the following categories.

- *[The edX Partner Portal](#)*
- *[The Open edX Portal](#)*
- *[System Status](#)*
- *[Resources for Course Teams](#)*
- *[Resources for Researchers](#)*
- *[Resources for Developers](#)*
- *[Resources for Open edX](#)*
- *[Resources for Learners](#)*

All members of the edX community are encouraged to make use of the resources described in this preface. We welcome your feedback on these edX information resources. Contact the edX documentation team at docs@edx.org.

1.2.1 The edX Partner Portal

The [edX Partner Portal](#) is the destination for partners to learn, connect, and collaborate with one another. Partners can explore rich resources and share success stories and best practices while staying up-to-date with important news and updates.

To use the edX Partner Portal, you must register and request verification as an edX partner. If you are an edX partner and have not used the edX Partner Portal, follow these steps.

1. Visit partners.edx.org, and select **Create New Account**.
2. Select **Request Partner Access**, then fill in your personal details.
3. Select **Create New Account**. You will receive a confirmation email with your account access within 24 hours.

After you create an account, you can sign up to receive email updates about edX releases, news from the product team, and other announcements. For more information, see [Release Announcements by Email](#).

Course Team Support in the edX Partner Portal

EdX partner course teams can get technical support in the [edX Partner Portal](#). To access technical support, submit a support ticket, or review any support tickets you have created, go to partners.edx.org and select **Course Staff Support** at the top of the page. This option is available on every page in the Partner Portal.

1.2.2 The Open edX Portal

The [Open edX Portal](#) is the destination for all edX users to learn about the edX roadmap, as well as hosting, extending the edX platform, and contributing to Open edX. In addition, the Open edX Portal provides product announcements, the Open edX blog, and other rich community resources.

All users can view content on the Open edX Portal without creating an account and logging in.

To comment on blog posts or the edX roadmap, or subscribe to email updates, you must create an account and log in. If you do not have an account, follow these steps.

1. Visit open.edx.org/user/register.
2. Fill in your personal details.
3. Select **Create New Account**. You are then logged in to the [Open edX Portal](#).

Release Announcements by Email

To receive and share product and release announcements by email, you can subscribe to announcements on one of the edX portal sites.

1. Create an account on the [Open edX Portal](#) or the [edX Partner Portal](#) as described above.
2. Select **Community** and then **Announcements**.
3. Under **Subscriptions**, select the different types of announcements that you want to receive through email. You might need to scroll down to see these options.
4. Select **Save**.

You will now receive email messages when new announcements of the types you selected are posted.

1.2.3 System Status

For system-related notifications from the edX operations team, including outages and the status of error reports. On [Twitter](#), you can follow @edxstatus.

Current system status and the uptime percentages for edX servers, along with the Twitter feed, are published on the [edX Status](#) web page.

1.2.4 Resources for Course Teams

Course teams include faculty, instructional designers, course staff, discussion moderators, and others who contribute to the creation and delivery of courses on [edx.org](#) or edX Edge.

The edX Learning Series

The courses in the edX Learning Series provide foundational knowledge about using the edX platform. These courses are available on [edx.org](#).

edX101: Overview of Creating a Course

The [edX101](#) course is designed to provide a high-level overview of the course creation and delivery process using Studio and the edX LMS. It also highlights the extensive capabilities of the edX platform.

StudioX: Creating a Course with edX Studio

After you complete [edX101](#), [StudioX](#) provides more detail about using Studio to create a course, add different types of content, and configure your course to provide an optimal on-line learning experience.

VideoX: Creating Video for the edX Platform

[VideoX](#) presents strategies for creating videos for course content and course marketing. The course provides step-by-step instructions for every stage of video creation, and includes links to exemplary sample videos created by edX partner institutions.

Documentation

Documentation for course teams is available on the [docs.edx.org](#) web page.

- [Building and Running an edX Course](#) is a comprehensive guide with concepts and procedures to help you build a course in edX Studio, and then use the Learning Management System (LMS) to run a course.
When you are working in edX Studio, you can access relevant sections of this guide by selecting **Help** on any page.
- [Using edX Insights](#) describes the metrics, visualizations, and downloadable .csv files that course teams can use to gain information about student background and activity.
- The [edX Release Notes](#) summarize the changes in each new version of deployed software.

These guides open in your web browser. The left side of each page includes a **Search docs** field and links to the contents of that guide. To open or save a PDF version, select **v: latest** at the lower right of the page, then select **PDF**.

Note: If you use the Safari browser, be aware that it does not support the search feature for the HTML versions of the edX guides. This is a known limitation.

Email

To receive and share information by email, course team members can:

- Subscribe to announcements and other new topics in the edX Partner Portal or the Open edX Portal. For information about how to subscribe, see [Release Announcements through the Open edX Portal](#).
- Join the [openedx-studio](#) Google group to ask questions and participate in discussions with peers at other edX partner organizations and edX staffers.

Wikis and Web Sites

The edX product team maintains public product roadmaps on [the Open edX Portal](#) and [the edX Partner Portal](#).

The [edX Partner Support](#) site for edX partners hosts discussions that are monitored by edX staff.

1.2.5 Resources for Researchers

At each partner institution, the data czar is the primary point of contact for information about edX data. To set up a data czar for your institution, contact your edX partner manager.

Data for the courses on [edx.org](#) and edX Edge is available to the data czars at our partner institutions, and then used by database experts, statisticians, educational investigators, and others for educational research.

Resources are also available for members of the Open edX community who are collecting data about courses running on their sites and conducting research projects.

Documentation

The [edX Research Guide](#) is available on the [docs.edx.org](#) web page. Although it is written primarily for data czars and researchers at partner institutions, this guide can also be a useful reference for members of the Open edX community.

The [edX Research Guide](#) opens in your web browser, with a **Search docs** field and links to sections and topics on the left side of each page. To open or save a PDF version, select **v: latest** at the lower right of the page, and then select **PDF**.

Note: If you use the Safari browser, be aware that it does not support the search feature for the HTML versions of the edX guides. This is a known limitation.

Discussion Forums and Email

Researchers, edX data czars, and members of the global edX data and analytics community can post and discuss questions in our public research forum: the [openedx-analytics](#) Google group.

The edX partner portal also offers community [forums](#), including a Research and Analytics topic, for discussions among edX partners.

Important: Please do not post sensitive data to public forums.

Data czars who have questions that involve sensitive data, or that are institution specific, can send them by email to data.support@edx.org with a copy to your edX partner manager.

Wikis

The edX Analytics team maintains the [Open edX Analytics](#) wiki, which includes links to periodic release notes and other resources for researchers.

The [edx-tools](#) wiki lists publicly shared tools for working with the edX platform, including scripts for data analysis and reporting.

1.2.6 Resources for Developers

Software engineers, system administrators, and translators work on extending and localizing the code for the edX platform.

Documentation

Documentation for developers is available on the docs.edx.org web page.

- The [edX Platform Developer's Guide](#) includes guidelines for contributing to Open edX, options for extending the Open edX platform, using the edX public sandboxes, instrumenting analytics, and testing.
- [Installing, Configuring, and Running the Open edX Platform](#) provides procedures for getting an edX developer stack (devstack) and production stack (fullstack) operational.
- [Open edX XBlock Tutorial](#) guides developers through the process of creating an XBlock, and explains the concepts and anatomy of XBlocks.
- [Open edX XBlock API Guide](#) provides reference information about the XBlock API.
- [edX Open Learning XML Guide](#) provides guidelines for building edX courses with Open Learning XML (OLX). Note that this guide is currently an alpha version.
- [edX Data Analytics API](#) provides reference information for using the data analytics API to build applications to view and analyze learner activity in your course.
- [edX Platform APIs](#) provide reference information for building applications to view course information and videos and work with user and enrollment data.

Note: If you use the Safari browser, be aware that it does not support the search feature for the HTML versions of the edX guides. This is a known limitation.

GitHub

These are the main edX repositories on GitHub.

- The [edx/edx-platform](#) repo contains the code for the edX platform.
- The [edx/edx-analytics-dashboard](#) repo contains the code for edX Insights.
- The [edx/configuration](#) repo contains scripts to set up and operate the edX platform.

Additional repositories are used for other projects. Our contributor agreement, contributor guidelines and coding conventions, and other resources are available in these repositories.

Community Discussions

The [Community Discussions](#) page in the Open edX Portal lists different ways that you can ask, and answer, questions.

Wikis and Web Sites

The [Open edX Portal](#) is the entry point for new contributors.

The edX Engineering team maintains an [open Confluence wiki](#), which provides insights into the plans, projects, and questions that the edX Open Source team is working on with the community.

The [edx-tools](#) wiki lists publicly shared tools for working with the edX platform, including scripts and helper utilities.

1.2.7 Resources for Open edX

Hosting providers, platform extenders, core contributors, and course staff all use Open edX. EdX provides release-specific documentation, as well as the latest version of all guides, for Open edX users. The following documentation is available.

- [Open edX Release Notes](#) provides information on the contents of Open edX releases.
- [Building and Running an Open edX Course](#) is a comprehensive guide with concepts and procedures to help you build a course in Studio, and then use the Learning Management System (LMS) to run a course.

When you are working in Studio, you can access relevant sections of this guide by selecting **Help** on any page.

- [Open edX Learner's Guide](#) helps students use the Open edX LMS to take courses. This guide is available on the docs.edx.org web page. Because learners are currently only guided to this resource through the course, we encourage course teams to provide learners with links to this guide as needed in course updates or discussions.
- [Installing, Configuring, and Running the Open edX Platform](#) provides information about installing and using devstack and fullstack.
- The [edX Platform Developer's Guide](#) includes guidelines for contributing to Open edX, options for extending the Open edX platform, using the edX public sandboxes, instrumenting analytics, and testing.
- [Open edX XBlock Tutorial](#) guides developers through the process of creating an XBlock, and explains the concepts and anatomy of XBlocks.
- [Open edX XBlock API Guide](#) provides reference information on the XBlock API.
- [EdX Open Learning XML Guide](#) provides guidelines for building edX courses with Open Learning XML (OLX). Note that this guide is currently an alpha version.
- [EdX Data Analytics API](#) provides reference information for using the data analytics API to build applications to view and analyze learner activity in your course.
- [EdX Platform APIs](#) provide reference information for building applications to view course information and videos and work with user and enrollment data.

Note: If you use the Safari browser, be aware that it does not support the search feature for the HTML versions of the edX guides. This is a known limitation.

1.2.8 Resources for Learners

Documentation

The [EdX Learner's Guide](#) and the [Open edX Learner's Guide](#) are available on the docs.edx.org web page. Because learners are currently only guided to this resource through the course, we encourage course teams to provide learners with links to these guides as needed in course updates or discussions.

In a Course

All edX courses have a discussion forum where you can ask questions and interact with other students and with the course team: select **Discussion**. Many courses also offer a wiki for additional resources and materials: select **Wiki**.

Other resources might also be available, such as a course-specific Facebook page or Twitter feed, or opportunities for Google Hangouts. Be sure to check the **Home** page for your course as well as the **Discussion** and **Wiki** pages.

From time to time, the course team might send email messages to all students. While you can opt out of these messages, doing so means that you can miss important or time-sensitive information. To change your preferences for course email, select **edX** or **edX edge** at the top of any page. On your dashboard of current courses, locate the course and then select **Email Settings**.

From edX

To help you get started with the edX learning experience, edX offers a course (of course!). You can find the edX [Demo](#) course on the edX web site. EdX also maintains a list of [frequently asked questions](#) and answers.

If you still have questions or suggestions, you can get help from the edX support team: select **Contact** at the bottom of any edX web page or send an email message to info@edx.org.

For opportunities to meet others who are interested in edX courses, check the edX Global Community [meetup](#) group.

1.3 edX Browser Support

The edX platform runs on the following browsers.

- [Chrome](#)
- [Safari](#)
- [Firefox](#)
- [Microsoft Edge](#) and [Microsoft Internet Explorer 11](#)

The edX platform is routinely tested and verified on the current version and the previous version of each of these browsers. We generally encourage the use of, and fully support only, the latest version.

Note: If you use the Safari browser, be aware that it does not support the search feature for the guides on docs.edx.org. This is a known limitation.

1.4 Change Log

The edX documentation team no longer maintains a change log for each guide. For a weekly summary of platform changes, refer to the *EdX Release Notes* on the docs.edx.org website.

For information about changes made to the edX documentation set, the [edx-documentation](#) repository on GitHub provides a complete record.

Date	Change
12 January 2016	Added the Adding Custom Fields to the Registration Page topic.
16 December 2015	Updated the default value for the <code>ENABLE_INSTRUCTOR_LEGACY_DASHBOARD</code> feature flag in the Index of

Table 1.1 – continued from previous page

Date	Change
8 December 2015	Added the <i>Changing the Appearance of Open edX Sites</i> section.
1 December 2015	Added the <i>Prohibiting Submission of Specified File Types</i> section.
20 November 2015	Updated the manual command that can be used to <i>test an enabled SAML provider</i> .
9 November 2015	Added <i>Index of EdX Feature Flags</i> page.
4 November 2015	Added the <i>Installing an XBlock</i> topic.
26 October 2015	Added the <i>Using Elastic MapReduce on AWS</i> topic.
23 October 2015	Updated all topics in <i>Configuring the Open edX Platform</i> to instruct users to modify the <code>lms.env.json</code> and
	Removed the instructions to install the Google Drive XBlock, as it is now included by default in the Open edX
20 October 2015	Added the <i>Add Keys to the LMS Configuration File</i> topic.
5 October 2015	Added the <i>Enabling Social Sharing of Courses and Certificates</i> section.
2 October 2015	Added the <i>Options for LTI Authentication and User Provisioning</i> section.
15 September 2015	Added the <i>Configuring an edX Instance as an LTI Tool Provider</i> section.
	Added the <i>Installing Open edX Analytics Devstack</i> section.
10 August 2015	Added the <i>Open edX Cypress Release</i> section.
	Added the <i>Setting Up the YouTube API Key</i> section.
	Added the <i>Configuring ORA2 to Upload Files to Alternative Storage Systems</i> section.
6 Aug 2015	Added the <i>Enabling Third Party Authentication</i> section.
5 Aug 2015	Added the <i>Enabling Third Party Authentication with edX Edge</i> section.
16 June 2015	Added the <i>Enabling Custom Courses</i> section.
8 June 2015	Added <i>Enabling Open edX Search</i> .
	Added <i>Enabling Certificates</i> .
	Added <i>Enabling Badging</i> .
	Updated the <i>Setting Up the Open edX Mobile Applications</i> section to include configuration for push notification
28 May 2015	Added <i>Enabling Course and Video Licensing</i> .
02 Mar 2015	Updated the <i>Other edX Resources</i> to include information about the <i>The edX Partner Portal</i> and the <i>The Open edX</i>
24 Feb 2015	Updated for the <i>Open edX Birch Release</i> .
	Added the section <i>Configuring the Open edX Platform</i> .
20 Jan 2015	Added the section <i>Options for Installing edX Insights</i> .
14 Jan 2015	Added the section <i>Setting Up the Open edX Mobile Applications</i> .
07 Jun 2014	Added the section <i>Installing Open edX Fullstack</i> .
21 May 2014	The initial release of this guide, with the sections <i>Installing Open edX Devstack</i> and <i>Starting Open edX Devstack</i>

Open edX Platform Releases

The following sections provide information about releases of the Open edX platform.

2.1 Open edX Eucalyptus Release

This section describes how to install the Open edX Eucalyptus release.

- *What's Included in Eucalyptus*
- *What Is the Eucalyptus Git Tag?*
- *Installing the Eucalyptus Release*
- *Upgrading from Dogwood to Eucalyptus*
- *Upgrading to a Subsequent Eucalyptus Release*

2.1.1 What's Included in Eucalyptus

The Open edX Eucalyptus release contains several new features for learners, course teams, and developers. For more information, see [Open edX Eucalyptus Release](#).

2.1.2 What Is the Eucalyptus Git Tag?

A Git tag identifies the version of Open edX code that is the Eucalyptus release. You can find the most up-to-date Git tag for the current Open edX release on the [Open edX Releases Wiki page](#).

The following Open edX Git repositories have the Eucalyptus Git tag.

- edx-platform
- configuration
- cs_comments_service
- xqueue
- XBlock
- notifier
- edx-ora2
- edx-documentation

- edx-certificates
- edx-analytics-data-api-client
- edx-analytics-configuration
- edx-analytics-dashboard
- edx-analytics-data-api
- edx-analytics-pipeline

2.1.3 Installing the Eucalyptus Release

You can install the Open edX Eucalyptus release using *Devstack* or *Fullstack*.

Review the prerequisites and instructions for each option, and then choose the option that best meets your needs. Ensure that you install the required software to run the Open edX platform.

If you are upgrading from the Dogwood release, see *Upgrading from Dogwood to Eucalyptus*.

Eucalyptus releases have Git tag names like `open-release/eucalyptus.1`. The available names are detailed on the [Open edX Releases Wiki page](#).

2.1.4 Upgrading from Dogwood to Eucalyptus

You can upgrade an Open edX instance that is running the Dogwood release to the Eucalyptus release. EdX provides the `upgrade.sh` script if you have a simple Dogwood installation and want to upgrade it automatically. If you have a more complex or customized installation, you may need to upgrade manually.

The `upgrade.sh` script is in the edX configuration repository on GitHub.

Note: The upgrade script is only for upgrading instances running the Dogwood release. If your instance is running a release prior to the Dogwood release, follow the instructions to upgrade it to each intervening release, and then upgrade from Dogwood to Eucalyptus.

Caution: Before upgrading your Open edX instance, back up all data and configuration files. Then verify that you can restore your Open edX instance from the backup files.

On the computer or virtual machine that is running the Dogwood release of Open edX, run the upgrade script for your type of installation.

1. Download the script.

```
$ export OPENEDX_RELEASE=open-release/eucalyptus.1
$ curl -OL https://raw.githubusercontent.com/edx/configuration/$OPENEDX_RELEASE/util/vagrant/upgrade.sh
```

2. Run the script.

- For devstack, run `bash upgrade.sh -c devstack`.
- For fullstack, run `bash upgrade.sh -c fullstack`.

You can find the most up-to-date Git tag for the current Open edX release on the [Open edX Releases Wiki page](#).

You can also run `bash upgrade.sh -h` to see which other options the script accepts.

The script creates a temporary directory in which it upgrades Open edX, then cleans up extra files and directories when it finishes running.

After upgrading Open edX to the Eucalyptus release, start the LMS and Studio and verify that course content and data was migrated correctly.

2.1.5 Upgrading to a Subsequent Eucalyptus Release

Occasionally, we release updates to Eucalyptus. For example, the second release of Eucalyptus is `open-release/eucalyptus.2`. The steps to upgrade differ based on your original installation method.

Upgrading a Vagrant Installation

Devstack and Fullstack are installed using Vagrant. To upgrade to a Eucalyptus point release, follow these steps in the host operating system.

```
$ export OPENEDX_RELEASE=open-release/eucalyptus.2
$ vagrant provision
```

Upgrading a Native Installation

If you installed Open edX using the [Open edX Native Installation](#), re-run those steps using your desired Eucalyptus tag as the new value for `OPENEDX_RELEASE`.

2.2 Open edX Dogwood Release

This section describes how to install the Open edX Dogwood release.

- *What's Included in Dogwood*
- *What Is the Dogwood Git Tag?*
- *Installing the Dogwood Release*
- *Upgrading from Cypress to Dogwood*
- *Upgrading to a Dogwood Point Release*

2.2.1 What's Included in Dogwood

The Open edX Dogwood release contains several new features for learners, course teams, and developers. See the release notes for the [Open edX Dogwood Release](#) for more details.

2.2.2 What Is the Dogwood Git Tag?

A Git tag identifies the version of Open edX code that is the Dogwood release. You can find the most up-to-date Git tag for the current Open edX release on the [Open edX Releases Wiki page](#).

The following Open edX Git repositories have the Dogwood Git tag.

- `edx-platform`
- `configuration`
- `cs_comments_service`

- xqueue
- XBlock
- notifier
- edx-ora2
- edx-documentation
- edx-certificates
- edx-analytics-data-api-client
- edx-analytics-configuration
- edx-analytics-dashboard
- edx-analytics-data-api
- edx-analytics-pipeline

2.2.3 Installing the Dogwood Release

You can install the Open edX Dogwood release using *Devstack* or *Fullstack*.

Review the prerequisites and instructions for each option, and then choose the option that best meets your needs. Ensure that you install the required software to run the edX platform.

If you are upgrading from the Cypress release, see *Upgrading from Cypress to Dogwood*.

For new installations, follow these steps.

1. *Download the Vagrant Box* or *Download the BitTorrent File*.

Caution: The Vagrant boxes have a large file size (between 4 and 5 GB). If you have a slow or unreliable Internet connection, use BitTorrent to download the Vagrant box you need.

2. *Set the OPENEDX_RELEASE Environment Variable*.
3. *Install the Vagrant Box*.

Download the Vagrant Box

If you have a fast and reliable Internet connection, you can download the Vagrant box directly or by running `vagrant up` when you install *Devstack* or *Fullstack*.

To access the latest Vagrant boxes, see the [Open edX Releases Wiki page](#).

For more information about working with vagrant boxes, see [Vagrant's documentation on boxes](#).

Download the BitTorrent File

You can also download the BitTorrent file for the option you selected. BitTorrent is recommended if you have a slow or unreliable data connection. You then use the BitTorrent file to download the Vagrant box. If the Internet connection is temporarily lost while you are downloading the Vagrant box through BitTorrent, you can later continue the download without data loss or corruption.

To access the latest Vagrant box torrents, see the [Open edX Releases Wiki page](#).

For more information about downloading BitTorrent files, see [BitTorrent](#).

If you download the Vagrant box through BitTorrent, you must add the box to Vagrant before continuing with the installation process.

Be sure to verify that you have the most up-to-date Git tag for the Open edX releases on the [Open edX Releases Wiki page](#).

- For devstack installations, run the following command.

```
$ vagrant box add /{path-to-downloaded-box}/{vagrant-box-name} --name {Git-tag}
```

- For fullstack installations, run the following command.

```
$ vagrant box add /{path-to-downloaded-box}/{vagrant-box-name} --name {Git-tag}
```

Set the OPENEDX_RELEASE Environment Variable

Before installing the Vagrant box, you must set the value of the OPENEDX_RELEASE environment variable to the Git tag for the Dogwood release. To do so, use the Linux `export` command.

```
export OPENEDX_RELEASE="{Git tag}"
```

Install the Vagrant Box

When you have completed the previous steps, install the Dogwood release by following the installation instructions for *Devstack* or *Fullstack*.

2.2.4 Upgrading from Cypress to Dogwood

You can upgrade an Open edX instance that is running the Cypress release to the Dogwood release. EdX provides the `migrate.sh` script if you have a simple Cypress installation and want to upgrade it automatically. If you have a more complex or customized installation, you may need to upgrade manually.

Automatic Upgrading

The `migrate.sh` script is in the edX configuration repository on GitHub.

Note: The upgrade scripts provided are verified only for upgrading instances running the Cypress release. If you are running any other version of the Open edX Platform, the upgrade scripts might not work.

Caution: Before upgrading your Open edX instance, back up all data and configuration files. Then verify that you can restore your Open edX instance from the backup files.

On the computer or virtual machine that is running the Cypress release of Open edX, run the upgrade script for your type of installation.

- For devstack, run `./migrate.sh -c devstack -t named-release/dogwood`.
- For fullstack, run `./migrate.sh -c fullstack -t named-release/dogwood`.

You can find the most up-to-date Git tag for the current Open edX release on the [Open edX Releases Wiki page](#).

You can also run `./migrate.sh -h` to see which other options the script accepts.

The script creates a temporary directory in which it upgrades Open edX, then cleans up extra files and directories when it finishes running.

After upgrading Open edX to the Dogwood release, start the LMS and Studio and verify that course content and data was migrated correctly.

Upgrade Process Overview

This is an overview of what happens during an upgrade from Cypress to Dogwood. The `migrate.sh` script implements this process. You may need to understand this process if your installation is customized in some way, or if you need to diagnose problems during the upgrade.

Upgrading Cypress to Dogwood is more involved than most Open edX release upgrades.

- Dogwood upgrades the Django framework from version 1.4 to 1.8, which changed the database migration tool from South to Django. When you upgrade from Cypress to Dogwood, it is important to take special care with the database migrations.
- Dogwood upgrades Python from 2.7.3 to 2.7.10. This means virtualenvs have to be recreated.

The upgrade from Cypress to Dogwood includes these steps.

1. Applies a [forum migration described on the Open edX wiki](#) to support teams discussion filtering.
2. Updates edx-platform to the `release-2015-11-09` tag. This is the last released version that used Django 1.4.
3. Recreates the virtualenvs to use Python 2.7.10 instead of 2.7.3.
4. Migrates the database. This makes the database current with the last 1.4 code.
5. Uninstalls South so that it does not interfere with the new Django migrations.
6. Updates edx-platform to the `dogwood-first-18` tag. This is the first version of the code that used Django 1.8.
7. Applies all the initial Django migrations. This gets your database ready to use the new Django 1.8 migration mechanism.
8. Updates edx-platform to Dogwood.
9. Runs Django database migrations.
10. Runs two management commands to update records in the database: `generate_course_overview` and `post_cohort_membership_fix`.
11. Runs the forum migration again. This step processes any discussion topics created during the running of the script.

Similar steps are followed to upgrade other repositories such as xqueue.

2.2.5 Upgrading to a Dogwood Point Release

Occasionally, we release updates to Dogwood. The first of these is named Dogwood.1, then Dogwood.2, and so on. The steps differ based on your original installation method. You will need to know the name of the Dogwood tag you want to install, for example `named-release/dogwood.2`.

Upgrading a Vagrant Installation

Devstack and Fullstack are installed using Vagrant. To upgrade to a Dogwood point release, follow these steps in the host operating system.

```
$ export OPENEDX_RELEASE={desired-dogwood-tag}
$ vagrant provision
```

Upgrading a Native Installation

If you installed Open edX using the [Open edX Native Installation](#), re-run those steps using your desired Dogwood tag as the new value for OPENEDX_RELEASE.

2.3 Open edX Cypress Release

This section describes how to install the Open edX Cypress release.

- [What's Included in Cypress](#)
- [What is the Cypress Git Tag?](#)
- [Installing the Cypress Release](#)
- [Upgrading from Birch to Cypress](#)

2.3.1 What's Included in Cypress

The Open edX Cypress release contains several new features for learners, course teams, and developers. See the Open edX Release Notes for more details.

Note: There are several new features in the Cypress release that are available, but not enabled by default in new installations. For details, see the following topics.

- [Enabling Open edX Search](#)
 - [Enabling Badging](#)
 - [Enabling Custom Courses](#)
 - [Enabling Course and Video Licensing](#)
-

2.3.2 What is the Cypress Git Tag?

The Git tag for the Cypress release is named-release/cypress. You use this tag to identify the version of Open edX code that is the Cypress release.

The following Open edX Git repositories have the Git tag named-release/cypress.

- edx-platform
- configuration
- cs_comments_service

- notifier
- edx-certificates
- xqueue
- edx-documentation
- edx-ora2
- XBlock

2.3.3 Installing the Cypress Release

You can install the Open edX Cypress release using *Devstack* or *Fullstack*.

Review the prerequisites and instructions for each option, and then choose the option that best meets your needs. Ensure that you install the required software to run the edX Platform.

If you are upgrading from the Birch release, see *Upgrading from Birch to Cypress*.

For new installations, follow these steps.

1. *Download the Vagrant Box* or *Download the BitTorrent File*.

Caution: The Vagrant boxes have a large file size (about 2.5GB). If you have a slow or unreliable Internet connection, use BitTorrent to download the Vagrant box you need.

2. *Set the OPENEDX_RELEASE Environment Variable.*
3. *Install the Vagrant Box.*

Download the Vagrant Box

If you have a fast and reliable Internet connection, you can download the Vagrant box directly or by running `vagrant up` when installing *Devstack* or *Fullstack*.

See the [Open edX Releases Wiki page](#) to access the latest Vagrant boxes.

See [Vagrant's documentation on boxes](#) for more information.

Download the BitTorrent File

You can also download the BitTorrent file for the option you selected. BitTorrent is recommended if you have a slow or unreliable data connection. You then use the BitTorrent file to download the Vagrant box. If the Internet connection is temporarily lost while you are downloading the Vagrant box through BitTorrent, you can later continue the download without data loss or corruption.

See the [Open edX Releases Wiki page](#) to access the latest Vagrant boxes.

See [BitTorrent](#) for more information.

If you download the Vagrant box through BitTorrent, you must add the box to Vagrant before continuing with the installation process.

- For devstack installations, run the following command.

```
$ vagrant box add /path-to-downloaded-box/name-of-vagrant-box --name  
cypress-devstack
```

- For fullstack installations, run the following command.

```
$ vagrant box add /path-to-downloaded-box/name-of-vagrant-box --name  
cypress-fullstack
```

Set the OPENEDX_RELEASE Environment Variable

Before installing the Vagrant box, you must set the value of the OPENEDX_RELEASE environment variable to the Git tag for the Cypress release. Use the Linux `export` command.

```
export OPENEDX_RELEASE="named-release/cypress"
```

Install the Vagrant Box

When you have completed the previous steps, install the Cypress release by following the installation instructions for *Devstack* or *Fullstack*.

2.3.4 Upgrading from Birch to Cypress

You can upgrade an Open edX instance that is running the Birch release to the Cypress release, by using the `migrate.sh` script in the configuration repository, [available here](#).

Note: The upgrade scripts provided are verified only for upgrading instances running the Birch release. If you are running any other version of the Open edX Platform, the upgrade scripts might not work.

Caution: Before upgrading your Open edX instance, back up all data and configuration files. Then verify that you can restore your Open edX instance from the backup files.

On the computer or virtual machine that is running the Birch release of Open edX, run the upgrade script for your type of installation:

- For devstack, run `./migrate.sh -c devstack`.
- For fullstack, run `./migrate.sh -c fullstack`.
- You can also run `./migrate.sh -h` to see which other options the script accepts.

The script creates a temporary directory in which it upgrades Open edX, then cleans up extra files and directories when it finishes running.

After upgrading Open edX to the Cypress release, start the LMS and Studio and verify that course content and data was migrated correctly.

2.4 Open edX Birch Release

This section describes how to install the Open edX Birch release.

- [What's Included in Birch](#)
- [What is the Birch Git Tag?](#)
- [Installing the Birch Release](#)
- [Upgrading from Aspen to Birch](#)

Note: Now that the Open edX Cypress release is available, edX no longer supports the Birch release.

2.4.1 What's Included in Birch

The Open edX Birch release contains several new features for students, course teams, and developers. See the Open edX Release Notes for more details.

Note: There are several new features in the Birch release that are available, but not configured in new installations. For details, see the following topics.

- [Enabling Course Prerequisites](#)
 - [Enabling Entrance Exams](#)
-

2.4.2 What is the Birch Git Tag?

The Git tag for the Birch release is **named-release/birch.2**. You use this tag to identify the version of Open edX code that is the Birch release.

The following Open edX Git repositories have the Git tag **named-release/birch.2**:

- edx-platform
- configuration
- cs_comments_service
- notifier
- edx-certificates
- xqueue
- edx-documentation
- edx-ora2
- XBlock

2.4.3 Installing the Birch Release

You can install the Open edX Birch release using [Devstack](#) or [Fullstack](#).

Review the prerequisites and instructions for each option, then choose the option that best meets your needs. Ensure you install the required software to run the edX Platform.

If you are upgrading from the Aspen release, see [Upgrading from Aspen to Birch](#).

For new installations, follow the steps below.

1. *Download the Vagrant Box or Download the BitTorrent File.*

Caution: The Vagrant boxes have a large file size (about 2.5GB). If you have a slow or unreliable Internet connection, use BitTorrent to download the Vagrant box you need.

2. *Set the `OPENEDX_RELEASE` Environment Variable.*
3. *Install the Vagrant Box.*

Download the Vagrant Box

If you have a fast and reliable Internet connection, you can download the Vagrant box directly or by running `vagrant up` when installing *Devstack* or *Fullstack*.

See the [Open edX Releases Wiki](#) page to access the latest Vagrant boxes.

See [Vagrant's documentation on boxes](#) for more information.

Download the BitTorrent File

You can also download the BitTorrent file for the option you selected. BitTorrent is recommended if you have a slow or unreliable data connection. You then use the BitTorrent file to download the Vagrant box. If the Internet connection is temporarily lost while you are downloading the Vagrant box through BitTorrent, you can later continue the download without data loss or corruption.

See the [Open edX Releases Wiki](#) page to access the latest Vagrant boxes.

See [BitTorrent](#) for more information.

If you download the Vagrant box through BitTorrent, you must add the box to Vagrant before continuing with the installation process.

- For devstack installations, run the following command.

```
$ vagrant box add /path-to-downloaded-box/vagrant-images-birch-2-devstack.box --name birch-d
```

- For fullstack installations, run the following command.

```
$ vagrant box add /path-to-downloaded-box/vagrant-images-birch-2-fullstack.box --name birch-
```

Set the `OPENEDX_RELEASE` Environment Variable

Before installing the Vagrant box, you must set the value of the `OPENEDX_RELEASE` environment variable to the Git tag for the Birch release:

```
export OPENEDX_RELEASE="named-release/birch.2"
```

Install the Vagrant Box

When you have completed the previous steps, install the Birch release by following the installation instructions for *Devstack* or *Fullstack*.

2.4.4 Upgrading from Aspen to Birch

You can upgrade your Open edX instance that is running the Aspen release to the Birch release, using the `migrate.sh` script in the configuration repository, [available here](#).

Note: The upgrade scripts provided are verified only for upgrading instances running the Aspen release. If you are running any other version of the Open edX Platform, the upgrade scripts might not work.

Caution: Before upgrading your Open edX instance, back up all data and configuration files. Then verify that you can restore your Open edX instance from the backup files.

On the computer or virtual machine running the Aspen release of Open edX, run the upgrade script for your type of installation:

- For devstack, run `./migrate.sh -t named-release/birch.2 -c devstack`.
- For fullstack, run `./migrate.sh -t named-release/birch.2 -c fullstack`.
- You can also run `./migrate.sh -h` to see which other options the script accepts.

The script creates a temporary directory in which it upgrades Open edX, then cleans up extra files and directories when it finishes running.

After upgrading Open edX to the Birch release, run the edX Platform and verify that course content and data was migrated correctly.

Installing and Starting the Open edX Platform

The following sections provide information about how to install and start the latest version of the Open edX platform.

3.1 Open edX Platform Installation Options

This section describes Open edX installation options and the components that each option includes.

- *Open edX Platform Virtual Machines*
- *Virtual Machine Components*
- *Virtual Machine Configuration Options*

3.1.1 Open edX Platform Virtual Machines

You can install the Open edX developer stack (**devstack**), the Open edX full stack (**fullstack**), or the Open edX analytics developer stack (**analytics devstack**).

- Devstack is a Vagrant virtual machine instance designed for local development. For more information, see *Open edX Devstack*.
- Fullstack is a Vagrant virtual machine instance designed for deploying all edX services on a single server. For more information, see *Open edX Fullstack*.
- Analytics devstack is a modified version of the devstack virtual machine that allows you to run edX Analytics. For more information, see *Open edX Analytics Devstack*.

Open edX Devstack

Devstack is a Vagrant instance designed for local development. Devstack has the same system requirements as *Fullstack*. This allows you to discover and fix system configuration issues early in development.

Devstack simplifies certain production settings to make development more convenient. For example, `nginx` and `gunicorn` are disabled in devstack; devstack uses Django's runserver instead.

Devstack is in the [edx configuration repository](#) on GitHub.

For information about devstack and other installation and configuration options from edX and the Open edX community, see the [edx configuration repository wiki](#). Specifically, the following pages have more information about devstack.

- [Devstack wiki](#)
- [Developing on Devstack](#)

Note: Because of the large number of dependencies needed to develop extensions to Open edX Insights, a separate development environment is available to support Analytics development. For more information, see * [Installing and Starting Analytics Devstack](#).

For more information about Vagrant, see the [Vagrant documentation](#).

Open edX Fullstack

Fullstack is a Vagrant instance designed for deploying all edX services on a single server. Fullstack is in the [edx configuration repository](#) on GitHub.

For information about fullstack and other installation and configuration options from edX and the Open edX community, see the [edx configuration repository wiki](#).

For more information about Vagrant, see the [Vagrant documentation](#).

Ubuntu 12.04 64-bit

You can install fullstack on a single Ubuntu 12.04 64-bit server. More Ubuntu information is planned for future versions of this guide.

For information about Ubuntu and other installation and configuration options from edX and the Open edX community, see the [edx configuration repository wiki](#).

Open edX Analytics Devstack

Some users might want to develop Analytics features on their instance of the Open edX platform. Because of the large number of dependencies needed to develop extensions to Analytics, edX has created a separate developer stack, known as analytics devstack. We strongly recommend that you install the Analytics Devstack instead of adding Analytics extensions to an instance of devstack.

Analytics devstack is a modified version of the [Open edX developer stack](#). This development environment provides all of the services and tools needed to modify the Open edX Analytics Pipeline, Data API, and Insights projects.

3.1.2 Virtual Machine Components

Fullstack, devstack, and analytics devstack all include the following edX components.

- The Learning Management System (LMS).
- edX Studio.
- Discussion Forums.
- Open Response Assessments (ORA).

Devstack also includes the following edX components.

- A demonstration edX course.
- EdX Search.

Fullstack also includes the following edX components.

- XQueue, the queuing server that uses [RabbitMQ](#) for external graders.
- [Discern](#), the machine-learning-based automated textual classification API service.
- [Ease](#), a library for the classification of textual content.

Analytics devstack also includes the following edX components.

- edX Analytics Data API.
- edX Insights.
- The components needed to run the Open edX Analytics Pipeline. This is the primary extract, transform, and load (ETL) tool that extracts and analyzes data from the other Open edX services.

Default Accounts

When you install devstack, fullstack, or analytics devstack, the following user accounts are created by default.

Account	Description
staff@example.com	An LMS and Studio user with course creation and editing permissions. This user is a course team member with the Admin role, which gives rights to work with the demonstration course in Studio, the LMS, and Insights.
verified@example.com	A student account that you can use to access the LMS for testing verified certificates.
audit@example.com	A student account that you can use to access the LMS for testing course auditing.
honor@example.com	A student account that you can use to access the LMS for testing honor code certificates.

The password for all of these accounts is `edx`.

3.1.3 Virtual Machine Configuration Options

When you install devstack, fullstack, or analytics devstack you can customize the environment. This section provides information about configuration options for edX virtual machines.

Set Up Ability to Preview Units (Mac/Linux Only)

If you are installing an edX virtual machine on a Linux or Mac computer, you must configure your installation to use the preview feature in edX Studio.

1. *Connect to the Devstack virtual machine.*
2. In the `etc/hosts` file, add the following line.

<code>192.168.33.10 preview.localhost</code>
--

Customize the Source Code Location

You can customize the location of the edX source code that gets cloned when you provision an edX virtual machine. You may want to do this to have the edX virtual machine work with source code that already exists on your computer.

By default, the source code location is the directory in which you run `vagrant up`. To change this location, follow these steps.

1. *Connect to the Devstack virtual machine.*
2. Set the `VAGRANT_MOUNT_BASE` environment variable to set the base directory for the `edx-platform` and `cs_comments_service` source code directories.

3.2 Installation Prerequisites

To use devstack, fullstack, or analytics devstack, you must meet the following knowledge and software requirements.

3.2.1 Knowledge Prerequisites

Devstack, fullstack, and analytics devstack require an understanding of the following items.

- Basic terminal usage. If you are using a Mac computer, see [Introduction to the Mac OS X Command Line](#). If you are using a Windows computer, see [Windows Command Line Reference](#).
- Vagrant commands. See the [Vagrant Getting Started](#) guide for more information.
- Diagnosing and fixing failures may involve many different technologies and skills. It will help to know these things.
 - The basics of how Python web applications are built, installed and deployed.
 - How to manage a Linux system, including supervisor.
 - The basics of configuration management and automation. We use [Ansible](#) to automate the installation process.

3.2.2 Software Prerequisites

Devstack, fullstack, and analytics devstack require the following software.

- [VirtualBox](#) 4.3.12 or later.
- [Vagrant](#) 1.6.5 or later.
- A Network File System (NFS) client, if your operating system does not include one. Devstack uses VirtualBox Guest Editions to share folders through NFS.

3.3 Getting Help

This section describes resources for getting help if you need technical assistance during the installation process or after your site is running.

3.3.1 Where to Find Help

There are a number of places to get help, including mailing lists and real-time chat. Please choose an appropriate venue for your question. This helps ensure that you get good prompt advice, and keeps discussion focused. For details of your options, see the [Getting Help](#) page.

3.3.2 Information to Provide When Asking for Help

Keep in mind when asking for help that you have much more knowledge of your situation than others do. You need to provide that context so that people can understand your problem and provide good help. To make it easier for others to help you, please provide the following types of information.

- **What you are trying to do?** What version of the code are you installing, and why? What is your larger goal?
- **What you have done?** What instructions were you following? What commands did you run? Did you deviate from the instructions at any point, even a little bit? Full output of those commands, even if it seems overwhelming, can be very useful.
- **What has gone wrong?** Are there errors in log files? Did you get specific failures in the terminal? Provide full details about the undesired results you saw.

While working through your problem, helpers often need additional information. Stay involved in the discussion, and try to give them what they need. They know best what information they need to solve the problem.

3.4 Installing and Starting Devstack

The following sections provide information about how to install and start devstack.

3.4.1 Installing Open edX Devstack

This section describes how to install the Open edX developer stack (devstack).

- *Installation Prerequisites for Devstack*
- *Install Devstack*

Installation Prerequisites for Devstack

Before you install devstack, make sure that you have met the *installation prerequisites*.

You must also ensure that you have the administrator password for your local computer. The administrator password is needed to configure NFS (network file system) to allow users to access code directories directly from your computer.

Selecting a Download Option

The Open edX virtual machine box file has a file size of approximately four gigabytes. To download the box file, you can use one of these methods.

- Install devstack with a direct Vagrant box download. The first time you start the devstack virtual machine, the Vagrant virtual machine management tool downloads the box file.
- Use a BitTorrent client to download the Vagrant box file before you install devstack. The first time you start the devstack virtual machine, the Vagrant virtual machine management tool uses the previously downloaded box file.

If your internet connection is limited or intermittent, edX recommends that you torrent the box file. However, this step is optional.

Torrent the Box File (Optional)

1. Download the torrent file for the latest Open edX release. For the URLs and box names of the Open edX releases, see [Open edX Releases Wiki page](#).

```
curl -OJL https://s3.amazonaws.com/edx-static/vagrant-images/eucalyptus-devstack-2016-08-19.box?
```

2. Use a BitTorrent client to open the `.torrent` file and download the `.box` file. For more information about using a BitTorrent client, see the documentation for the client software that you choose.
3. Find the name of the Vagrant box for the Open edX release that you are installing. To find the Vagrant box name for an Open edX release, see [Open edX Releases Wiki page](#).
4. Create a Vagrant box using the `vagrant box add` command. Specify the name of the box and the file path of the box file that you downloaded. The name of the box you create must match the box name for the Open edX release you are installing.

```
vagrant box add {BOX-NAME} {BOX-FILE}
```

Where `{BOX-NAME}` is the name of the box specified on the Open edX Releases Wiki page, and `{BOX-FILE}` is the path to the box file you downloaded. For example, the following command creates a box named `eucalyptus-devstack-2016-08-19`.

```
vagrant box add eucalyptus-devstack-2016-08-19 /path/to/box/eucalyptus-devstack-2016-08-19.box
```

Install Devstack

To install devstack, follow these steps.

1. Ensure the `nfsd` client is running. You can use the `nfsd status` command.

```
sudo nfsd status
nfsd service is enabled
nfsd is running (pid 313, 8 threads)
```

If the `nfsd` service is not running, use the `nfsd start` command.

```
sudo nfsd start
Starting the nfsd service
```

2. Create a `devstack` directory and navigate to it in the command prompt.

```
mkdir devstack
cd devstack
```

3. Set the `OPENEDX_RELEASE` environment variable to the Git tag name of the release of the Open edX platform that you are installing. For information about the latest Open edX releases and the Git tag names for them, see [Open edX Releases Wiki page](#).

For example, `open-release/eucalyptus.1` is the Git tag name for the first Eucalyptus release. The following command sets the value of the `OPENEDX_RELEASE` environment variable to `open-release/eucalyptus.1`.

```
export OPENEDX_RELEASE="open-release/eucalyptus.1"
```

If you do not set this environment variable, Vagrant will install the most recent snapshot version of the Open edX platform. The snapshot version is not a supported release.

4. Download the install script.


```
curl -OL https://raw.githubusercontent.com/edx/configuration/$OPENEDX_RELEASE/util/install/install_stack.sh
```

5. Run the install script to create and start the devstack virtual machine.

```
bash install_stack.sh devstack
```

Note: This step downloads the box file if you did not already *use Torrent* to download and add it.

If you destroy and recreate the virtual machine, Vagrant reuses the box file and does not download it again. See [Vagrant's documentation on boxes](#) for more information.

When prompted, enter the administrator password for your local computer.

When you have completed these steps, see [Starting Open edX Devstack](#) to begin using devstack.

For help with the devstack installation, see [Troubleshooting Devstack Installation](#).

3.4.2 Starting Open edX Devstack

This section describes how to start the components in the Open edX developer stack (devstack). To do this, you must connect to the devstack virtual machine, and then start each component.

- [Connecting to the Devstack Virtual Machine](#)
- [Starting the Components](#)

Connecting to the Devstack Virtual Machine

1. To connect to the devstack virtual machine, use the following SSH command from the devstack directory.

```
vagrant ssh
```

2. To connect as the **edxapp** user, run the following command.

```
sudo su edxapp
```

This command loads the edxapp environment from the file `/edx/app/edxapp/edxapp_env`. This puts `venv python` in your search path.

This command also sets the current working directory to the edx-platform repository (`/edx/app/edxapp/edx-platform`).

Starting the Components

After you connect to the devstack virtual machine as the **edxapp** user, you can start the individual components.

Starting the LMS

When you run the LMS on devstack, the command updates requirements and compiles assets, unless you use the `fast` option.

The command uses the file `lms/envs/devstack.py`. This file overrides production settings for the LMS.

To run the LMS on devstack, follow these steps.

1. *Connect to the Devstack virtual machine.*
2. Run the following command.

```
paver devstack lms
```

Or, to start the LMS without updating requirements and compiling assets, use the `fast` option.

```
paver devstack lms --fast
```

The LMS starts.

3. Open the LMS in your browser at `http://localhost:8000/`.

Vagrant forwards port 8000 to the LMS server running in the virtual machine.

Starting Studio

When you run Studio on devstack, the command updates requirements and compiles assets, unless you use the `fast` option.

You run Studio on devstack with the file `cms/envs/devstack.py`. This file overrides production settings for Studio.

To run Studio on devstack, follow these steps.

1. *Connect to the Devstack virtual machine.*
2. Run the following command.

```
paver devstack studio
```

Or, to start Studio without updating requirements and compiling assets, use the `fast` option.

```
paver devstack studio --fast
```

Studio starts.

3. Open Studio in your browser at `http://localhost:8001/`.

Vagrant forwards port 8001 to the Studio server running in the virtual machine.

Viewing Available Studio Commands To view all available commands for Studio, enter the following command.

```
./manage.py cms -h --settings=devstack
```

Starting Course Discussions

To run course discussions on devstack, follow these steps.

1. *Connect to the Devstack virtual machine.*
2. Switch to the discussion forum account by entering the following command.

```
sudo su forum
```

3. Update Ruby requirements.

```
bundle install
```

Note: If you get a message for entering a password to install the bundled RubyGems to the system, you can safely exit by entering `control+c` on a Mac or `Ctrl+C` on Windows. The RubyGems will still be installed correctly for the forum user.

4. Start the discussion forums server.

```
ruby app.rb -p 18080
```

The discussions forum server starts. You can access the discussion forums API at `http://localhost:18080/`.

3.4.3 Troubleshooting Devstack Installation

In some cases, you see an error when you attempt to create an edX virtual machine (`vagrant up`). For example:

```
mount.nfs: mount to NFS server '192.168.33.1:/path/to/edx-platform' failed:
timed out, giving up
```

This error situation arises because Vagrant uses a host-only network in Virtualbox to communicate with your computer. If a network does not exist, one is created on `vagrant up`. If this network is created with the VPN up, it will not work. You must recreate the network with the VPN down.

To resolve the error, follow these steps.

1. Stop the VPN.
2. Type `vagrant halt`.
3. Open Virtualbox.
4. Navigate to **Preferences > Network > Host-only Networks** and remove the most-recently-created host-only network.
5. Type `vagrant up`.

3.5 Installing and Starting Fullstack

The following sections provide information about how to install and start fullstack.

3.5.1 Installing Open edX Fullstack

This section describes how to install the Open edX full stack (fullstack).

- *Installation Prerequisites for Fullstack*
- *Install Fullstack*

Installation Prerequisites for Fullstack

Before you install Open edX fullstack, make sure that you have met the *installation prerequisites*.

Selecting a Download Option

The Open edX virtual machine box file has a file size of approximately four gigabytes. To download the box file, you can use one of these methods.

- Install fullstack with a direct Vagrant box download. The first time you start the fullstack virtual machine, the Vagrant virtual machine management tool downloads the box file.
- Use a BitTorrent client to download the Vagrant box file before you install fullstack. The first time you start the fullstack virtual machine, the Vagrant virtual machine management tool uses the previously downloaded box file.

If your internet connection is limited or intermittent, edX recommends that you torrent the box file by following the instructions in *Torrent the Box File (Optional)*. However, this step is optional.

Install Fullstack

To install fullstack follow these steps.

1. Create the `fullstack` directory and navigate to it in the command prompt.

```
mkdir fullstack
cd fullstack
```

2. Set the `OPENEDX_RELEASE` environment variable to the Git tag name of the release of the Open edX platform that you are installing. For information about the latest Open edX releases and the Git tag names for them, see the [Open edX Releases Wiki page](#).

For example, `open-release/eucalyptus.1` is the Git tag name for the first Eucalyptus release. The following command sets the value of the `OPENEDX_RELEASE` environment variable to `open-release/eucalyptus.1`.

```
export OPENEDX_RELEASE="open-release/eucalyptus.1"
```

3. Download the install script.

```
curl -OL https://raw.githubusercontent.com/edx/configuration/$OPENEDX_RELEASE/util/install/install_stack.sh
```

4. Run the install script to create and start the fullstack virtual machine.

```
bash install_stack.sh fullstack
```

Note: This step downloads the box file if you did not already *use Torrent* to download and add it.

If you destroy and recreate the virtual machine, Vagrant reuses the box file and does not download it again. See [Vagrant's documentation on boxes](#) for more information.

When you have completed these steps, see *Starting Open edX Fullstack in a Browser* to begin using fullstack.

3.5.2 Starting Open edX Fullstack in a Browser

In your browser, go to `preview.localhost`, which is an alias entry for `192.168.33.10` that was created in your `/etc/hosts` file.

The latest version of fullstack is preloaded with the demonstration course and a set of *default user accounts*.

3.5.3 Troubleshooting Fullstack Installation

In some cases, you see an error when you attempt to create an edX virtual machine (`vagrant up`). For example:

```
mount.nfs: mount to NFS server '192.168.33.1:/path/to/edx-platform' failed:
timed out, giving up
```

This error situation arises because Vagrant uses a host-only network in Virtualbox to communicate with your computer. If a network does not exist, one is created on `vagrant up`. If this network is created with the VPN up, it will not work. You must recreate the network with the VPN down.

To resolve the error, follow these steps.

1. Stop the VPN.
2. Type `vagrant halt`.
3. Open Virtualbox.
4. Navigate to **Preferences > Network > Host-only Networks** and remove the most-recently-created host-only network.
5. Type `vagrant up`.

3.6 Installing and Starting Analytics Devstack

The following sections provide information about how to install and start analytics devstack.

3.6.1 Installing Open edX Analytics Devstack

This section describes how to install and run the Open edX Analytics developer stack.

Note: Before you install analytics developer stack, make sure that you have met the *installation prerequisites*.

Installing Analytics Devstack

To install analytics devstack extensions directly from the command line, follow these steps.

1. Halt any running Open edX devstacks. Navigate to the directory that contains the Vagrantfile for the devstack and run `vagrant suspend` or `vagrant halt`.

```
$ cd ~/open-edx/devstack/
$ vagrant suspend
```

2. Create the `analyticstack` directory and navigate to it in the command prompt.

```
$ mkdir analyticstack
$ cd analyticstack
```

3. Download the analytics devstack Vagrant file.

```
$ curl -L https://raw.githubusercontent.com/edx/configuration/open-release/eucalyptus.master/vag
```

4. Create the analytics devstack virtual machine.

```
$ export OPENEDX_RELEASE="open-release/eucalyptus.2"
$ vagrant up
```

5. Clone the edx-analytics-pipeline repository.

```
$ git clone git@github.com:edx/edx-analytics-pipeline.git ./edx-analytics-pipeline/
```

6. Prepare the data pipeline inside the virtual machine.

```
$ vagrant ssh
$ cd /edx/app/analytics_pipeline/
$ sudo mkdir venvs
$ sudo chown vagrant:vagrant venvs
$ virtualenv venvs/analytics_pipeline/
$ . venvs/analytics_pipeline/bin/activate
$ cd analytics_pipeline
$ make system-requirements
$ make develop
```

Note: The version of edx-analytics-pipeline that you checked out on your host will be mounted at /edx/app/analytics_pipeline/analytics_pipeline inside the virtual machine. Vagrant directory sharing allows the code to be modified using an editor on the host machine and executed within the virtual machine.

7. Run tests and quality checks.

```
$ make coverage
```

8. Run the acceptance tests as the hadoop user.

```
$ sudo su hadoop
$ cd /edx/app/analytics_pipeline/
$ . venvs/analytics_pipeline/bin/activate
$ cd analytics_pipeline

# The next step will take hours to run.
$ make test-acceptance-local
```

A subset of acceptance tests can be run using the ONLY_TESTS parameter.

```
$ make test-acceptance-local ONLY_TESTS=edx.analytics.tasks.tests.acceptance.test_enrollments
```

Acceptance tests usually destroy any existing state before running. This behavior can be disabled by setting the DISABLE_RESET_STATE environment variable.

```
$ DISABLE_RESET_STATE=true make test-acceptance-local ONLY_TESTS=edx.analytics.tasks.tests.acceptance.test_enrollments
```

Note: Acceptance tests emulate deployment of the code to a remote Hadoop cluster. During this process the tests check out a new copy of the code from the repo. For this reason, all changes must be committed before running the test.

9. Display parameters for a task. You can use the following technique to see the parameters for any task.

```
$ launch-task ImportEnrollmentsIntoMysql --help
```

The [EdX Analytics Pipeline Reference Guide](#) contains a more detailed list of available tasks and their parameters.

3.6.2 Starting Open edX Analytics Devstack

This section describes how to start the Open edX Analytics developer stack (analytics devstack).

- *Starting the Open edX LMS*
- *Starting the Open edX Analytics Data API*
- *Starting Open edX Insights*
- *Starting the Open edX Analytics Pipeline*

Starting the Open edX LMS

1. Log in to analytics devstack.

```
$ vagrant ssh
```

2. Switch to the `edxapp` user.

```
$ sudo su edxapp
```

3. Start the LMS.

```
$ paver devstack lms
```

Starting the Open edX Analytics Data API

1. Log in to analytics devstack.

```
$ vagrant ssh
```

2. Switch to the `analytics_api` user.

```
$ sudo su analytics_api
```

3. Start the Data API.

```
$ ~/venvs/analytics_api/bin/python ~/analytics_api/manage.py runserver 0.0.0.0:8100 --insecure
```

Starting Open edX Insights

1. Log in to analytics devstack.

```
$ vagrant ssh
```

2. Switch to the `insights` user.

```
$ sudo su insights
```

3. Enable features that are disabled by default.

```
$ ~/venvs/insights/bin/python ~/edx_analytics_dashboard/manage.py switch display_verified_enroll
$ ~/venvs/insights/bin/python ~/edx_analytics_dashboard/manage.py switch enable_course_api on --
```

For a complete list of the waffle switches that are available, see [Feature Gating](#).

4. Start Insights.

```
$ ~/venvs/insights/bin/python ~/edx_analytics_dashboard/manage.py runserver 0.0.0.0:8110 --insec
```

5. Open the URL `http://127.0.0.1:8110` in a browser on the host.

Important: Be sure to use the IP address `127.0.0.1` instead of `localhost`. Using `localhost` will prevent you from logging in.

Starting the Open edX Analytics Pipeline

1. In the Devstack LMS, register a new user and enroll in the demo course.
2. Navigate to the course body and submit answers to a few problems.
3. Navigate to the location where `edx-analytics-pipeline` project was cloned on the host.

```
$ cd edx-analytics-pipeline
```

4. Run the enrollment task.

```
$ export WHEEL_URL=http://edx-wheelhouse.s3-website-us-east-1.amazonaws.com/Ubuntu/precise
# On Mac OS X replace the date command below with $(date -v+1d +%Y-%m-%d)
$ remote-task --vagrant-path <path to `analyticstack`> --remote-name devstack --override-config
  ImportEnrollmentsIntoMysql --local-scheduler --interval-end $(date +%Y-%m-%d -d "tomorrow") --
```

5. Run the answer distribution task.

```
$ export WHEEL_URL=http://edx-wheelhouse.s3-website-us-east-1.amazonaws.com/Ubuntu/precise
$ export UNIQUE_NAME=$(date +%Y-%m-%dT%H_%M_%SZ)
$ remote-task --vagrant-path <path to `analyticstack`> --remote-name devstack --override-config
  AnswerDistributionWorkflow --local-scheduler \
    --src hdfs://localhost:9000/data/ \
    --include '*tracking.log*' \
    --dest hdfs://localhost:9000/edx-analytics-pipeline/output/answer_distribution_raw/$UNIQUE
    --name $UNIQUE_NAME \
    --output-root hdfs://localhost:9000/edx-analytics-pipeline/output/answer_distribution/ \
    --marker hdfs://localhost:9000/edx-analytics-pipeline/output/answer_distribution_raw/$UNIQUE
    --n-reduce-tasks 1
```

3.6.3 Troubleshooting Analytics Devstack Installation

In some cases, you see an error when you attempt to create an edX virtual machine (`vagrant up`). For example:

```
mount.nfs: mount to NFS server '192.168.33.1:/path/to/edx-platform' failed:
timed out, giving up
```

This error situation arises because Vagrant uses a host-only network in Virtualbox to communicate with your computer. If a network does not exist, one is created on `vagrant up`. If this network is created with the VPN up, it will not work. You must recreate the network with the VPN down.

To resolve the error, follow these steps.

1. Stop the VPN.
2. Type `vagrant halt`.
3. Open Virtualbox.
4. Navigate to **Preferences > Network > Host-only Networks** and remove the most-recently-created host-only network.
5. Type `vagrant up`.

Configuring the Open edX Platform

The following sections provide information about Open edX Platform configuration options.

4.1 Guidelines for Updating the Open edX Platform

When you update the Open edX Platform, you should not change configuration files on a running server. Doing so can result in unpredictable problems.

If you need to change settings on a running server, take the following steps.

1. Provision a new server that matches the running server.
2. Make configuration changes on the new server.
3. Start the new server.
4. Reroute traffic from the old server to the new server.
5. Decommission the old server.

4.2 Configuring Open edX Sites

By default, an Open edX installation has one site for users to interact with. You can configure multiple sites within your Open edX installation. A site presents Open edX courses and content in an individual way. If you set up multiple sites, you can configure them independently of each other. For example, you can assign a different theme to each site and specify which courses are available on each site.

You host each site on a separate domain or subdomain from your Open edX installation. You configure the domain name for a site in the Django admin site when you create it. For example, you might configure one site with the domain name `university.edu` and another site with the domain name `advancedplacement.edu`. Or you might configure one site with the domain name `arts.myuniversity.edu` and another site with the domain name `sciences.myuniversity.edu`.

4.2.1 Create an Open edX Site

To create an Open edX site, follow these steps.

1. Sign in to the Django administration console for your base URL. For example, `http://{your_URL}/admin`.
2. Select **Sites** to open the `http://{your_URL}/admin/sites/site` page.

3. Enter the domain name for the site. This is the domain name in the URL for the site. For example, `myuniversity.edu`.

To make a site available on a non-default port that is entered as part of the URL, the domain name must include the port number. For example, if an LMS site is available at `my-site.localhost:8000`, the domain name for the site must be `my-site.localhost:8000`.

4. Select **Save**.

4.2.2 Configuring Sites Independently

You can set configuration properties independently for individual sites. The values that you define for individual sites override the default values that are present in the `cms.env.json` or `lms.env.json` files. For example, you can set the `PLATFORM_NAME` property to a different value for each of your sites to indicate that the sites present courses for different organizations or audiences.

Configuring a Site

To set configuration properties for a site, follow these steps.

1. Sign in to the Django administration console for your base URL. For example, `http://{your_URL}/admin`.
2. Select **Site Configurations**.
3. Select **Add site configuration**.
4. From the **Site** menu, select the site you want to configure.
5. Enter configuration properties in the **Values** field. Structure all properties in valid JavaScript Object Notation (JSON) format.

The following example shows a set of configuration properties for a site.

```
{
  "course_email_from_addr": "courses@onlineu.edu",
  "university": "Online University",
  "PLATFORM_NAME": "Online University",
  "email_from_address": "courses@onlineu.edu",
  "payment_support_email": "payments@onlineu.edu",
  "SITE_NAME": "onlineu.edu",
  "site_domain": "onlineu.edu",
  "SESSION_COOKIE_DOMAIN": "onlineu.edu"
}
```

Note: To make courses site-specific, you set the `course_org_filter` property to an organization identifier. Only that organization's courses are available from the site.

6. When you are ready for the configuration settings to take effect, select **Enabled**.

The configuration properties that you set do not affect the site until you select **Enabled**. If needed, you can return to the **Site configurations** screen for this site to enable the configuration properties later.

7. Select **Save**.

Site Configuration Reference

An example of the properties that you define to configure a site follows.

```
{
  "ECOMMERCE_API_URL": "https://my-site.sandbox.edx.org/api/v2",
  "ECOMMERCE_PUBLIC_URL_ROOT": "https://my-site.sandbox.edx.org",
  "ECOMMERCE_API_SIGNING_KEY": "ecommerce-secret",
  "COURSE_CATALOG_VISIBILITY_PERMISSION": "see_in_catalog",
  "COURSE_ABOUT_VISIBILITY_PERMISSION": "see_about_page",
  "ENABLE_COMBINED_LOGIN_REGISTRATION": true,
  "ENABLE_PAID_COURSE_REGISTRATION": true,
  "ENABLE_SHOPPING_CART": true,
  "ENABLE_SHOPPING_CART_BULK_PURCHASE": false,
  "course_email_template_name": "my-site",
  "course_email_from_addr": "my-site@example.com",
  "ALLOW_AUTOMATED_SIGNUPS": true,
  "domain_prefix": "my-site",
  "university": "Education Programs",
  "PLATFORM_NAME": "Education Programs",
  "platform_name": "Education Programs",
  "show_only_org_on_student_dashboard": true,
  "email_from_address": "my-site@example.com",
  "payment_support_email": "payments@example.com",
  "SITE_NAME": "my-site.sandbox.edx.org",
  "site_domain": "my-site.sandbox.edx.org",
  "SESSION_COOKIE_DOMAIN": "my-site.sandbox.edx.org",
  "course_org_filter": "MyOrgX",
  "course_index_overlay_text": "<img src='/static/my-site/images/400x103.png' width='400' height='103' />",
  "homepage_overlay_html": "<img src='/static/my-site/images/400x103.png' width='400' height='103' />",
  "payment_email_signature": "Education Programs<br>The Digital Programs Team<br>my-site@example.com<br>"
}
```

For more information about themes, see *Changing Themes for an Open edX Site*.

4.3 Changing the Appearance of Open edX Sites

This section describes how you can customize your Open edX sites to change the way they look.

4.3.1 Changing Themes for an Open edX Site

The theme for a website defines the appearance of its user interface (UI): the logo, the color scheme, and the links in the page headers and footers are examples of different aspects of an Open edX site that are defined by its theme.

Open edX provides a default theme that is defined by page templates, CSS styling, and assets such as images that are provided in the Open edX code. You can change the appearance of the following parts of an Open edX site.

- The Studio UI, which is used by course teams.
- The learning management system (LMS) UI, which is used by learners and course teams.
- The UI of the E-commerce service, which is used by course offering and order managers.

The topics in this section describe how you can change the way an Open edX site looks, without changing how it works.

Themes Overview

After you configure one or more sites for your Open edX installation, including their domain and platform names, you can set up a theme to use for each site.

To override the files that constitute the default Open edX theme, you create replacements for one or more of those files, place them in file paths that are constructed and named in parallel to the default file locations, configure your Open edX instance to use the files in your theme's directories instead of the default locations, and then compile the theme.

EdX first looks for files in your theme directories, and uses any file that matches the exact file path and file name of a default UI file. If matching files are not found, then Open edX looks in the default location.

For more information about Open edX sites, see [Configuring Open edX Sites](#).

Root Directories for Theme Files

Themes are located outside of the Open edX source directories, in any location you like. For example, you might make a directory called `/my-open-edx-themes`.

Within that directory, you create a separate directory for each Open edX repository that you want to create a theme for, such as `edx-platform` and `ecommerce`.

Within each of those directories, you create another directory and name it for your theme, such as `my-theme`. You can create a number of themes, each with their own name. Within each theme directory, you create directories and files to parallel the structure in the corresponding Open edX repository.

After you create these directories, you might have a structure like this one.

```
my-open-edx-themes
-- ecommerce
|   -- my-theme
-- edx-platform
|   -- my-theme
|       -- cms
|       -- lms
```

You must give the files that you create for a theme the same relative file paths and file names as the default files that they override. Different root directories for the relative paths apply to Studio, the LMS, and the E-commerce service.

- For Studio and the LMS, relative file paths are from the root directory of the local clone of the `edx/edx-platform` repository in your installation directory.
- For the E-commerce service, relative file paths are from the `ecommerce` directory of the local clone of the `edx/ecommerce` repository in your installation directory.

For example, the root directory for the relative file paths of your theme files might be at one of the following file paths.

- For the LMS UI or Studio UI, `/edx/app/edxapp/edx-platform`.
- For the UI of the E-commerce service, `/edx/app/ecommerce/ecommerce/ecommerce`.

The following subdirectories hold the UI files that you can override.

- `static` holds media files such as images and styling resources such as Syntactically Awesome Style Sheet (Sass) files that produce Cascading Style Sheet (CSS) files.
- `templates` holds Python web application page templates that produce the HTML for UI pages.

Creating a Theme

To create a theme, you add a theme directory to the installation-wide themes directory that you added when you enabled theming for your Open edX installation. Then, you copy default UI files into your theme directory and update them to change the appearance of the Open edX site or sites that will use that theme.

- *Understanding Which UI Files to Customize*
- *Example File Path for a Theme File*
- *Naming a Theme Directory*

For more information about the themes directory, see *Enabling and Applying Themes*.

Understanding Which UI Files to Customize

You can customize the default images, Sass files, and web application template files for the Open edX components in your installation.

- To replace an image, you can override any of the images in the `static/images` directories for the components you are theming.
- For the LMS and Studio, you can customize any of the Sass files in the `static/sass` directories.
- For the E-commerce service, you can customize any of the Sass files in the `static/sass/partials` directory.

Note: Do not customize Sass files in the `static/sass/base` directory.

- You can override any of the HTML templates in the `lms/templates` or `cms/templates` directories for the components that you want to apply a theme to.

UI files for the LMS are stored in the directories shown in the following example theme directory. You can examine the default UI files in the source repository of the component that you want to apply the theme to.

```
/my-open-edx-themes/edx-platform/my-theme/lms/static/images  
/my-open-edx-themes/edx-platform/my-theme/lms/static/sass  
/my-open-edx-themes/edx-platform/my-theme/lms/templates
```

Example File Path for a Theme File

The default Open edX theme includes an image file named `logo.png` that appears in the header of most LMS pages. The file path of that image in the `edx-platform` repository is `lms/static/images/logo.png`.

The following example shows an absolute file path of the LMS logo image in a theme directory. The file path after `/my-open-edx-themes/edx-platform/my-theme/` matches the relative file path of that image in the default directory for the LMS UI.

```
/my-open-edx-themes/edx-platform/my-theme/lms/static/images/logo.png
```

Naming a Theme Directory

The name of the directory that you create to hold your versions of the image, theme, and Sass styling files identifies the theme. As a result, if you want to create a theme named `my-theme`, the name of the directory within your

installation-wide themes directory must be `my-theme`.

Note: If you add more than one site-specific theme directory, make sure that each of the directory names is unique. The Open edX installation looks for a theme with a certain name, and selects the first one that matches.

Because the UI files for the LMS and Studio are located together in the `edx-platform` repository, you can create one theme that applies to both the LMS and Studio. If you want to create a theme for the E-commerce service, you must add a separate theme directory for its files.

The theme directory holds the UI files that override the corresponding default files.

For example, if the themes directory for your site is `/my-open-edx-themes`, the files in the following example create a theme named `my-theme`.

```
/my-open-edx-themes/edx-platform/my-theme/lms/static/images/logo.png
/my-open-edx-themes/edx-platform/my-theme/lms/static/sass/partial/base/_variables.scss
/my-open-edx-themes/edx-platform/my-theme/lms/templates/navigation.html
/my-open-edx-themes/edx-platform/my-theme/cms/static/images/studio-logo.png
/my-open-edx-themes/edx-platform/my-theme/cms/static/images/logo.png
/my-open-edx-themes/edx-platform/my-theme/cms/templates/login.html
```

Because the theme directory includes UI files in both the `lms` and `cms` subdirectories, you can apply the theme to both the LMS and Studio.

Note: After you create or make changes to a theme, you must update the theme. Updating a theme compiles Sass files to create the CSS files that style your UI. For more information, see [Compiling a Theme](#).

Enabling and Applying Themes

You must enable the use of themes for your Open edX installation before you can apply themes to your Open edX sites. If your installation has only one site, you apply a theme to that default site.

For more information about Open edX sites, see [Configuring Open edX Sites](#).

- [Enable Themes](#)
- [Apply a Theme to a Site](#)

Enable Themes

To enable the use of themes for your Open edX installation, follow these steps.

1. Create an installation-wide themes directory to hold the customized UI files for all of the themes that you create. This directory will hold subdirectories for each theme. Your Open edX installation will look in the directory to find the themes you apply to sites.

You can create this directory at any location on a file system that is accessible to your Open edX installation. For example, you might place it at the root of the file system in a directory named `/my-open-edx-themes`.

2. Set the file permissions on the themes directory, and all of its subdirectories, to enable read+write permissions for the Ubuntu user.

For example, to allow the `devstack edxapp` user for the LMS and Studio to read and write files in the themes directory, you might use the following commands.


```
sudo chown -R edxapp:edxapp /my-open-edx-themes
sudo chmod -R u+rw /my-open-edx-themes
```

On fullstack and native installations the Ubuntu user is `www-data`.

3. For each Open edX component that you want to theme, set the `ENABLE_COMPREHENSIVE_THEMING` configuration property to `true`.

The specific method that you use to configure Open edX components depends on the type of environment you are using. For example, you can set the configuration property in the following files.

- For the LMS, you edit `/edx/app/edxapp/lms.env.json` to set `"ENABLE_COMPREHENSIVE_THEMING": true`.
- For Studio, you edit `/edx/app/edxapp/cms.env.json` to set `"ENABLE_COMPREHENSIVE_THEMING": true`.
- For the E-commerce service, you edit `/edx/etc/ecommerce.yml` to set `ENABLE_COMPREHENSIVE_THEMING: true`.

If any of these files do not exist, you can add them to define this configuration setting.

4. For each Open edX component that you want to apply a theme to, add the absolute path of the themes directory to the `COMPREHENSIVE_THEME_DIRS` configuration property.

The specific method that you use to configure Open edX components depends on the type of environment you are using. For example, you can set the configuration property in the following files.

- For Studio, add the path to `COMPREHENSIVE_THEME_DIRS` in `/edx/app/edxapp/cms.env.json`.

```
"COMPREHENSIVE_THEME_DIRS": [
  "/my-open-edx-themes/edx-platform"
],
```

- For the LMS, add the path to `COMPREHENSIVE_THEME_DIRS` in `/edx/app/edxapp/lms.env.json`.

```
"COMPREHENSIVE_THEME_DIRS": [
  "/my-open-edx-themes/edx-platform"
],
```

- For the E-commerce service, add the path to `COMPREHENSIVE_THEME_DIRS` in `/edx/etc/ecommerce.yml`.

```
COMPREHENSIVE_THEME_DIRS: ["/my-open-edx-themes/ecommerce"]
```

5. Restart all servers.

Note: You can create more than one themes directory for your Open edX installation. To use multiple themes directories, include the path to each directory in the `COMPREHENSIVE_THEME_DIRS` configuration property. The following example shows the configuration for multiple themes directories.

```
"COMPREHENSIVE_THEME_DIRS": [
  "/my-open-edx-themes/edx-platform",
  "/my-other-open-edx-themes/edx-platform"
],
```

Apply a Theme to a Site

To apply a theme to an Open edX site, follow these steps.

1. Make sure that you have enabled theming for your Open edX installation and that you have configured an installation-wide themes directory. For more information, see [Enabling and Applying Themes](#).
2. Make sure that you have created a theme and that you know the identifier of the theme. The identifier of a theme is the name of the directory for that theme, within your installation-wide themes directory. For more information, see [Creating a Theme](#).
3. Sign in to the Django administration console for your base URL. For example, `http://{your_URL}/admin`.
4. Select **Site themes**.
5. Select **Add site theme**.
6. From the **Site** menu, select the site you want to apply a theme to.
7. In the **Theme dir name** field, enter the identifier of the theme.
8. Select **Save**.

Compiling a Theme

To update a theme, you compile the Sass files to create the CSS files that style your UI when you apply the theme.

- [Update a Theme for the LMS or Studio](#)
- [Update a Theme for the E-commerce Service](#)

Update a Theme for the LMS or Studio

To update a theme for Studio or the LMS, follow these steps.

1. Log in to the Open edX machine as the `edxapp` user.
2. Change to the `/edx/app/edxapp/edx-platform` directory.
3. Execute the `paver update_assets` command to update all themes.

If you want to update specific themes, use the arguments described in the following table.

Argument	Description
<code>--theme-dir</code>	Provide a space-separated list of the theme directories that you want to update. Only files in the theme directories that you include are updated.
<code>--themes</code>	Provide a space-separated list of the themes that you want to update. Only the themes that you include are updated.

Update a Theme for the E-commerce Service

For the E-commerce service, commands are available for you to update all themes at once, or to update only the themes you specify.

To update a theme for the E-commerce service, follow these steps.

1. Log in to the server for the E-commerce service as the `ecommerce` user.

2. Change to the `/edx/app/ecommerce/ecommerce` directory.
3. To update all themes, execute one of these commands.
 - `make migrate`
 - `python manage.py update_assets`
4. To specify a theme or set of themes to update, or to include optional arguments, execute `python manage.py update_assets` with the options described in the following table.

Argument	Description
<code>--settings</code>	Provide the name of a Django settings module in Python package syntax. For example, <code>--settings=ecommerce.settings.production</code> .
<code>--themes</code>	Provide a space-separated list of the themes that you want to update. Only the themes that you include are updated.
<code>--output-style</code>	Defines the coding style for the compiled CSS files. Possible values are <code>nested</code> , <code>expanded</code> , <code>compact</code> , and <code>compressed</code> . The default value is <code>nested</code> .
<code>--skip-system</code>	Disables Sass file compilation for the default Sass files provided in the Open edX software. Use this option if you have only updated the Sass files in your theme.
<code>--skip-collect</code>	Only compile the Sass files and do not deploy the resulting CSS files.
<code>--enable-source</code>	Include the location of the source file as comments in the resulting CSS files. Enabling this argument can be useful when you are testing a theme.

In addition, an [example theme](#) is available for review.

4.3.2 Styling Drag and Drop Problems

You can customize the appearance of drag and drop problems in your Open edX site using custom Cascading Style Sheet (CSS) files. The style that you configure applies to all drag and drop problems in all courses. For more information about drag and drop problems, see [Drag and Drop Problem](#).

The following two methods apply CSS styling to drag and drop problems.

- You can apply CSS styles to drag and drop problems by creating a theme for your site and updating the Synthetically Awesome Style Sheet (Sass) files that produce the CSS files. For more information, see [Creating a Theme](#).
- You can apply CSS styles to drag and drop problems by adding a Python programming language module that includes a CSS file to your Open edX site. For more information, see the instructions in this section.

Note: This section provides information about styling the drag and drop problem type that was added to the edX platform in 2016. This drag and drop problem type replaced an earlier drag and drop problem type. You should use the latest drag and drop problem type for all new course problems.

Note: Course teams can also style the background and text colors of the draggable items in an individual drag and drop problem, without adding CSS files or configuring a Python module. For more information, see [Changing the Visual Style of a Drag and Drop Problem](#).

To customize the style of drag and drop problems by adding a CSS file in a Python module, follow these steps.

1. Create a custom CSS style sheet that applies styles to the drag and drop problem user interface. You can base your style sheet on the [example CSS file for drag and drop problems](#) that is included in the drag and drop problem module.

2. Create a Python module that includes your custom CSS style sheet. For example, the following Python module files include a CSS style sheet.

```
./my_drag_and_drop_style
./my_drag_and_drop_style/css
./my_drag_and_drop_style/css/my_drag_and_drop_style.css
./my_drag_and_drop_style/__init__.py
./setup.py
```

For more information about creating and installing Python modules, see the documentation for the Python programming language.

3. Install your Python module on the LMS server as the `edxapp` user.

```
pip install /path/to/my/module
```

4. Edit the `lms.env.json` file for your LMS server. Add the `drag-and-drop-v2` object to the `XBLOCK_SETTINGS` object. Include the content shown in the following example.

```
"XBLOCK_SETTINGS": {
  "drag-and-drop-v2": {
    "theme": {
      "package": "my_drag_and_drop_style",
      "locations": ["css/my_drag_and_drop_style.css"]
    }
  }
}
```

Enter the name of your Python module in the value of the `package` object. The value in the example above is `my_drag_and_drop_style`.

Enter the path to your CSS style sheet file in the value of the `locations` object. The value in the example above is `css/my_drag_and_drop_style.css`. The path must be relative to your Python module installation directory. You can include more than one path in the `locations` array, separated by commas.

5. Restart the LMS.

4.4 Adding Custom Fields to the Registration Page

This topic describes how to add custom fields to the registration page for your instance of Open edX.

- [Overview](#)
- [Add Custom Fields to the Registration Page](#)

4.4.1 Overview

By default, the registration page for each instance of Open edX has fields that ask for information such as a user's name, country, and highest level of education completed. You can add custom fields to the registration page for your own Open edX instance. These fields can be different types, including text entry fields and drop-down lists.

4.4.2 Add Custom Fields to the Registration Page

Before you add a custom field to the registration page, you must make sure that the combined sign-in and registration form is enabled for your Open edX instance. To do this, open the `lms.env.json` and `cms.env.json` files, and

set the `ENABLE_COMBINED_LOGIN_REGISTRATION` feature flag to `True`. These files are located one level above the `edx-platform` directory.

To add custom fields to the registration page, follow these steps.

1. Start the LMS and sign in to your instance of Open edX.
2. Use Python to create a Django form that contains the fields that you want to add to the page, and then create a Django model to store the information from the form.

For more information about how to create Django forms, see [Django Forms](#) on the [Django website](#).

3. In the `lms.env.json` file, add the app for your model to the `ADDL_INSTALLED_APPS` array.
4. In the `lms.env.json` file, set the `REGISTRATION_EXTENSION_FORM` setting to the path of the Django form that you just created, as a dot-separated Python string.

For example, if your form is named “`ExampleExtensionForm`” and is located at “`path/to/the_form.py`”, the value of the setting is `path.to.the_form.ExampleExtensionForm`.

5. Run database migrations.
6. Restart the LMS and verify that the new fields appear on your registration form.

Note: For an example app for custom forms, see the OpenCraft [custom_form_app](#) page in [GitHub](#).

4.5 Specifying Allowed Registration Email Patterns

This topic describes how to restrict registration on your site by specifying which email address patterns are allowed in registration emails.

- [Overview](#)
- [Configure Allowed Email Patterns](#)

4.5.1 Overview

By default, all email addresses are accepted when learners register for an account on your Open edX site. You have the option of restricting registrations to learners who use an allowed email address pattern. Doing so can be useful in cases where you want to allow only learners who are members of a school, organization, or corporation to register and access your courses.

Note: Configuring your site using the procedure below only restricts registration to learners whose email addresses match the specified patterns. It does not hide courses from any learners, or prevent access to pages on your site that can be accessed without registration.

4.5.2 Configure Allowed Email Patterns

To specify the email patterns that are allowed for registration, follow these steps.

1. Locate the `lms.env.json` and `cms.env.json` files, which are located one level above the `edx-platform` directory. You make the same changes to both files.

2. In the `lms.env.json` and `cms.env.json` files add the `REGISTRATION_EMAIL_PATTERNS_ALLOWED` setting.

```
"REGISTRATION_EMAIL_PATTERNS_ALLOWED": null
```

If the value for this setting is `null`, there are no restrictions, and all email addresses are accepted for registration.

3. Use one or more Python regular expressions to specify the email domains that allowed email addresses must match.

The following example allows email addresses using the pattern `example.com` or `any.example.com` to register. It also allows `school.tld` addresses, but only if those addresses have a `.` before the `@` symbol.

```
"REGISTRATION_EMAIL_PATTERNS_ALLOWED" = [  
  
    "^.*@(.*\.\.)*example\.\.com$",  
    "^(^\\w+\\.\\.\\w+)?@school\.\.tld$" ]
```

4. Save the `lms.env.json` and `cms.env.json` files.
5. Restart your `edxapp` instances.




4.6 Adding the CourseTalk Widget


This topic describes how to add [CourseTalk](#) widgets to your instance of Open edX. When you add the CourseTalk widget, it is visible for all courses.

- [Overview](#)
- [Add the CourseTalk Widget](#)


4.6.1 Overview

The CourseTalk widget allows learners to write reviews of your course and see reviews that other learners have written. Learners write reviews on the course **Home** page, and the reviews are visible on the course **About** page. Learners can write or read reviews of your course at any time.




 Course Number

DemoX


 Classes Start


Feb 05, 2013

 Price

Free


1 Course Review





XXXXX

-122 seconds ago




XXXXX XXXXX XXXXX XXXXX XXXXX XXXXX
XXXXX XXXXX XXXXX XXXXX XXXXX XXXXX
XXXXX XXXXX XXXXX XXXXX XXXXX XXXXX
XXXXX XXXXX XXXXX XXXXX XXXXX XXXXX
XXXXX XXXXX XXXXX XXXXX XXXXX XXXXX

Was this review helpful?

Yes

0

[Review Guidelines](#) [Terms Of Use](#)



4.6.2 Add the CourseTalk Widget

To add the CourseTalk widget, follow these steps.

1. Sign in to the Django administration console for your base URL. For example, go to `http://{your_URL}/admin`.
2. In the navigation pane, locate **Coursetalk**, and then select **Course talk widget configurations**.
3. On the **Select course talk widget configuration to change** page, select **Add course talk widget configuration**.
4. On the **Add course talk widget configuration** page, select the **Enabled** check box, enter a value in the **Platform key** field, and then select **Save**.

Note: You can use any text that you want as your platform key. EdX recommends that you use your domain name, or part of your domain name.

5. In the LMS, open the **Home** page for any of your Open edX courses, and verify that the CourseTalk widget appears at the bottom of the page.
6. Sign out of your instance of Open edX and view the About page for any course.
7. In the right pane, verify that the CourseTalk widget appears below the course information panel.

4.7 Enabling Open edX Search

You can add a search feature to your Open edX site so that prospective learners can find courses more easily. When a learner searches for a key word or words, the search feature returns a list of the courses that are currently open for enrollment and that match the entered key words.

This section describes how to enable search in your instance of Open edX.

- *Overview*
- *Search Engines and edX Search*
- *edX Search Requirements*
- *Install edX Search*
- *Enable Indexing*
- *Supported Flags*

4.7.1 Overview

EdX Search is a Django application that provides access to search services from within edX Platform applications. Searching is accomplished by creating an index of documents, and then searching within that index for matching information.

When you install the Open edX devstack, edX search is enabled by default. You must enable this feature to use it with Open edX fullstack.

Note: Before proceeding, review *Guidelines for Updating the Open edX Platform*.

4.7.2 Search Engines and edX Search

By default, edX Search uses MockSearchEngine for testing and ElasticSearch Engine for production. You can configure edX Search to use a different search engine.

MockSearchEngine

MockSearchEngine is a simple implementation using a JSON file for index storage. It has no specific requirements, but it does not scale well and should only be used for testing.

ElasticSearchEngine

ElasticSearchEngine is a ElasticSearch back-end implementation. It uses same ElasticSearch version that is already part of Open edX Platform. The current version is v0.90.13, and Django Elasticsearch is 0.4.5.

4.7.3 EdX Search Requirements

EdX Search requires the following applications.

- Django (edX Platform version)
- pyMongo (edX Platform version)
- pytz
- Django elasticsearch (0.4.5)

4.7.4 Install edX Search

EdX Search is included in Open edX Platform GitHub requirements and is installed automatically when you install the Open edX Platform.

For existing installations, you must install edX Search manually.

To install edX Search, make sure you are logged to your server as the `edxapp` user and are located in `edx-platform` directory.

If you are not, run the following before continuing:

```
sudo su edxapp -s /bin/bash cd ~source edxapp_env
```

Then install edX Search using one of the three following options.

Option 1 – Add Requirement

Add the GitHub link to `edx-search` to the `requirements/edx/github.txt` file.

```
-e git+https://github.com/edx/edx-search.git@ae459ead41962c656ce794619f58cdae46eb7896#egg=edx-search
```

Then reinstall GitHub requirements.

```
pip install -r requirements/edx/github.txt
```

Option 2 – Install Locally

Checkout the `edx-search` GitHub repository.

Then in the `edx-search` directory, run the following command.

```
pip install -e ./
```

Option 3 – Install from GitHub

Run `pip` with a GitHub link.

```
pip install -e git+https://github.com/edx/edx-search.git@ae459ead41962c656ce794619f58cdae46eb7896
```

4.7.5 Enable Indexing

You enable course indexing by setting the `ENABLE_COURSEWARE_INDEX` flag.

You enable library indexing by setting the `ENABLE_LIBRARY_INDEX` flag.

Indexing is done from Studio as a Celery task. Every publish event triggers the reindex procedure.

You can also reindex the course manually through the **Reindex** button in the **Course Overview** page.

Note: The search feature only returns results for courses that are currently open for enrollment. Course teams can set the enrollment start and end dates for their courses in Studio. For more information, see [Determining Start and End Dates](#) in the *Building and Running an Open edX Course* guide.

Which Data Gets Indexed

Which data gets indexed is determined by the module `index_dictionary()` function implementation. Modules supporting this method are `Sequence`, `Vertical`, `Video`, and `HTML Block`. You can add support to any module type.

Course metadata, including the name, description, and start and end dates are also indexed.

4.7.6 Supported Flags

The following flags are supported in the CMS and LMS applications.

CMS

- `ENABLE_COURSEWARE_INDEX`: Enables and disables courseware content and course info indexing.
- `ENABLE_LIBRARY_INDEX`: Enables and disables library content indexing.
- `SEARCH_ENGINE`: Sets the search engine to use. There are two predefined values.
 - `"search.elastic.ElasticSearchEngine"`
 - `"search.tests.mock_search_engine.MockSearchEngine"`

- `ELASTIC_FIELD_MAPPINGS`: Sets any additional field mappings that elastic search should be aware of. For example, the following code includes the course start date.

```
ELASTIC_FIELD_MAPPINGS = {  
  "start_date": {  
    "type": "date"  
  }  
}
```

LMS

- `ENABLE_COURSEWARE_SEARCH`: Enables and disables Courseware Search feature (in course searching).
- `ENABLE_DASHBOARD_SEARCH`: Enables and disables Dashboard Search feature (in enrolled courses searching).
- `ENABLE_COURSE_DISCOVERY`: Enables and disables Course Discovery feature (over courses searching and facet filtering).
- `COURSE_DISCOVERY_FILTERS`: If provided, overrides the list of facets that are used in the Course Discovery feature to filter the results. By default, all facets will be displayed. The list of available facets includes:
 - Course organization: `"org"`
 - Course type: `"modes"`
 - Course language: `"language"`
- `SEARCH_ENGINE`: Sets the search engine to use. The following values are predefined.
 - `"search.elastic.ElasticSearchEngine"`
 - `"search.tests.mock_search_engine.MockSearchEngine"`
- `SEARCH_INITIALIZER`: Used to set custom `SearchInitializer`. `SearchInitializer` provides an extension to achieve masquerade and other presearch environmental settings.
 - default: `SearchInitializer`
 - LMS implementation: `lms.lib.courseware_search.lms_search_initializer.LmsSearchInitializer`
- `SEARCH_RESULT_PROCESSOR`: Used to set custom `SearchResultProcessor`. `SearchResultProcessor` does post processing and data manipulation on a result set returned by `SearchEngine`.
 - default: `SearchResultProcessor`
 - LMS implementation: `lms.lib.courseware_search.lms_result_processor.LmsSearchResultProcessor`
- `SEARCH_FILTER_GENERATOR`: Used to set custom `SearchFilterGenerator`. `SearchFilterGenerator` sets filters defined by current active user. Basic implementation sets only course start date filter.
 - default: `SearchFilterGenerator`
 - LMS implementation: `lms.lib.courseware_search.lms_filter_generator.LmsSearchFilterGenerator`

4.8 Enabling Badging

This topic describes how to enable and configure badging in your instance of Open edX.

Note: Before proceeding, make sure you have read [Guidelines for Updating the Open edX Platform](#).

- *Overview*
- *Prerequisites*
- *Enable Badges in Studio and the Learning Management System*
- *Enable Badges Within Each Course*
- *Configure Badges for Your Open edX Instance*
- *Create Course Completion Badges for Your Open edX Instance*
- *Create Course Event Badges for Your Open edX Instance*
- *Creating New Badges for Your Open edX Instance*

4.8.1 Overview

Badges provide a way for learners to share their course achievements. For courses that have course completion badges enabled, learners receive a badge at the same time that they receive a course certificate, and have the option of sharing their badges to a badging site such as Mozilla Backpack. For more information, see *Course Completion Badges*.

In addition to badges that are awarded for completing single courses, you can also issue badges for various cross-course events. For example, you can create badges to be awarded when any of the following events occur.

- A learner enrolls in a certain number of courses.
- A learner receives a completion certificate for a certain number of courses.
- A learner receives a completion certificate for every course in a specified list of courses.

For more information, see *Course Event Badges*.

By default, Open edX supports Open Badges (<http://openbadges.org/>), an open standard originally developed by the Mozilla Foundation. Open Badges provides a badge generator called Badgr Server, which is used by default in Open edX.

You can use a badge generator other than Badgr Server. For information, see *Specify a Badge Generator Other Than Badgr Server*.

For information about creating custom badges, see *Create New Badges for Your Open edX Instance*.

4.8.2 Prerequisites

Setting up the badges feature on your instance of Open edX involves performing the set up and configuration tasks that are described in the topics below.

- *Make Sure Certificates Are Enabled*
- *Specify a Badge Generator*

Make Sure Certificates Are Enabled

Badge generation depends on certificate generation. Badges for course completion are automatically generated when a course certificate is generated for a learner. Make sure certificates are enabled on your Open edX instance. For information, see *Enabling Certificates*.

Specify a Badge Generator

By default, Open edX uses Badgr Server as the badge generator. If you do not use Badgr Server, you can configure a different badge generator.

- To use Badgr Server, see *Install Badgr Server* and *Specify a Badge Issuer for Your Organization*.
- To use a different badge generator, see *Specify a Badge Generator Other Than Badgr Server*.

Install Badgr Server

Badgr Server provides an API for issuing Open Badges. Follow the instructions at <https://github.com/concentricsky/badgr-server> to install and run Badgr Server.

Important: You must install Badgr Server at a publicly accessible IP address, to allow the Open edX LMS and services such as Mozilla Backpack to contact Badgr Server.

If you do not use Badgr Server, you can configure a different badge generator. See *Specify a Badge Generator Other Than Badgr Server*.

Specify a Badge Issuer for Your Organization

Note: This step is required only if you use Badgr Server for your badge generator.

If you are using Badgr Server, log in to your installation of Badgr Server and add an issuer of Open Badges for your organization.

For more information about issuing Open Badges, see the *Issuing Badges* topic on the Mozilla wiki.

Specify a Badge Generator Other Than Badgr Server

By default, Open edX uses Badgr Server as the badge generator. To use Badgr Server, see *Install Badgr Server* and *Specify a Badge Issuer for Your Organization*.

To specify a different badge generator, follow these steps.

Note: Because the services and software used for badge generation can differ significantly, the steps described in this topic are intended as guidelines and not as exact procedures.

1. Add the `BadgingBackend` object as a subclass to `lms/djangoapps/badges/backends/base.py`.

A `BadgingBackend` object must include the `award` function

```
award(badge_class, user_id, evidence_url=None)
```

where

- `badge_class` is the definition of the badge class for which the `BadgeAssertion` is created,
- `user_id` is an ID for the user that the `BadgeAssertion` is to be created for, and
- `evidence_url` is an optional URL for the location where the badge evidence can be viewed.

The `award` function is responsible for the server awarding a badge to a learner. It is also responsible for all checks leading up to the awarding of the badge, including making sure the badge specification does not already exist on the target before creating a new badge specification.

The `award` function should either return a `BadgeAssertion` object or raise an exception if the award cannot be given. By default, a base badge generation object called `BadgeBackend` exists, and contains a dummy version of the `award` function.

```
class BadgeBackend(object):
    """
    Defines the interface for badge generators.
    """
    __metaclass__ = ABCMeta

    @abstractmethod
    def award(self, badge_class, user, evidence_url=None):
        """
        Create a badge assertion for the user using this badge generator.
        """
```

2. In the `BadgeAssertion` object that is generated by the `award` function, make sure the `backend` value is the name of the `BadgingBackend` subclass.

```
class BadgeAssertion(models.Model):
    """
    Tracks badges on our side of the badge baking transaction
    """
    user = models.ForeignKey(User)
    badge_class = models.ForeignKey(BadgeClass)
    data = JSONField()
    backend = models.CharField(max_length=50)
    image_url = models.URLField()
    assertion_url = models.URLField()
```

3. In the `lms.env.json` and `cms.env.json` files, set the value of `BADGING_BACKEND` as a dot-separated python path specification to the object that you use to create and assign badges.
4. When you have finished modifying your configuration files for the badge generator, restart the Studio and Learning Management System processes so that the updated environment configurations are loaded.

4.8.3 Enable Badges in Studio and the Learning Management System

To enable badges, you modify the `lms.env.json` and `cms.env.json` files, which are located one level above the `edx-platform` directory.

1. In the `lms.env.json` and `cms.env.json` files, set the value of `ENABLE_OPENBADGES` to `True`.

```
# Enable OpenBadge support.
'ENABLE_OPENBADGES': True,
```

2. In `lms.env.json`, set the values for the following parameters.
 - `BADGR_API_TOKEN`: A string containing the API token for the Badgr superuser account. Obtain the token from the `/v1/user/auth-token` page while logged in to the API as the superuser.
 - `BADGR_BASE_URL`: A string containing the base URL for Badgr Server. The Badgr Server must be installed at a publicly accessible IP address.

- `BADGR_ISSUER_SLUG`: A string that is the slug for the Badgr issuer. The slug can be obtained from the URL of the Badgr Server page that displays the issuer. For example, in the URL `http://exampleserver.com/issuer/test-issuer`, the issuer slug is `test-issuer`.

```
##### Badgr OpenBadges generation #####  
  
BADGR_API_TOKEN = None  
# Do not add the trailing slash here.  
BADGR_BASE_URL = "http://localhost:8005"  
BADGR_ISSUER_SLUG = "test-issuer"
```

3. Save the `lms.env.json` and `cms.env.json` files.
4. Run database migrations.
5. Restart the Studio and Learning Management System processes so that the updated environment configurations are loaded.

4.8.4 Enable Badges Within Each Course

The ability to issue course completion badges is enabled by default for all courses, but can be turned off or on again using an advanced setting in Studio. For information, see [Enable or Disable Badges for Your Course](#) in *Building and Running an Open edX Course*.

Note: The course-level setting to enable badges does not affect badges that are issued for completing or enrolling in multiple courses.

4.8.5 Configure Badges for Your Open edX Instance

You can configure several types of badges to issue to learners in your Open edX instance.

- *Course Completion Badges*
- *Course Event Badges*

In addition, you can use the badging framework to create custom badges. For information, see [Creating New Badges for Your Open edX Instance](#).

Course Completion Badges

For courses that have badges enabled, learners receive a course completion badge at the same time as they receive a course certificate, and have the option of sharing their badges to a badging site such as Mozilla Backpack.

You can configure a different course completion badge for each course mode that you support (for example, “professional”, “advanced”, or “basic”). You can also specify a different badge image for each of the badges that you configure. For information, see [Enable Badges Within Each Course](#) and [Create Course Completion Badges for Your Open edX Instance](#).

Course Event Badges

Course event badges are awarded across courses, and can be awarded when any of the following events occur.

- A learner enrolls in a certain number of courses.
- A learner receives a completion certificate for a certain number of courses.
- A learner receives a completion certificate for every course in a specified list of courses.

You can customize these course event badges with your parameters and badge images. For information, see [Create Course Event Badges for Your Open edX Instance](#).

4.8.6 Create Course Completion Badges for Your Open edX Instance

Important: Default images are supplied in Open edX for course completion badges. Be sure to replace these default badge images with your organization's own badge images before any badges are issued. When the first badge is issued for a given course, badge images are uploaded to Badgr Server. All badges issued in future for this course will use this uploaded original badge image, even if you subsequently change badge images in the Django Administration badge image configuration.

1. Access the Django Administration website for your instance of Open edX. To do this, go to `https://<host name of your Open edX instance>/admin`. For example, this might be `https://YourOrganization.org/admin`.
2. Select **Site Administration > Badges > Course complete image configurations**, and then define a course complete image configuration for each course mode on your platform for which you want to issue badges upon course completion. Examples of course modes are “professional” or “advanced” or “basic”.
3. For each course complete badge image configuration, set these parameters.
 - Mode: The course mode for which the badge image should be used.
 - Icon: The badge image to use for the specified course mode.

Important: Be sure to replace the default badge images with your organization's own badge images before any badges are issued.

4. Optionally, you can define a badge image that will be used as the default badge image for any course modes that do not have an explicitly specified badge image.

To do so, in the course complete image configuration that references the image you want to use as a default, select the **Default** checkbox. After you save the configuration, this badge image is used for any course completion badge configurations that do not have a badge image explicitly specified.

Note: You can specify only one default badge image.

5. Save each configuration parameter and exit the Django Administration website.

4.8.7 Create Course Event Badges for Your Open edX Instance

Open edX provides several customizable course event badges that can be awarded when any of the following events occur.

- A learner enrolls in a certain number of courses.
- A learner receives a completion certificate for a certain number of courses.

- A learner receives a completion certificate for every course in a specified list of courses.

Before course event badges can be awarded, you must customize them with your parameters and badge images. To customize any of the course event badges, follow these steps.

Note: You can also use the badging framework to create custom badges. For information, see [Creating New Badges for Your Open edX Instance](#).

1. Access the Django Administration website for your instance of Open edX. To do this, go to `https://<host name of your Open edX instance>/admin`. For example, this might be `https://YourOrganization.org/admin`
2. Select **Site Administration > Badges > Badge Classes**.
3. Add a badge class for each course event for which you want to issue badges. Examples of course events might be enrolling in five courses, or completing three required courses.
4. For each badge class, set the following parameters.

- **Slug:** A unique identifier that you choose to identify the badge class. This identifier can contain only numbers, lowercase letters, underscores, or hyphens. The slug, combined with the Issuing Component value, uniquely identifies a badge.
- **Issuing Component:** Identifies the part of the platform that is issuing the badge. This identifier can contain only numbers, lowercase letters, underscores, or hyphens. For the three customizable course event badges that are included in the Open edX platform, the value for **Issuing Component** must be `openedx__course` (with two underscores). For [course completion badges](#) that are included in the Open edX platform, the issuing component value should be empty.

For new badge types that you create, specify an **Issuing Component** value that identifies the software component responsible for issuing the badge. For example, if badges are issued by the course management component, you might define **Issuing Component** as `platform__course`; if badges are issued based on activity in course discussions, you might define **Issuing Component** as `platform__discussions`.

- **Display name:** The human readable badge name that is used when badges are shown to learners, for example, in the Accomplishments view of learners' profiles.
- **Course ID:** This value should be blank for course event type badges, as they are not associated with a single course.
- **Description:** A description of this badge.
- **Criteria:** A description of the criteria for awarding this badge.
- **Mode:** The course mode for the course associated with this badge, if applicable.
- **Image:** The badge image to use for this badge. Badge images should be square .png files less than 250KB in size.

An example of a badge class configuration might have the following values.

- `slug: enrolled_three`
- `issuing_component: openedx__course`
- `display_name: Enroll in Three Courses`
- `description: Enrolled in three courses`
- `criteria: A learner must enroll in three courses to receive this badge`
- `image: triple_enrollment_badge_image.png`

5. When you have finished defining the badge class, select **Save**.
6. Next, you create a new course event badge configuration that defines all of the course event badges that you want to issue. Select **Site Administration > Badges > Course event badges configurations > Add course event badges configuration**.

Important: You can create more than one course event badge configuration, but you can only mark one configuration as **Enabled**. Only the most recently activated course event badge configuration is used.

7. Within the new course event badge configuration, set the following parameters.
 - **Courses completed:** Define badges to be awarded for completing a certain number of courses, or completion of specific courses. Define one badge per line. On each line, enter the number of courses that must be completed, followed by a comma and then the slug of the badge class to associate with this badge.

For example, to configure two badges, one that is awarded when a learner completes 3 courses, and another that is awarded when a learner completes 8 courses, you add two lines to the **Courses completed** field.

`3,completed_three 8,completed_eight`

where `completed_three` and `completed_eight` are badge slugs that you previously defined in badge classes.
 - **Courses enrolled:** Define badges to be awarded for enrolling in a certain number of courses, or enrolling in specific courses. Define one badge per line. On each line, enter the number of courses that must be enrolled in, followed by a comma and then the slug of the badge class to associate with this badge.

For example, to configure a badge that is awarded when a learner enrolls in 5 courses, you add this definition.

`5,enrolled_five`

where `enrolled_five` is a badge slug that you previously defined in a badge class.
 - **Course groups:** Define badges to be awarded for completing a list of specific courses. Define one badge per line. On each line, enter the slug of the badge class, a comma, then the list of course keys.

For example, to configure a badge that is awarded when a learner completes the 3 prerequisite courses in a series, you add this definition.

`prereq_computerscience_badge_slug,course1_identifier,course2_identifier,course3_identifier`

where `prereq_computerscience_badge_slug` is a badge slug that you previously defined in a badge class, and `course1_identifier`, `course2_identifier`, and `course3_identifier` are the Course IDs for the three courses that must be completed for this badge.
8. When you have finished defining badges in the configuration, select **Save**.
9. To activate this configuration, select **Enabled** at the top of the configuration page.

Important: You can create more than one course event badge configuration, but you can only mark one configuration as **Enabled**. Only the most recently activated course event badge configuration is used.

4.8.8 Creating New Badges for Your Open edX Instance

In addition to using the default customizable badges that are provided with the Open edX platform, you can design new badges that are generated when a particular XBlock-related or course-related action occurs.

Before you create new badges, you should understand the following concepts.

- **Badge Class** - The specification of the badge that is to be awarded. Parameters for badge classes are described in Step 4 of the [Create Course Event Badges for Your Open edX Instance](#) topic.
- **Slug** - This field in the `BadgeClass` model uniquely identifies the badge class.
- **Issuing Component** - This field in the `BadgeClass` model identifies the part of the software that is issuing the badge. For example, the name of the `XBlock` where the occurrence of some event triggers the awarding of a badge.

For the customizable [course event badges](#) that are included with the Open edX platform, the value for `issuing_component` must be `openedx__course` (with two underscores). For [course completion badges](#) that are included in the Open edX platform, the issuing component value should be empty.

For new badge types that you create, specify an **Issuing Component** value that identifies the software component responsible for issuing the badge. For example, if badges are issued by the course management component, you might define **Issuing Component** as `platform__course`; if badges are issued based on activity in course discussions, you might define **Issuing Component** as `platform__discussions`.

- The combination of badge slug, issuing component, and optionally, course ID uniquely identifies an awarded badge.
- **Award method** - The `award` function on the `Badge Class` instantiates the badge generator and calls the badge generator's `award` function, which is responsible for awarding a badge to a learner, as well as for performing all checks leading up to the awarding of the badge. On the `Badge Class`, the `award` function either returns a `BadgeAssertion` object or raises an exception if the award cannot be given.
- **Badge Assertion** - A specific generated instance of a badge that a learner has been awarded. Badge assertions contain data about the learner who earned the badge, and the date and time that the badge was awarded. Multiple badge assertions can be awarded for a specific `BadgeClass`, including to the same learner. You can get all assertions for a particular learner for a specific `BadgeClass` using the `get_for_user` method, using `user` as an argument.

For instructions for creating new badges, see [Create New Badges for Your Open edX Instance](#).

Create New Badges for Your Open edX Instance

To create new badges for use within your Open edX platform, follow these steps.

1. Create a `Badge Class` specifying the details of the badge you want to award. You can do this using the `get_badge_class` method from the `badges` app, or create a badge class in Django Admin. For Django Admin instructions, see steps 1-5 in [Create Course Event Badges for Your Open edX Instance](#).

`BadgeClass.get_badge_class` creates the requested badge class if one does not already exist that has the same combination of `slug` and `issuing_component`. Badge classes are uniquely identified by a combination of their `slug` and `issuing_component` fields, and optionally also the `course_id` field if a badge is associated with an individual course.

If a badge class already exists with the same combination of `slug` and `issuing_component` that is in the request, the existing badge is returned. No new badge class is created, and no changes are made to the values of the existing badge class.

2. In the `XBlock` or component where badges are to be awarded based on some event occurring, add declarations to the `badging` and `user` services.

The following example illustrates creating a badge class and awarding it from an `XBlock`.

```
from xblock import XBlock
from xblock.fragment import Fragment
import pkg_resources
```

```
@XBlock.wants('badging')
@XBlock.wants('user')
class AwardBlock(XBlock):
    """
    A Block that awards a badge when a learner visits it.
    """
    def award_badge(self, user_service, badge_service):
        user = user_service.get_current_user()
        badge_class = badge_service.get_badge_class(
            slug='general_award', issuing_component='my_org__award_block',
            description="A shiny badge, given to anyone who finds it!",
            criteria="Visit a page with an award block.",
            # This attribute not available in all runtimes,
            # but if we have both of these services, it's a safe bet we're in the LMS.
            course_id=self.runtime.course_id,
            # The path to this file should be somewhere relative to your XBlock's package.
            image_file_handle=pkg_resources.get_resource_stream(__name__, 'badge_images/award.png')
            # Badge image should be a square PNG file less than 250KB in size.
        )
        # Award the badge.
        if not badge_class.get_for_user(user):
            badge_class.award(user)

    def student_view(self, context=None):
        """
        Displayed to the learner when they visit.
        """
        # If the user and badge services are not present, we cannot award the badge.
        # If they are, we are ready to award one.
        user_service = self.runtime.service(self, 'user')
        badge_service = self.runtime.service(self, 'badging')
        if user_service and badge_service:
            self.award_badge(user_service, badge_service)
        return Fragment(u"<div><p>You just earned a badge!</p></div>")
```

Get Information about Badge Assertions

Depending on the type of badge you have created and are awarding, you might want to limit the number of times that a learner can receive a badge. You can find whether a specific learner has already received a particular badge in either of the following ways.

- Use `badge_class.assertions_for_user` with `user` as an argument, to return a list of all assertions that the specified user has received for the badge class.
- Use the Badges API GET method to return a list of assertions for a particular username. For example, use `GET /api/badges/v1/assertions/user/{username}/`.

For more information and additional parameters, see [Supported Badges API Endpoint](#).

Supported Badges API Endpoint

The Badges API supports the endpoint `GET /api/badges/v1/assertions/user/{username}/`. You can use the following query parameters with this endpoint.

Note: All of these query parameters are optional.

Parameters	Description
slug	If used, filters by the badge class identified by this slug. Unless <code>issuing_component</code> is also specified, assumes a null/empty issuing component.
issuing_component	If used, also requires <code>slug</code> to be specified. Filters by the badge class.
courseid	If used, returns only badge assertions that were awarded as part of the specific course.

For example, to get a list of badge assertions issued for a badge with an `issuing_component` value of `openedx__course` and a `slug` value of `enroll_in_three_courses`, the query would be

```
<openedx_instance>/api/badges/v1/assertions/user/?issuing_component=openedx__course&slug=enroll_in_three_courses
```

where `<openedx_instance>` is the site URL for your Open edX instance.

Possible query results are as follows.

- 200 on success, with a list of badge assertions.
- 403 if a user who does not have permission to masquerade as another user specifies a username other than their own.
- 404 if the specified user, badge class, or course does not exist.

4.9 Enabling Certificates

This topic describes how to enable certificates in your instance of Open edX.

- [Overview](#)
- [Enable Certificates in Studio and the Learning Management System](#)
- [Configure Certificates for Your Open edX Instance](#)
- [Customize Certificate Templates For Your Organization](#)
- [Configure Certificates Within Each Course](#)
- [Generate Certificates For a Course](#)

4.9.1 Overview

Organizations and course teams can generate certificates for learners who pass a course. Learners can view, print, or share their certificates.

For information about certificates, see [Setting Up Course Certificates](#) in *Building and Running an Open edX Course* or [Print a Web Certificate](#) in the *Open edX Learner's Guide*.

To enable this feature on your instance of Open edX, you must enable a feature flag in both Studio and the Learning Management System and complete the configuration tasks described in this topic.

Note: Before proceeding, review [Guidelines for Updating the Open edX Platform](#).

4.9.2 Enable Certificates in Studio and the Learning Management System

To enable certificates, you modify the `lms.env.json` and `cms.env.json` files, which are located one level above the `edx-platform` directory.

1. In the `lms.env.json` and `cms.env.json` files, set the value of `CERTIFICATES_HTML_VIEW` within the `FEATURES` object to `true`.

```
"FEATURES": {  
  ...  
  'CERTIFICATES_HTML_VIEW': true,  
  ...  
}
```

2. Save the `lms.env.json` and `cms.env.json` files.
3. If it does not exist already, create the folder `/tmp/certificates` owned by the user and group `www-data`. Depending on your configuration, this folder might not survive reboots, and so might need to be created by a script.
4. Run database migrations.

4.9.3 Configure Certificates for Your Open edX Instance

1. Access the Django Administration website for your instance of Open edX. To do this, go to `https://<host name of your Open edX instance>/admin`. For example, this might be `https://YourOrganization.com/admin`.
2. Under **Site Administration > Certificates**, add an HTML View Configuration, and select **Enabled**.
3. Modify the configuration parameters. You must set the following certificates-related parameters for your Open edX instance.

- `platform_name`
- `company_about_url`
- `company_privacy_url`
- `company_tos_url`
- `company_verified_certificate_url`
- `logo_src`
- `logo_url`

For each course mode, such as “honor” or “verified”, define `certificate_type`, `certificate_title`, and `document_body_class_append`. The mode name should match your course mode name exactly. An example follows.

```
{  
  "default": {  
    "accomplishment_class_append": "accomplishment-certificate",  
    "platform_name": "YourPlatformName",  
    "company_about_url": "http://www.YourOrganization.com/about-us",  
    "company_privacy_url": "http://www.YourOrganization.com/our-privacy-policy",  
    "company_tos_url": "http://www.YourOrganization.com/our-terms-service",  
    "company_verified_certificate_url": "http://www.YourOrganization.com/about_verified_certificate",  
    "logo_src": "/static/certificates/images/our_logo.svg",  
    "logo_url": "www.YourOrganization.com"  
  },  
  "honor": {  
    "certificate_type": "honor",  
    "certificate_title": "Honor Certificate",  
    "document_body_class_append": "is-honorcode"  
  }  
}
```

```

    },
    "verified": {
        "certificate_type": "verified",
        "certificate_title": "Verified Certificate",
        "document_body_class_append": "is-idverified"
    },
    "base": {
        "certificate_type": "base",
        "certificate_title": "Certificate of Achievement",
        "document_body_class_append": "is-base"
    },
    "distinguished": {
        "certificate_type": "distinguished",
        "certificate_title": "Distinguished Certificate of Achievement",
        "document_body_class_append": "is-distinguished"
    }
}

```

4. Save the configuration parameters and exit the Django Administration website.
5. Restart the Studio and Learning Management System processes so that the updated environment configurations are loaded.

Discontinue Audit Track Certificates

Organizations that offer certificates to audit track learners who pass a course can discontinue generation of this type of certificate. For example, your organization makes a strategic decision to offer certificates only to learners who select an enrollment mode other than “audit”. Learners can continue to audit courses, but they no longer receive certificates.

An outline of the steps you might take if your organization decides to stop offering certificates for learners in the audit track follows.

1. Stop advertising audit track certificates for new courses.
2. Identify running courses that offer an audit track certificate and, for those courses, determine the course end date that is furthest in the future.
3. Select a cutoff date for generating audit track certificates that is after the last course end date identified in step 2.
4. Set `AUDIT_CERT_CUTOFF_DATE` to a date in YYYY-MM-DD format. Specifying this date ensures that certificates are not generated for audit track learners in any course after the specified date.

The `AUDIT_CERT_CUTOFF_DATE` feature flag affects only the generation of audit certificates. Learners who audit courses continue to receive grades, which are shown on the course **Progress** page.

4.9.4 Customize Certificate Templates For Your Organization

Set up the templates for certificates that your organization will issue. Base templates are included, but you must ensure that they are customized for your organization. For example, you can change the images that appear on certificates for each course mode that your organization supports, as well as fonts and colors that are used on certificates.

Assets for HTML certificates exist in the following locations.

- `lms/templates/certificates` - this folder contains .html files for certificates. The file `valid.html` is an example of a certificate file. Files with names that start with an underscore, such as `_certificate_footer.html`, are partial files that can be referenced in the main certificate .html files.

- `lms/static/certificates` - subfolders of this folder contain assets used in creating certificates, such as images, fonts, and sass/css files.

Note: The organization logo on a certificate is uploaded in Studio. For details, see [Setting Up Course Certificates](#) in *Building and Running an Open edX Course*.

4.9.5 Configure Certificates Within Each Course

Within Studio, course team members with the Admin role can create and edit a certificate configuration that is used to generate certificates for their course, including adding signatories and images for organization logo and signature images for signatories. For details, [Setting Up Course Certificates](#) in *Building and Running an Open edX Course*.

4.9.6 Generate Certificates For a Course

To generate certificates for a course, run the `manage.py` script with the following settings. When the script finishes running, grades are calculated for learners who are enrolled in the course, and certificates are generated for eligible learners.

1. Obtain the course ID for the course for which you are generating certificates. When you view course content in your browser, the course ID appears as part of the URL. For example, in the URL `http://www.edx.org/course/course-v1:edX+demoX_Demo_2015`, the course ID is `course-v1:edX+demoX_Demo_2015`. For some courses, the course ID contains slashes. For example, `edX/DemoX/Demo_2014`.
2. Run `manage.py` with the following settings, replacing `{CourseID}` with the actual course ID. Do not include beginning or trailing slashes.

```
./manage.py lms --settings=aws ungenerated_certs -c {CourseID}
```

For example,

```
./manage.py lms --settings=aws ungenerated_certs -c course-v1:edX+demoX_Demo_2015.
```

Note: If the LMS is running on a server that does not have https support (such as a locally run fullstack for testing) you will need to use the `--insecure` flag so that the certificate generation service contacts the LMS on http instead of on https.

3. View the certificate generation status for a course using `gen_cert_report`. An example follows.

```
./manage.py lms --settings=aws gen_cert_report -c course-v1:edX+demoX_Demo_2015.
```

4.10 Enabling Custom Courses

To enable designated users to create custom courses (CCX) on your instance of Open edX, you must configure the `server-vars.yml` file in the edX platform.

Note: Before proceeding, review [Guidelines for Updating the Open edX Platform](#).

1. Stop the LMS server.

2. Create or update the file `/edx/app/edx_ansible/server-vars.yml` to include the CCX feature flag.

```
EDXAPP_FEATURES:  
  CUSTOM_COURSES_EDX: true
```

3. Run the command `/edx/bin/update`.

```
sudo /edx/bin/update edx-platform <your-branch-name>
```

4. Restart the LMS server.

4.11 Enabling Entrance Exams

This topic describes how to enable entrance exams in your instance of Open edX.

- [Overview](#)
- [Configure the Milestones Application](#)
- [Enable Entrance Exams in Studio and the Learning Management System](#)

4.11.1 Overview

Course teams can create an entrance exam for the course. Learners must pass the entrance exam before participating in the course.

To enable this feature on your instance of Open edX, you must enable entrance exams in Studio and the Learning Management System.

For information about entrance exams, see the *Building and Running an Open edX Course* and *Open edX Learner's* guides.

Note: Before proceeding, review [Guidelines for Updating the Open edX Platform](#).

4.11.2 Configure the Milestones Application

1. Set the value of `MILESTONES_APP` in the `lms.env.json` and `cms.env.json` files to `True`.

```
# Milestones application flag  
'MILESTONES_APP': True,
```

2. Save the `lms.env.json` and `cms.env.json` files.
3. Run database migrations.

4.11.3 Enable Entrance Exams in Studio and the Learning Management System

To enable entrance exams, you modify the `lms.env.json` and `cms.env.json` files, which are located one level above the `edx-platform` directory.

1. Set the value of `ENTRANCE_EXAMS` in the `lms.env.json` and `cms.env.json` files to `True`.

```
# Entrance exams feature flag
'ENTRANCE_EXAMS': True,
```

2. Save the `lms.env.json` and `cms.env.json` files.

4.12 Configuring Open Response Assessments

You can change the default configuration for the Open Response Assessment (ORA2) application. You can change the default file storage system or change the default set of files that learners are prohibited from submitting.

4.12.1 Configuring ORA2 to Upload Files to Alternative Storage Systems

By default, the Open Response Assessment (ORA2) application stores files that learners upload in an Amazon S3 bucket.

You can configure ORA2 to store files in an alternate system. To have learners' files stored in a system other than Amazon S3, follow these steps.

1. In the ORA-2 repository, implement the `BaseBackend` class defined in the `base.py` file.
For example, the `S3.py` file in the same directory is an implementation of `BaseBackend` for Amazon S3. You must implement the equivalent class for the storage system you intend to use.
2. Configure ORA2 to use your alternative storage system by modifying the value of `backend_setting` in `init file` to point to your implementation of `BaseBackend`.
3. Add code to instantiate the new implementation to the `get_backend()` function in the `init.py` file.
4. Configure ORA2 to use the alternative storage system by modifying the value of `ORA2_FILEUPLOAD_BACKEND` in the Django settings to point to your implementation of `BaseBackend`.

4.12.2 Prohibiting Submission of Specified File Types

Course teams can configure open response assessments so that learners can upload files along with their text responses. During the peer review stage of the assessment, other learners download the submitted file and read the response.

To protect learners from exposure to files with malicious content, the ORA2 application uses a “blacklist” to identify a set of file types that learners are not permitted to upload.

To add or remove file types from the blacklist, follow these steps.

1. In the ORA-2 repository, use an editor to open the `submission_mixin.py` file.
2. Locate the `FILE_EXT_BLACK_LIST` parameter in the file. By default, this parameter lists the following file types.

```
FILE_EXT_BLACK_LIST = [
    'exe', 'msi', 'app', 'dmg', 'com', 'pif', 'application', 'gadget',
    'msp', 'scr', 'hta', 'cpl', 'msc', 'jar', 'bat', 'cmd', 'vb', 'vbs',
    'jse', 'ws', 'wsf', 'wsc', 'wsh', 'scf', 'lnk', 'inf', 'reg', 'ps1',
    'ps1xml', 'ps2', 'ps2xml', 'psc1', 'psc2', 'msh', 'msh1', 'msh2', 'mshxml',
    'msh1xml', 'msh2xml', 'action', 'apk', 'app', 'bin', 'command', 'csh',
    'ins', 'inx', 'ipa', 'isu', 'job', 'mst', 'osx', 'out', 'paf', 'prg',
    'rgs', 'run', 'sct', 'shb', 'shs', 'u3p', 'vbscript', 'vbe', 'workflow',
    'htm', 'html',
]
```

3. Add or remove values from this list.
4. Save your changes to `submission_mixin.py`.
5. Restart the Studio (CMS) and Learning Management System (LMS) processes so that your updates are loaded.

For more information and examples of how course teams might set up an open response assessment, see [Open Response Assessments](#) in the *Building and Running an Open edX Course* guide.

4.13 Enabling Course Prerequisites

This topic describes how to enable course prerequisites in your instance of Open edX.

- [Overview](#)
- [Configure the Milestones Application](#)
- [Enable Prerequisite Courses in Studio and the Learning Management System](#)

4.13.1 Overview

Course teams can set prerequisites for a course. Learners must complete the prerequisite courses before participating in the course.

To use this feature on your instance of Open edX, you must configure the Milestones application, then enable prerequisites in Studio and the Learning Management System.

For information about prerequisites, see the *Building and Running an Open edX Course* and *Open edX Learner's* guides.

Note: Before proceeding, review [Guidelines for Updating the Open edX Platform](#).

4.13.2 Configure the Milestones Application

1. Set the value of `MILESTONES_APP` in the `lms.env.json` and `cms.env.json` files to `True`.

```
# Milestones application flag
'MILESTONES_APP': True,
```

2. Save the `lms.env.json` and `cms.env.json` files.
3. Run database migrations.

4.13.3 Enable Prerequisite Courses in Studio and the Learning Management System

To enable prerequisite courses, you modify the `lms.env.json` and `cms.env.json` files, which are located one level above the `edx-platform` directory.

1. Set the value of `ENABLE_PREREQUISITE_COURSES` in the `lms.env.json` and `cms.env.json` files to `true`.

```
# Prerequisite courses feature flag
'ENABLE_PREREQUISITE_COURSES': true,
```

2. Save the `lms.env.json` and `cms.env.json` files.

4.14 Enabling Course and Video Licensing

This topic describes how to enable licensing in your instance of Open edX.

- [Overview](#)
- [Enable Licensing in Studio](#)

4.14.1 Overview

Course teams can specify licensing options for course content as well as for each video in a course.

Course teams can select one of the following license options.

- All Rights Reserved
- Creative Commons

By specifying the license, course teams communicate to learners whether and how they can reuse course content.

To enable this feature on your instance of Open edX, you must enable licensing in both Studio and the Learning Management System.

Note: Before proceeding, review [Guidelines for Updating the Open edX Platform](#).

4.14.2 Enable Licensing in Studio

To enable licensing, you modify the `lms.env.json` and `cms.env.json` files, which are located one level above the `edx-platform` directory.

1. In the `lms.env.json` and `cms.env.json` files, in the `FEATURES` dictionary, add `'LICENSING': True`:

```
FEATURES = {
    'LICENSING': True,
    . . .
```

2. Save the `lms.env.json` and `cms.env.json` files.

4.15 Configuring an edX Instance as an LTI Tool Provider

You can configure your edX instance to be a learning tool interoperability (LTI) provider to other systems and applications. You can use this LTI capability to present content from an edX course in any application that is configured to be a consumer of that content. After you enable your edX instance as an LTI tool provider and configure credentials for the tool consumers, course teams can reuse course content from the edX instance in contexts other than the edX LMS.

4.15.1 Enable LTI Provider Functionality

LTI provider functionality is provided in the `lti_provider` app, located in `edx-platform/lms/djangoapps/lti_provider`.

By default, the `lti_provider` app is not used by edX installations. To enable this functionality throughout the platform, follow these steps.

1. In the `edx/app/edxapp/lms.env.json` file, edit the file so that it includes the following line in the features section.

```
"FEATURES" : {
  ...
  "ENABLE_LTI_PROVIDER": true
}
```

2. Save the `edx/app/edxapp/lms.env.json` file.
3. Run database migrations.
4. Restart the LMS server.

To verify that the LTI provider functionality is enabled, you can check for the presence of the following database tables.

```
lti_provider_gradedassignment
lti_provider_lticonsumer
lti_provider_ltiuser
lti_provider_outcomeservice
```

If these tables are not present, check that the migrations have run properly.

4.15.2 Configuring Credentials for a Tool Consumer

For each external learning management system or application (external LMS) that you want to allow access to your edX instances as an LTI tool consumer, you create OAuth1 credentials, and then configure your edX instance to allow access. Each external LMS that you *configure as a tool consumer* must have separate credentials.

After you complete the configuration of a tool consumer on your edX system, you can add the consumer credentials to your external LMS. For examples of how course teams might set up a course on an external LMS as a consumer of edX course content, see [Using Open edX as an LTI Tool Provider](#) in the *Building and Running an edX Course* guide.

Configure the Tool Consumer

To configure an LTI tool consumer to have access to your Open edX installation, follow these steps.

1. Sign in to the Django administration console for your base URL. For example, `http://{your_URL}/admin`.
2. In the **LTI Provider** section, next to **LTI Consumers** select **Add**.
3. Enter the following information.
 - **Consumer Name:** An identifying name for the tool consumer.
 - **Consumer Key:** The console generates a unique key value for this tool consumer. Alternatively, you can use an external application to generate the key, and then enter it here.
 - **Consumer Secret:** The console generates a unique secret value for this tool consumer. Alternatively, you can use an external application to generate the secret, and then enter it here.

Important: Do not supply a value for the **Instance guid** field. The tool consumer generates and supplies a globally unique identifier.

4. Select **Save** at the bottom of the page.

4.15.3 Define an Interval for Grade Aggregation (Optional)

When an external LMS links to problem components in a graded edX subsection, the edX system grades the answers to those problems, and then transfers the grades back to the external LMS.

- If the link is to an individual problem component on the edX system, the edX system returns the grade for each learner immediately.
- If the link is to a unit or subsection, you can configure an interval of time for the edX system to delay before returning the grades. The edX system aggregates all of the problems in the unit or subsection that the learner answers during that interval. Aggregating grades can reduce the number of notification messages that learners receive.

By default, the edX system aggregates grades for units and subsections every 15 minutes.

To change the interval for returning aggregated grades, follow these steps.

1. In `edx/app/edxapp/lms.env.json`, change the value for the following parameter.

```
LTI_AGGREGATE_SCORE_PASSBACK_DELAY = 15 * 60
```

You specify a time value in seconds.

2. Save the `/lms/envs/common.py` file.
3. Restart the Learning Management System processes so that the updated environment configurations are loaded.

4.15.4 Options for LTI Authentication and User Provisioning

When you use your Open edX system as an LTI tool provider, data is collected by the Open edX system for all learner activity. Each learner has a user account on the Open edX system that is linked to the user account on the tool consumer system, so that activity, grades, and state can be passed from one system to the other.

The Open edX system supports these user authentication flows for LTI.

- *Anonymous User Authentication*
- *Open edX User Authentication*

Anonymous User Authentication

The first time a learner encounters an Open edX resource in a course, the Open edX content is immediately launched by a POST to the URL. Without requiring any action from the learner, the Open edX system creates a user account and provisions it with a system-generated username, and links it to the tool consumer user account for that learner. Learners never interact with the Open edX system directly.

This authentication flow presents a virtually seamless experience that significantly reduces user error. The Open edX system passes learner data to the tool consumer with no subsequent reconciliation of data between the systems.

After you *configure your edX instance as an LTI tool provider*, no further configuration is needed on your Open edX system for this user authentication flow.

Open edX User Authentication

The first time a learner encounters an Open edX resource in a course, he is prompted to either sign in with existing credentials or create a user account. The Open edX system creates a user account and provisions it with the supplied values, and links it to the tool consumer user account for that learner. The POST to the URL then delivers the Open edX resource in the tool consumer. After the initial sign in or account creation step, learners do not interact with the Open edX system directly.

In this authentication flow, learners knowingly establish or use credentials on the Open edX system. This flow provides a smooth learner experience that can also satisfy legal requirements or privacy concerns.

After you configure your edX instance as an LTI tool provider, you can *configure Open edX user authentication* between your Open edX system and the tool consumer.

4.15.5 Configuring Open edX User Authentication for LTI

Every learner who accesses content on an Open edX system must have a user account. The Open edX system uses the accounts to collect data for learner interactions with course content.

After you *configure your edX instance as an LTI tool provider*, you can configure Open edX user authentication between your Open edX installation and an LTI tool consumer.

For more information about the authentication flows that are available, see *Options for LTI Authentication and User Provisioning*.

- *Configure Open edX User Authentication for LTI*
- *Test LTI Authentication*

Configure Open edX User Authentication for LTI

To configure Open edX user authentication between your Open edX installation and an LTI tool consumer, follow these steps.

Note: A consumer key and secret are required. The Django administration console provides a hexadecimal string for the secret, but does not provide a hexadecimal string for the key. You must use an external tool to generate the key.

1. Sign in to the Django administration console for your base URL. For example, `http://{your_URL}/admin`.
2. In the **Third_Party_Auth** section, next to **Provider Configuration (LTI)** select **Add**.
3. Select **Enabled**.
4. Enter the **Name** of your Open edX system, as you want it to appear on the registration page that is presented to learners who access Open edX content from this LTI tool consumer.
5. To customize the registration process for learners, you can make selections for these optional fields.
 - **Skip Registration Form:** If you select this option, users are not asked to confirm any user account data that is supplied for them by the LTI tool consumer (name, email address, and so on).

By default, this option is cleared and learners review a registration form with the account details supplied by the tool consumer.

- **Skip Email Verification:** If you select this option, users are not required to confirm their email addresses, and their accounts are activated immediately upon registration.

By default, this option is cleared and learners receive an email message and must select a link in that message to activate their user accounts.

6. Enter the following information.

- **Lti consumer key:** Enter the hexadecimal string of the key.
- **Lti consumer secret:** The system generates a hexadecimal string value for this field. Alternatively, you can replace it with a secret generated by an external tool.

7. Optionally, change the default value for the **Lti max timestamp age**.

8. Select **Save** at the bottom of the page.

Test LTI Authentication

To verify the sign in process for an LTI provider configuration, follow these steps.

1. Have the LTI consumer key and secret for the LTI provider configuration available. For example, use the Django administration console to open the **Change Provider Configuration (LTI)** page.
2. Use a separate browser window or tab to open the [IMS LTI 1.1 Consumer Launch](#) page.
3. As the **Launch URL**, enter your base URL followed by `/auth/login/lti/`. For example, `http://{your_URL}/auth/login/lti/`.
4. Copy the **Lti consumer key** value, and then on the IMS LTI 1.1 Consumer Launch page paste it in as the **Key**.
5. Copy the **Lti consumer secret** value, and then on the IMS LTI 1.1 Consumer Launch page paste it in as the **Secret**.
6. Optionally, change the default values in the **Launch Data** section of the **IMS LTI 1.1 Consumer Launch** page to match the set of values that the tool consumer is configured to supply.
7. To test the workflow for a learner who does not yet have a user account on your Open edX system, follow these steps.
 - Use a separate browser window or tab to make sure that you are signed out of your Open edX LMS.
 - On the **IMS LTI 1.1 Consumer Launch** page, select **Recompute Launch Data** and then select **Press to Launch**.

The page that is configured for delivery to an unauthenticated user loads at the bottom of the page. In the example that follows, the registration page appears (that is, it was not configured to be skipped) and the learner is prompted to complete required fields.

IMS LTI 1.1 Consumer Launch

This is a very simple reference implementation of the LMS side (i.e. consumer) for IMS LTI 1.1.

[Toggle Resource and Launch Data](#)

LTI Resource

Launch URL:

Key:

Secret:

Launch Data

[toggle_debug_data](#)



[REGISTER](#)

[Sign in](#)

We couldn't create your account.

- Please enter your Email.
- Please select your Country.

You've successfully signed into We just need a little more information before you start learning with edX.

Email *

Full name *

Needed for any certificates you may earn

Public username *

The name that will identify you in your courses - (cannot be changed later)

Country *

Gender

Year of birth

Highest level of education completed

Mailing address

Tell us why you're interested in edX

☒ I agree to the edX [Terms of Service and Honor Code.](#) *

4.15. Configuring an edX Instance as an LTI Tool Provider

[Create your account](#)

* Required field

8. To test the workflow for a learner who already has a user account on your Open edX system, follow these steps.
 - Use a separate browser window or tab to sign in to your Open edX LMS.
 - On the **IMS LTI 1.1 Consumer Launch** page, select **Recompute Launch Data** and then select **Press to Launch**.

Your Open edX user account is linked to the LTI provider configuration, and your learner dashboard on the Open edX site loads at the bottom of the page. To unlink your user accounts, select the arrow next to your username, and then select **Account**. In the **Connected Accounts** section, select **Unlink** next to the LTI provider configuration name.

For more information and examples of how course teams might set up a course on an external LMS as a consumer of edX course content, see [Using Open edX as an LTI Tool Provider](#) in the *Building and Running an edX Course* guide.

4.16 Enabling Social Sharing of Courses and Certificates

This section describes how to configure Open edX so that learners can share their certificates, and so course teams can enable learners to share their courses on social media.

- [Overview](#)
- [Configure Social Sharing](#)
- [Enable Custom Course URLs](#)

4.16.1 Overview

You can enable learners to share courses and certificates that they earn on social media sites such as Facebook and Twitter.

To use this feature on your instance of Open edX, you must configure social sharing settings.

Optionally, you can also enable course teams to set custom URLs for social sharing. If a course team sets a custom course URL, posts to the social sharing site can include a link back to that URL. If you do not enable custom course URLs, a link to the course About page in the LMS is used.

Note: Before proceeding, review [Guidelines for Updating the Open edX Platform](#).

4.16.2 Configure Social Sharing

To enable social sharing icons for courses, you modify the `lms.env.json` file, which is located one level above the `edx-platform` directory.

1. In the `lms.env.json` file, modify the `SOCIAL_SHARING_SETTINGS` dictionary as needed.

```
SOCIAL_SHARING_SETTINGS = {
    'CUSTOM_COURSE_URLS': True,
    'DASHBOARD_FACEBOOK': True,
    'CERTIFICATE_FACEBOOK': True,
    'CERTIFICATE_FACEBOOK_TEXT': None,
    'CERTIFICATE_TWITTER': True,
    'CERTIFICATE_TWITTER_TEXT': None,
    'DASHBOARD_TWITTER': True,
```

```
'DASHBOARD_TWITTER_TEXT': None
}
```

- (a) For each social sharing icon that you want to enable, set the value of the setting to `True`.
 - (b) If you set `DASHBOARD_TWITTER` or `CERTIFICATE_TWITTER` to `True`, you can also specify default text that learners will see in the Twitter sharing dialog and that can be included in their tweet. Set the default text in the `DASHBOARD_TWITTER_TEXT` and `CERTIFICATE_TWITTER_TEXT` values. Learners can edit this text before they select the **Share with Twitter** button in the LMS.
 - (c) If you set `CUSTOM_COURSE_URLS` to `True`, you must *Enable Custom Course URLs*.
2. Configure the `SOCIAL_MEDIA_FOOTER_NAMES` array to the order of links you want learners to see in the footer.

```
SOCIAL_MEDIA_FOOTER_NAMES = [
    "facebook",
    "twitter",
    "youtube",
    "linkedin",
    "google_plus",
    "reddit",
]
```

3. Configure the `SOCIAL_MEDIA_FOOTER_DISPLAY` dictionary to define how you want social media icons to be displayed. For each social media icon you enable, you define a title, icon, and action.

```
"facebook": {
    "title": _("Facebook"),
    "icon": "fa-facebook-square",
    "action": _("Like {platform_name} on Facebook")
},
"twitter": {
    "title": _("Twitter"),
    "icon": "fa-twitter",
    "action": _("Follow {platform_name} on Twitter")
},
"linkedin": {
    "title": _("LinkedIn"),
    "icon": "fa-linkedin-square",
    "action": _("Follow {platform_name} on LinkedIn")
}
}
```

4. Save the `lms.env.json` file.

4.16.3 Enable Custom Course URLs

In addition to enabling the social sharing icons, you can allow course teams to provide a custom URL for social sharing sites to link back to.

You must set the `CUSTOM_COURSE_URLS` parameter to `True` in both the `lms.env.json` and `cms.env.json` files. In the `cms.env.json` file, this parameter is the only social sharing setting.

```
SOCIAL_SHARING_SETTINGS = {
    'CUSTOM_COURSE_URLS': True
}
```

When finished, save the `lms.env.json` and `cms.env.json` files.

Set a Custom URL for a Course

When you enable custom course URLs in your instance of Open edX, course teams can then set custom URLs for their courses.

In Studio **Advanced Settings**, the course team specifies the custom course URL in the **Social Media Sharing URL** setting.

This URL is provided to the social sharing site for linking back to a course location. This URL is used only if you have enabled custom URLs in your instance of Open edX.

Note: If custom URLs are enabled but a course team does not provide a value in the **Social Media Sharing URL** advanced setting in Studio, social sharing icons are not visible in the LMS for that course.

4.17 Enabling Third Party Authentication

To enhance sign in options for your users, you can enable third party authentication between institutional authentication systems and your implementation of the edX platform. After you enable third party authentication, users can register and sign in to your Open edX site with their campus or institutional credentials.

4.17.1 Supported Identity Providers

In an exchange of authentication and authorization data, an identity provider securely asserts the identity and access rights of a set of users. Your implementation of the Open edX platform is the service provider that allows the users access on the basis of credentials sent by an identity provider.

For example, your Open edX installation hosts the courses of three different institutions. When you configure the open edX installation to be a service provider, and configure each of the three institutions to be identity providers, you permit learners who have valid user credentials at any of those institutions to access the Open edX site.

You can enable third party authentication between your Open edX system and identity providers that use the SAML 2.0 (Security Assertion Markup Language, version 2.0) standard for authentication. For more information, see [Integrating with a SAML Identity Provider](#).

At an Open edX installation, you begin by [enabling the third party authentication feature](#) for your installation.

At an institution that has a partner membership with edX, you can [enable third party authentication](#) between your institutional authentication systems and the edX Edge site.

If you are using [edX as an LTI tool provider](#) to a external learning management system or application, you can set up an authentication workflow between your Open edX system and the system that is the LTI tool consumer. For more information, see [Options for LTI Authentication and User Provisioning](#) and [Configuring Open edX User Authentication for LTI](#).

4.17.2 Enable the Third Party Authentication Feature

By default, third party authentication is not enabled on edX installations. To enable this feature, follow these steps.

1. In the `edx/app/edxapp/lms.env.json` file, edit the file so that it includes the following line in the features section.

```
"FEATURES" : {
  ...
  "ENABLE_COMBINED_LOGIN_REGISTRATION": true,
  "ENABLE_THIRD_PARTY_AUTH": true
}
```

2. Save the `edx/app/edxapp/lms.env.json` file.

4.17.3 Configuring your Installation as a SAML Service Provider

The first step in configuring your Open edX installation to act as a SAML SP is to create a credential key pair to ensure secure data transfers with identity providers. To complete the configuration procedure, you configure your Open edX installation as a SAML SP, which creates your metadata XML file.

- *Generate Public and Private Keys*
- *Add Keys to the LMS Configuration File*
- *Configure your Installation as a Service Provider*
- *Ensure that the SAML Authentication Backend is Loaded*

After you complete this configuration, you can share your metadata file with SAML identity providers, and configure them to assert identity and access rights for users to your installation. For more information, see [Integrating with a SAML Identity Provider](#).

Generate Public and Private Keys

To generate the keys for your Open edX installation, follow these steps.

1. On your local computer or on the server, open Terminal or a Command Prompt and run the following command.


```
openssl req -new -x509 -days 3652 -nodes -out saml.crt -keyout saml.key
```
2. Provide information at each prompt.

Two files, `saml.crt` and `saml.key`, are created in the directory where you ran the command.

Add Keys to the LMS Configuration File

Note: Configuration settings added to the `lms.auth.json` file are reset to their default values when you use Ansible to update `edx-platform`.

To configure your Open edX installation with your public and private SAML keys, follow these steps.

1. Open the `edx/app/edxapp/lms.auth.json` file in your text editor.
2. Edit the file to add a section for the SAML keys. For example, `# SAML KEYS`.
3. In the new section, add the `SOCIAL_AUTH_SAML_SP_PUBLIC_CERT` parameter followed by a colon (:), a space, and the YAML literal style indicator (!).

```
# SAML KEYS
SOCIAL_AUTH_SAML_SP_PUBLIC_CERT: |
```

4. Open the `saml.crt` file, copy its entire contents, and then paste them onto the line after the `SOCIAL_AUTH_SAML_SP_PUBLIC_CERT` parameter.

```
# SAML KEYS
SOCIAL_AUTH_SAML_SP_PUBLIC_CERT: |
-----BEGIN CERTIFICATE-----
SWP6P/ClypaYkmS...
...j9+hjvbBf3szk=
-----END CERTIFICATE-----
```

5. Add the `SOCIAL_AUTH_SAML_SP_PRIVATE_KEY` parameter followed by a colon (:), a space, and the YAML literal style indicator (!).

```
# SAML KEYS
SOCIAL_AUTH_SAML_SP_PUBLIC_CERT: |
-----BEGIN CERTIFICATE-----
SWP6P/ClypaYkmS...
...j9+hjvbBf3szk=
-----END CERTIFICATE-----
SOCIAL_AUTH_SAML_SP_PRIVATE_KEY: |
```

6. Open the `saml.key` file, copy its entire contents, and then paste them onto the line after the `SOCIAL_AUTH_SAML_SP_PRIVATE_KEY` parameter.

```
# SAML KEYS
SOCIAL_AUTH_SAML_SP_PUBLIC_CERT: |
-----BEGIN CERTIFICATE-----
SWP6P/ClypaYkmS...
...j9+hjvbBf3szk=
-----END CERTIFICATE-----
SOCIAL_AUTH_SAML_SP_PRIVATE_KEY: |
-----BEGIN RSA PRIVATE KEY-----
WlicmlkZN+FtM5h...
...s/psgLDn38Q==
-----END RSA PRIVATE KEY-----
```

7. Save and close the `lms.auth.json` file.

Configure your Installation as a Service Provider

To configure your Open edX installation as a SAML service provider, follow these steps.

1. Sign in to the Django administration console for your base URL. For example, `http://{your_URL}/admin`.
2. In the **Third_Party_Auth** section, next to **SAML Configuration** select **Add**.
3. Select **Enabled**.
4. Enter the following information.
 - **Entity ID:** Enter a URI for the server. To ensure that this value uniquely identifies your site, the naming convention that edX recommends is to include the server's domain name. For example, `http://saml.mydomain.com/`.
 - **Organization Info:** Use the format in the example that follows to specify a language and locale code and identifying information for your installation.

```
{
  "en-US": {
```

```

        "url": "http://www.mydomain.com",
        "displayname": "{Complete Name}",
        "name": "{Short Name}"
    }
}

```

- **Other config str:** Define the security settings for the IdP metadata files. For more information about the security settings, see the [Python SAML Toolkit](#). An example follows.

```

{
    "SECURITY_CONFIG": {
        "signMetadata": false,
        "metadataCacheDuration": ""
    }
}

```

1. Optionally, you can save your public and private keys in the Django administration console. Because this procedure saves your credentials in the database, edX recommends that you use the `lms.auth.json` file instead. For more information, see [Add Keys to the LMS Configuration File](#).
2. Select **Save**. You can direct identity providers to `{your LMS URL}/auth/saml/metadata.xml` for your metadata file.

Ensure that the SAML Authentication Backend is Loaded

By default, SAML is included as an approved data format for identity providers. The default configuration of the `/edx/app/edxapp/lms.env.json` file does not explicitly include the `THIRD_PARTY_AUTH_BACKENDS` setting.

If you have customized this file and added the `THIRD_PARTY_AUTH_BACKENDS` setting to it, you might need to verify that the `third_party_auth.saml.SAMLAuthBackend` python-social-auth backend class is specified for it. That backend is required before you can add SAML IdPs.

To verify that the SAML authentication backend is loaded on a devstack or fullstack installation, review the `/edx/app/edxapp/lms.env.json` file.

4.17.4 Integrating with a SAML Identity Provider

You can integrate your Open edX installation with federated identity solutions that use the SAML 2.0 (Security Assertion Markup Language, version 2.0) standard. An example is Shibboleth, a single sign on system that is used by many educational institutions.

- [Exchange Metadata](#)
- [Add and Enable a SAML Identity Provider](#)
- [Configuration Options for SAML Identity Providers](#)
- [Test an Enabled SAML Provider](#)

Exchange Metadata

SAML metadata is an XML file that contains the information necessary for secure interactions between identity providers and security providers. You send the URL of your metadata file, created when you *configured your installation as a SAML service provider*, to each identity provider that you want to add. Similarly, you obtain the metadata URLs from identity providers before you add and enable them for your installation.

Add and Enable a SAML Identity Provider

To add and enable a SAML 2.0 identity provider, follow these steps.

1. Log in to the Django administration console for your base URL. For example, `http://{your_URL}/admin`.
2. In the **Third_Party_Auth** section, next to **Provider Configuration (SAML IdPs)** select **Add**.

Note: If you want to change the configuration of an existing provider, next to **Provider Configuration (SAML IdPs)** select **Change**, and then select **Update** for the provider that you want to configure.

3. Enter the following information for the provider.
 - **Icon class:** Specifies a [Font Awesome](#) image for the button that users will select to access the sign in page for this IdP. The **fa-sign-in** icon is used by default. For university or institutional providers, a suggested alternative is **fa-university**.
 - **Name:** The name of the IdP as you want it to appear on the sign in page.
 - **Secondary:** Select this option to include the IdP in an intermediary list of providers that users access from a **Use my institution/campus credentials** button on the sign in page.
 - **Backend name:** The default, **tpa-saml**, is optimized for use with Open edX and works with most SAML providers. Select a different option only if you have added a custom backend that provides additional functionality.
 - **IdP slug:** A short, unique name to identify this IdP in the URL. The slug becomes part of a URL, so the value that you enter cannot include spaces.
 - **Entity ID:** The URI that identifies the IdP. This ID must match the value specified in the metadata XML file.
 - **Metadata source:** The URL of the XML file that contains this provider's metadata.
4. Specify your selections for any of the other, optional configuration options. For more information about these options, see [Configuration Options for SAML Identity Providers](#).
5. When you are ready to enable the provider, select **Enabled** at the top of the page. Alternatively, save your configuration settings and enable the provider at another time.
6. Select **Save** or one of the other save options at the bottom of the page.

Next, you can [test an enabled provider](#).

Configuration Options for SAML Identity Providers

To customize the registration process for IdP, you make selections for these optional fields on the Add Provider Configuration (SAML IdP) page.

- **Skip Registration Form:** If you select this option, users are not asked to confirm the user account data supplied for them by the IdP (name, email address, and so on). Select this option only for providers that are known to provide accurate user information.

By default, users review a registration form with the supplied account details.

- **Skip Email Verification:** If you select this option, users are not required to confirm their email addresses, and their accounts are activated immediately upon registration.

By default, users receive an email message and must select a link in that message to activate their user accounts.

- **User ID Attribute:** Required. This value is used to associate the user's edX account with the campus account. It is not displayed to users.

By default, uses `userid,urn:oid:0.9.2342.19200300.100.1.1`.

- **Optional user attributes:** You can indicate specific URN values for the following user attributes.

By default, the registration form includes all of the following attributes if they are sent by the IdP.

- **Full Name Attribute:** `commonName,urn:oid:2.5.4.3`
- **First Name Attribute:** `givenName,urn:oid:2.5.4.42`
- **Last Name Attribute:** `surname,urn:oid:2.5.4.4`
- **Username Hint Attribute:** `userid,urn:oid:0.9.2342.19200300.100.1.1`
- **Email Attribute:** `mail,urn:oid:0.9.2342.19200300.100.1.3`

If the identity provider sends a value that you do not want to be included on the the registration form, you can enter a value such as “DISABLED” or “IGNORE” in that field.

Test an Enabled SAML Provider

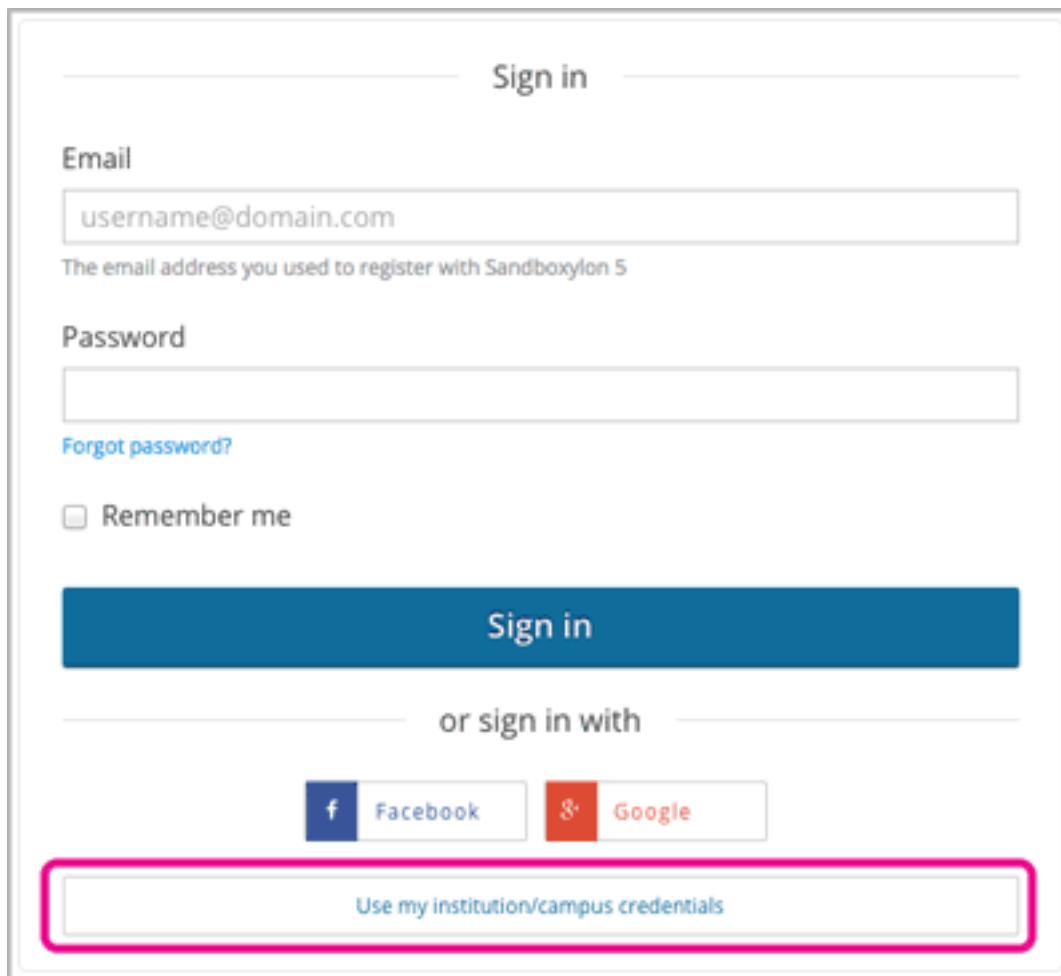
To verify the sign in process for an IdP that you have enabled, follow these steps.

1. On the Django administration console, in the **Third_Party_Auth** section, select **Provider Configuration (SAML IdPs)**.
2. Check the icon in the **Metadata ready** column for the IdP. After the provider's metadata is fetched successfully from the URL that you provided as the metadata source, a check mark in a green circle appears and the provider is ready for use immediately.

You might need to wait 30-60 seconds for the task to complete, and then refresh this page.

If the check mark does not appear, make sure that celery is configured correctly and is running. You can also manually trigger an update by running the management command `./manage.py lms saml --pull --settings=aws` on fullstack or `./manage.py lms saml --pull --settings=devstack` on devstack.

3. For additional information about the data fetched from the IdP, on the Django administration console select **SAML Provider Data**, and then select the provider. The page that opens reports data fetched from the metadata source URL and the date and time it was fetched.
4. To verify that users can use the IdP for sign in, go to the sign in page for your LMS. The page should include the institutional sign in button.



The image shows a web form for signing in to the Open edX platform. At the top, the text "Sign in" is centered. Below it, there is an "Email" label and a text input field containing "username@domain.com". A note below the email field states: "The email address you used to register with Sandboxylon 5". Below the email field is a "Password" label and a password input field. A link "Forgot password?" is located below the password field. Below the password field is a checkbox labeled "Remember me". A large blue "Sign in" button is positioned below the "Remember me" checkbox. Below the button, the text "or sign in with" is centered. Underneath this text are two buttons: "Facebook" (with a blue 'f' icon) and "Google" (with a red 'g' icon). At the bottom of the form, there is a button labeled "Use my institution/campus credentials", which is highlighted with a thick pink rectangular border.

5. Select **Use my institutional/campus credentials**. The list of providers that appears should include the IdP that you enabled.

Sign in with Institution/Campus Credentials

Choose your institution from the list below:

- Georgetown
- Harvard
- MIT
- Stanford
- University of British Columbia
- University of California, Berkeley (Coming soon)

or

[Back to sign in](#)

4.17.5 Enabling Third Party Authentication with edX Edge

Institutions that have partner memberships with edX can enable third party authentication between their campus or institutional authentication systems and the edX Edge site. Learners at sites that enable third party authentication can use their campus credentials to authenticate into edX Edge.

These procedures require collaboration between members of the DevOps (development operations) or IT teams at your partner institution and the DevOps team at edX, facilitated by your edX partner manager.

- *Integrating with Shibboleth (SAML) Systems*
 - *Validate Integration*
 - *Obtain edX SAML Metadata*
 - *Add edX as a Service Provider*
 - *Configure User Attributes*
 - *Send Shibboleth Configuration Data to edX*
 - *Test Edge Registration*

Integrating with Shibboleth (SAML) Systems

SAML 2.0 (Security Assertion Markup Language, version 2.0) is the standard that edX uses for the exchange of authentication and authorization data with institutional partners. Because it is built with SAML, this service is compatible with the Shibboleth single sign on system.

In the exchange of authentication and authorization data, your edX partner institution is an identity provider (IdP) that securely asserts the identity and access rights of a set of users. EdX is the service provider (SP) that allows the users access on the basis of those credentials.

All procedures described in this section should be performed by a member of your institution's IT team who is familiar with your institutional Shibboleth system. The designated IT contact must have access to test and production accounts, and be able to make configuration changes.

Validate Integration

Integrating an institutional IdP with edX Edge requires a preliminary validation phase.

1. All procedures are completed between a test Shibboleth account at the partner institution and the edX staging instance (“edX stage”).
2. After validation, all procedures are completed between the partner’s production system and edX Edge.

Your designated IT contact must set up or identify a Shibboleth test account, and securely transmit its credentials to edX, before integration testing can begin.

Note: The edX DevOps team prefers PGP encryption of account credentials. Other secure methods for transmitting credentials can be accommodated if necessary.

Obtain edX SAML Metadata

During the testing phase, you use a [test metadata file](#) to obtain the information necessary for interaction between edX stage and your test account.

When testing is complete, you access the service provider SAML metadata for Edge in an [XML file](#) on the Edge website. This file contains the information necessary for interaction between Edge as service provider and your campus production Shibboleth system as identity provider.

Add edX as a Service Provider

On your Shibboleth system, use the SAML metadata obtained from the XML file to add edX stage, and then edX Edge, to your whitelist of authorized service providers.

For example, you might add the following information to your `$IDP_HOME/conf/relying-party.xml` file for edX Edge.

```
<MetadataProvider xsi:type="FileBackedHTTPMetadataProvider" xmlns="urn:mace:shibboleth:2.0:metadata"
    id="edxEdgeMetadata"
    metadataURL="https://edge.edx.org/auth/saml/metadata.xml"
    backingFile="/tmp/idp-metadata-edx-edge.xml" />
```

For more information about defining a metadata source on a Shibboleth system, see the [Shibboleth configuration wiki](#).

Configure User Attributes

You work with edX to ensure that your identity provider will assert the user information that you want learners to see during initial sign in on edX Edge. When you [send your Shibboleth configuration data](#) to your institution’s edX partner manager, you include any customizations to the attributes that will be asserted.

Selecting User Attributes to Assert By default, the edX platform uses these attributes if your system provides them.

- User identifier: `userid`, `urn:oid:0.9.2342.19200300.100.1.1`. Required. This value is used to associate the user’s edX account with the campus account. It is not displayed to users.
- Full name: `commonName`, `urn:oid:2.5.4.3`
- First name: `givenName`, `urn:oid:2.5.4.42`

- Last name: surname, urn:oid:2.5.4.4
- Suggested username: userid, urn:oid:0.9.2342.19200300.100.1.1
- Email address: mail, urn:oid:0.9.2342.19200300.100.1.3

At your request, edX can configure Edge to use only some of these attributes. The only required attribute is the user identifier. If you choose not to provide an attribute, learners are prompted to enter that information themselves during initial sign in.

Specifying Alternative Attributes You can identify different attributes to assert user information than those listed above. For example, you might want to send the suggested username as `eduPersonPrincipalName` instead of `userid`.

Restricting Access with Attribute Assertions You can restrict access to edX Edge to a subset of your users using attributes defined on your Shibboleth system. For example, you might want to allow currently matriculated students to sign in to Edge, but not alumni. The `eduPersonEntitlement` attribute can be used to restrict access in this way.

Note: If you want access to be restricted to certain users, be sure to let edX know what provider assertions to use to determine access rights.

Send Shibboleth Configuration Data to edX

To complete the integration between a Shibboleth system and an edX system, you send a copy of the assertion document with the following information to your institution's edX partner manager.

- Metadata: The URL for your Shibboleth IdP metadata XML file.
- User Attributes: A list of the values that you want your Shibboleth system to assert when users sign in to edX Edge.

For more information about how you can work with edX to configure user attributes effectively, see [Configure User Attributes](#).

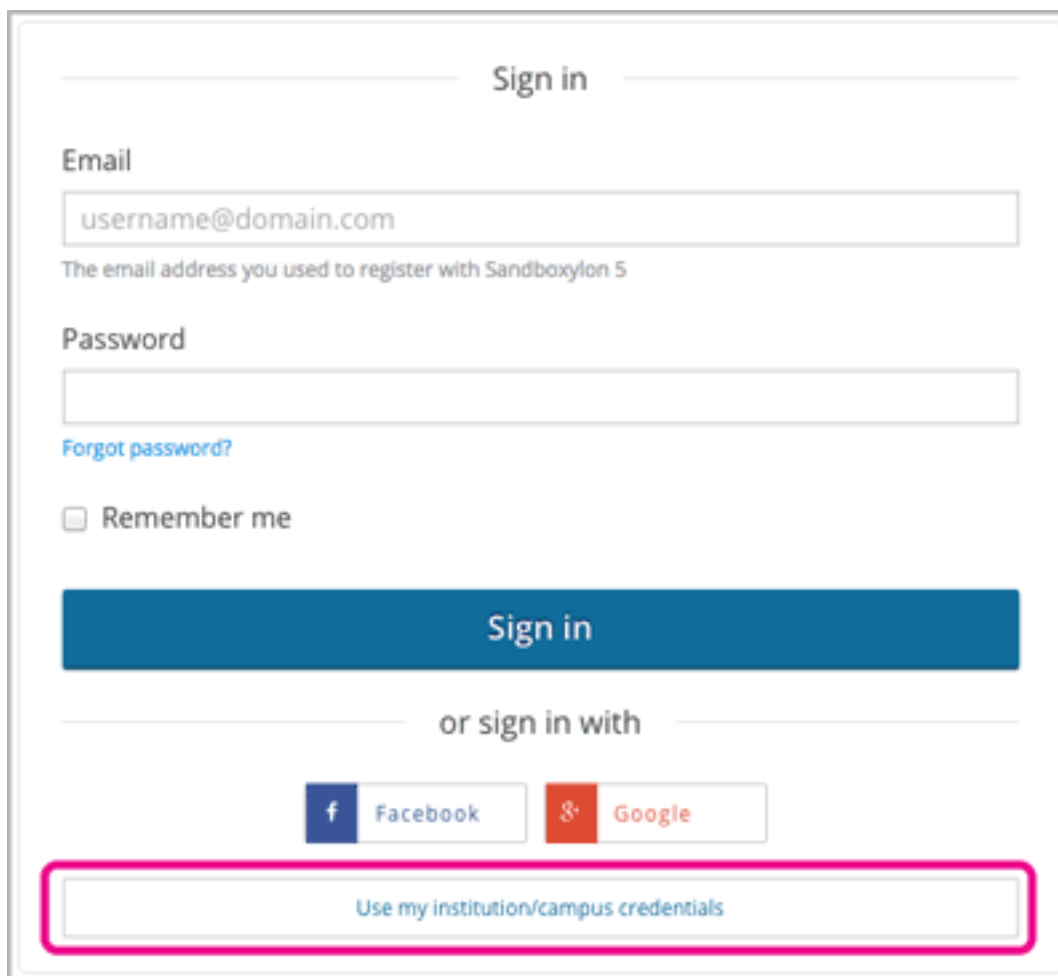
You send this information for both your test and production accounts so that integration can be validated.

Your edX partner manager notifies you when integration with your IdP is complete.

Test Edge Registration

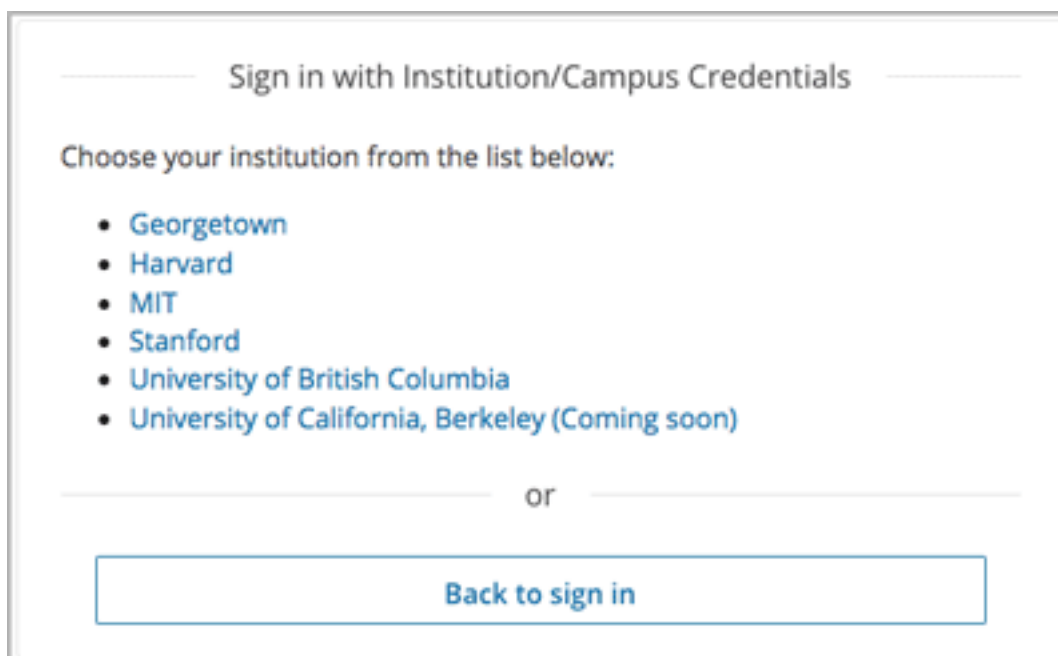
To verify that users can use their campus or institutional credentials for your IdP to sign in to Edge, follow these steps.

1. Go to the [Edge registration](#) page. The page should include the institutional sign in button.



The image shows the Open edX Sign in page. At the top, there is a "Sign in" header. Below it, there are two input fields: "Email" and "Password". The email field contains the placeholder text "username@domain.com". Below the email field, there is a note: "The email address you used to register with Sandboxylon 5". Below the password field, there is a link: "Forgot password?". Below these fields, there is a checkbox labeled "Remember me". Below the checkbox, there is a large blue button labeled "Sign in". Below the button, there is a section titled "or sign in with" with two buttons: "Facebook" and "Google". Below these buttons, there is a pink rectangular box containing the text "Use my institution/campus credentials".

2. Select **Use my institutional/campus credentials**. The list of providers that appears should include your IdP.



The image shows the Open edX Sign in with Institution/Campus Credentials page. At the top, there is a header: "Sign in with Institution/Campus Credentials". Below it, there is a text: "Choose your institution from the list below:". Below this text, there is a list of institutions: "Georgetown", "Harvard", "MIT", "Stanford", "University of British Columbia", and "University of California, Berkeley (Coming soon)". Below the list, there is a section titled "or" with a button labeled "Back to sign in".

3. Select your own IdP. The landing page for your Shibboleth system should open.

This section includes information for teams involved in identity management at Open edX installations, including DevOps (development operations) and IT. The *Enabling Third Party Authentication with edX Edge* topic describes procedures for members of the DevOps and IT teams at an edX partner institution.

4.18 Enabling Timed Exams

This topic describes how to enable the timed exams feature in your instance of Open edX.

- *Overview*
- *Enable Timed Exams in Studio and the Learning Management System*

4.18.1 Overview

Course teams can configure course subsections to limit the amount of time that learners have to complete problems in that subsection.

To use this feature on your instance of Open edX, you must enable the timed exams feature in Studio and the Learning Management System.

For information about how course teams set up timed exams, see the *Timed Exams* topic in *Building and Running an Open edX Course*. For information about the learner experience, see *Taking a Timed Exam* in the *Open edX Learner's Guide*.

Note: Before proceeding, review *Guidelines for Updating the Open edX Platform*.

4.18.2 Enable Timed Exams in Studio and the Learning Management System

To enable timed exams, you modify the `lms.env.json` and `cms.env.json` files, which are located one level above the `edx-platform` directory.

1. Set the value of `ENABLE_SPECIAL_EXAMS` in the `lms.env.json` and `cms.env.json` files to `true`.

```
# Timed exams feature flag
'ENABLE_SPECIAL_EXAMS': true,
```

2. Save the `lms.env.json` and `cms.env.json` files.
3. Restart the Studio (CMS) and Learning Management System (LMS) processes so that your updates are loaded.

4.19 Setting Up the YouTube API Key

This topic describes how to set the YouTube API key for your instance of Open edX.

- *Overview*
- *Get a YouTube API Key*
- *Install the YouTube API Key in Open edX*

4.19.1 Overview

If you intend for courses on your Open edX instance to include videos that are hosted on YouTube, you must get a YouTube API key and set the key in the Open edX Platform.

The Open edX Platform uses the [YouTube Data API v3](#), which requires that the application uses an API key.

4.19.2 Get a YouTube API Key

To get the YouTube API key, follow YouTube's [instructions for obtaining authorization credentials](#). YouTube provides two different options for API keys: server keys or browser keys. You should use a **browser key** for Open edX.

Note: Before proceeding, review [Guidelines for Updating the Open edX Platform](#).

4.19.3 Install the YouTube API Key in Open edX

After you obtain a YouTube API key, you must install that key into your Open edX installation. There are two different ways you can do this.

- *Option 1: Ansible (recommended)*
- *Option 2: JSON files*

Option 1: Ansible (recommended)

[Ansible](#) is the automation system used for installing and updating Open edX. If you set your YouTube API key in Ansible's configuration file, then Ansible will make sure that the YouTube API key remains in place when you update Open edX.

To set your YouTube API key in Ansible's configuration file, follow these steps.

1. Find the [configuration](#) repository on your Open edX server. If you are running devstack or fullstack, the directory is `/edx/app/edx_ansible/edx_ansible`.
2. In that repository, open the `playbooks/roles/edxapp/defaults/main.yml` file in a text editor.
3. Find the line for the YouTube API key.

```
EDXAPP_YOUTUBE_API_KEY: "PUT_YOUR_API_KEY_HERE"
```

Replace `PUT_YOUR_API_KEY_HERE` with your YouTube API key. Ensure that the YouTube API key is within by quotation marks.

4. Save and close the file.
5. Run Ansible so that it applies your YouTube API key to your Open edX installation.

For example, if you are running the Open edX Cypress release, run the following command.

```
/edx/bin/update edx-platform named-release/cypress
```


Option 2: JSON files

Ansible outputs information to several JSON files used by Open edX. If you prefer not to edit the Ansible configuration, you can edit these files directly.

However, every time you update Open edX, your edits will be overwritten by Ansible. As a result, we recommend setting your YouTube API key in Ansible's configuration instead.

To set your YouTube API key by editing JSON files, complete the following steps.

1. Find the `edx-platform` repository on your Open edX server. If you are running devstack or fullstack, the directory is `/edx/app/edxapp/edx-platform`.
2. In the directory *above* your repository, there should be several JSON files, including `lms.auth.json` and `cms.auth.json`. If you are running devstack or fullstack, the directory is `/edx/app/edxapp`.
3. Open the `lms.auth.json` file in your text editor.
4. Find the line for the YouTube API key.

```
"YOUTUBE_API_KEY": "PUT_YOUR_API_KEY_HERE",
```

Replace `PUT_YOUR_API_KEY_HERE` with your YouTube API key. Verify that the YouTube API key is between the quotation marks.

5. Save and close the file.
6. Open the `cms.auth.json` file and make the same change. If that line does not exist in this file, create it.
7. Save and close the file.

4.20 Installing an XBlock

The XBlock framework allows developers to expand the Open edX platform by building different learning experiences and deploying them as XBlocks. Before course teams can use an XBlock in courses running on an instance of the Open edX platform, both of the following tasks must be completed.

- A system administrator installs the XBlock in the instance of the Open edX platform.
- Course teams enable the XBlock in the specific courses that will use it.

To install an XBlock, follow these steps.

1. Obtain the GitHub location and commit, or PyPi package name and version, for the XBlock.
2. Run `pip` along with either the GitHub link to the XBlock or the PyPi package name.

An example of the GitHub link that installs the Oppia XBlock follows.

```
pip install git+https://github.com/oppia/xblock.git@9f6b95b7eb7dbabb96b77198a3202604f96adf65#egg=xblock
```

An example of the PyPi package name that installs the Peer Instruction XBlock follows.

```
pip install ubcpi-xblock==0.4.4
```

The course teams that want to include components that use the XBlock can then enable the XBlock for their courses. To do so, they add the name specified in the XBlock's `setup.py` file to each course's advanced module list. For more information, see [Enabling Additional Exercises and Tools](#).

4.21 Enabling a CDN for Course Assets

By default, all course assets are served directly from your Open edX instance. For courses with large enrollments, or courses with large assets, such as high-quality images, PDFs, or video files, this can increase not only the load on the instance but also the time it takes for learners to load the courses on their computers and mobile devices.

You can configure your Open edX instance to serve assets from a content delivery network (CDN) instead. Using a CDN offloads the work required to deliver assets from your Open edX instance.

Note: Whether you need a CDN depends on the type, and amount, of content in your courses. Not all installations need a CDN for their course assets.

- *Overview*
- *Guidelines for CDN Configuration*
- *Enable the CDN*

4.21.1 Overview

A CDN, or content delivery network, is a service that places servers all around the world, and keeps copies of content on those servers. Instead of serving files from one location, a CDN serves them from whichever server is closest to the end user. This results in faster download speeds and, ultimately, faster page loads.

4.21.2 Guidelines for CDN Configuration

When you configure a CDN for use with your Open edX instance, you should identify your Open edX instance as the origin server.

Choose the cache expiration carefully: you want content to be cached long enough to keep the load on your Open edX instance low, but not so long that changes made to course assets are not realized within a reasonable amount of time. As a starting point, edX recommends a cache expiration period of one hour.

4.21.3 Enable the CDN

After you configure your CDN, follow these steps.

1. Sign in to the Django administration console for your base URL. For example, `http://{your_URL}/admin`.
2. In the **Static_Replace** section, next to **Asset base url configs**, select **Add**.
3. Select **Enabled**.
4. Enter the hostname for your CDN provider.

For example, if you were using CloudFront, this would look something like `d37djvu3ytnwxt.cloudfront.net`.

Be sure not to include the scheme (`http://` or `https://`) when you specify the hostname.

5. Select **Save**.

Adding edX Insights for Course Teams

The following topics provide information about installing, configuring, and running [edX Insights](#) and its dependencies in a production environment.

5.1 Options for Installing edX Insights

This topic is intended for those who are interested in running [edX Insights](#) and its dependencies in a production environment. This topic does not provide complete installation procedures for edX Insights. It presents the introductory material that is available now.

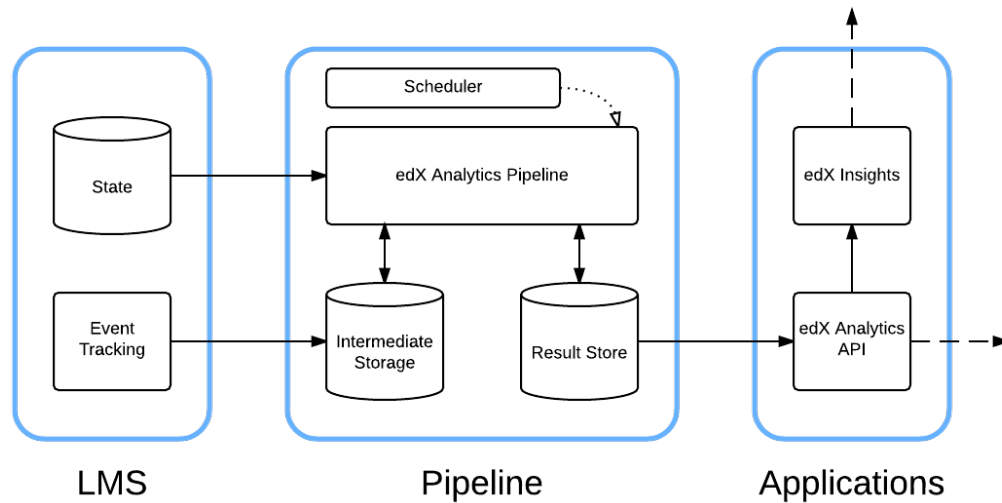
- *Overview*
- *What You Should Know Before You Start*
- *Planning Your Deployment*
- *Example Deployments*

5.1.1 Overview

Course teams use edX Insights to access data gathered from active courses. In edX Insights, course teams can display charts, summary statistics, and data tables.

The Learning Management System (LMS) gathers data about learner activity. This data is aggregated by the edX Analytics Pipeline. The aggregated data is exposed by the edX Analytics Data API. EdX Insights reads the data from the edX Analytics Data API and presents the data to course team members.

Architecture Diagram



Components

- *LMS*
- *edX Analytics Pipeline*
- *Scheduler*
- *edX Analytics Data API*
- *edX Insights*

LMS

The LMS records learner actions in tracking log files. The standard `logrotate` utility periodically compresses and copies these files into a file system that can be read by the edX Analytics Pipeline. The LMS also captures a lot of information in a MySQL database. The edX Analytics Pipeline connects directly to this database to extract information about learners.

edX Analytics Pipeline

The edX Analytics Pipeline reads the MySQL database used by the LMS as well as the tracking log files produced by the LMS. The data is processed and the resulting summary data is published to the result store. The result store is a MySQL database.

Requirements

- [Hadoop](#) version 1.0.3 or higher

- [Hive](#) version 0.11.0.2 or higher
- [Sqoop](#) version 1.4.5
- Python 2.7
- Either Debian version 6.0 or higher, or Ubuntu version 12.04 or higher
- A MySQL server version 5.6 or higher

Scheduler

The Scheduler schedules the execution of data computation tasks. Data computation tasks are run by the edX Analytics Pipeline. Data computation tasks are used to update parts of the result store.

edX Analytics Data API

The `ref:opendataapi:edX Analytics Data API` <edX Data Analytics API Overview> provides an HTTP interface for accessing data in the result store. Typically, the data in the result store is updated periodically by the edX Analytics Pipeline.

Requirements Python 2.7

edX Insights

EdX Insights uses the edX Analytics Data API to present data to users. Users access the data using a supported web browser. EdX Insights communicates directly with the LMS to authenticate users, authorize users, and read course structure information.

Requirements Python 2.7

5.1.2 What You Should Know Before You Start

To install edX Insights and deploy the edX Analytics Pipeline, you must understand the following concepts.

- Basic terminal usage.
- How the LMS has been deployed and configured.
- Basic computer network terminology.
- YAML file format.

If you plan to use Amazon Web Services, an understanding of AWS terminology is also required.

5.1.3 Planning Your Deployment

All edX Analytics services are designed to be relocatable. This means that they do not require a particular configuration of virtual servers. You are free to choose how the services should be distributed among the resources you have available.

Hadoop

Most of the computation performed by the edX Analytics Pipeline is implemented as Map Reduce jobs that must be executed by a Hadoop cluster. You can scale your Hadoop cluster based on your current and projected data sizes. Hadoop clusters can be scaled vertically and horizontally as your data grows. For very small installations of Open edX, a single virtual server should be sufficiently powerful to process your data.

Amazon's [Elastic MapReduce](#) (EMR) service offers preconfigured Hadoop clusters. If you are able to use Amazon Web Services (AWS), use of this service is recommended. Proper installation and configuration of Hadoop can be time consuming. For more information, see [Using Elastic MapReduce on AWS](#).

Additionally, vendors such as Cloudera and MapR offer simplified Hadoop administration experiences.

Hadoop is a distributed system that consists of several different services. It is worth noting that they are Java services and require a non-trivial amount of memory to run. The high memory requirement might prevent you from running all services on the same virtual server if it does not have enough memory available.

edX Applications

The edX Analytics Data API responds to a small number of requests every time a page is loaded in edX Insights. Small installations can probably host both services on the same virtual server. Larger installations should consider hosting the services on more than one virtual server. A load balancer is recommended for each service that requires more than one virtual server.

Result Store

The results of computations performed by the edX Analytics Pipeline are stored in a MySQL database. Even small installations should use a different MySQL server than the one used by the LMS. The edX Analytics Pipeline's write patterns to the result store are more I/O intensive than usual. Placing both databases on the same server can degrade the performance of the LMS.

Scheduler

Scheduling executions of the edX Analytics Pipeline can be accomplished in many different ways. Any tool that can periodically execute shell commands should work. The simplest tool that can perform this task is [cron](#). [Jenkins](#) is also a good candidate.

5.1.4 Example Deployments

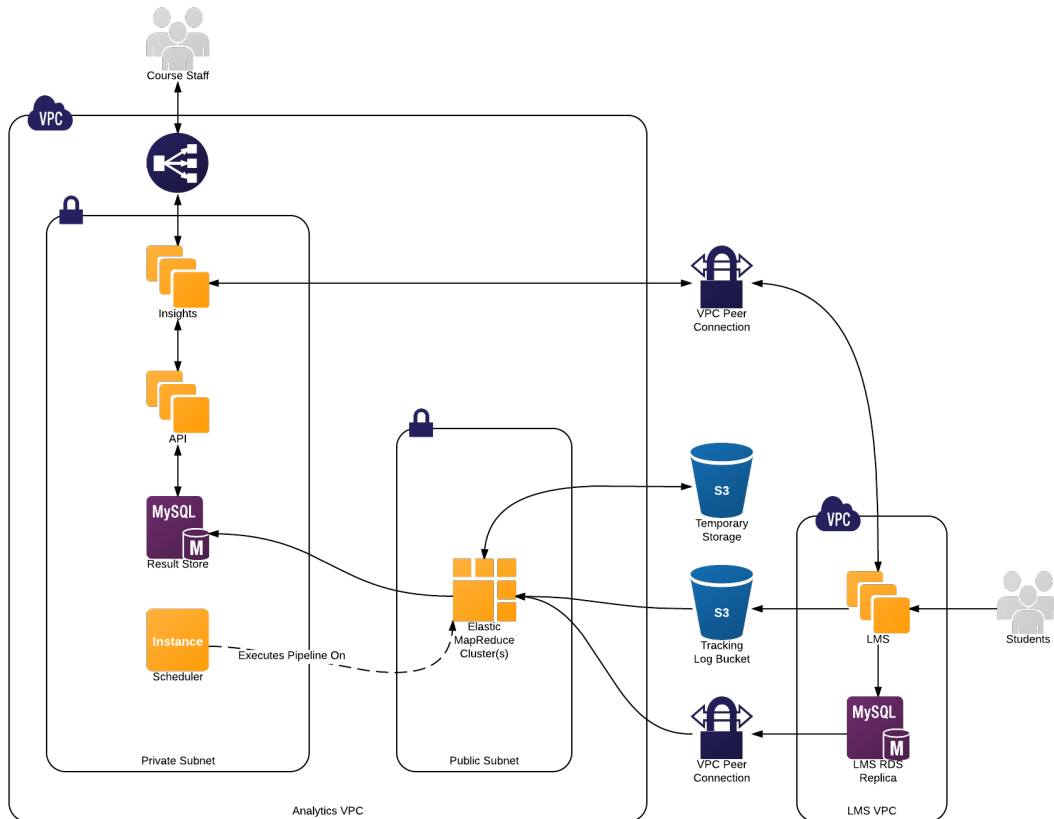
- *Small Scale Using Elastic MapReduce*
- *Large Scale Using Elastic MapReduce*
- *Large Scale Without Using Elastic MapReduce*

Small Scale Using Elastic MapReduce

A small deployment might consist of a single master node and a single core node. The Scheduler is deployed to the master node and periodically executes the edX Analytics Data Pipeline on this server. Additionally, the edX Analytics API, edX Insights and result store are deployed to the master node. These services run continuously. For more information, see [Using Elastic MapReduce on AWS](#).

Large Scale Using Elastic MapReduce

A large scale deployment consists of a single master node, several core nodes, and many task nodes deployed into a public subnet of a Virtual Private Cloud.



The edX Analytics API and edX Insights are each deployed into an auto-scaling group behind an Elastic Load Balancer which terminates SSL connections and distributes the load among the application servers. The application servers are deployed into a private subnet of the Virtual Private Cloud. A single virtual server is deployed into a private subnet to host the Scheduler. The Relational Database Service is used to deploy a MySQL server into a private subnet. The MySQL database will be used as the result store. For more information, see [Using Elastic MapReduce on AWS](#).

Large Scale Without Using Elastic MapReduce

A large deployment that does not use Elastic MapReduce requires additional configuration steps to establish a properly configured environment. A Hadoop cluster is deployed. The master node is considered the node that is running the Job Tracker service for Hadoop 1.X deployments or the Resource Manager service for Hadoop 2.X deployments. Hive and Sqoop are deployed to the master node. Several servers are deployed outside of the Hadoop cluster that host the remainder of the infrastructure. The edX Analytics API and edX Insights services are each deployed to at least one server. The Scheduler is deployed to another server. A MySQL database is deployed to a server that is configured to host a relational database.

5.2 Using Elastic MapReduce on AWS

This topic provides an overview of the components and services that you set up and configure in a deployment that uses Amazon EMR. Work to prepare complete installation procedures for edX Insights is in progress.

- *Virtual Private Cloud*
- *Identity and Access Management*
- *SSH Credentials*
- *Elastic Compute Cloud*
- *Relational Database Service*
- *Scheduler Service*
- *Simple Storage Solution Buckets*
- *MySQL Connector Library*
- *Cluster Configuration File*
- *EMR Cluster*

For more information about AWS, including detailed procedures, see the [AWS Documentation](#).

Important: The tasks described by this topic rely on the use of third-party services and software. Because the services and software are subject to change by their owners, the information provided here is intended as a guideline and not as exact procedures.

5.2.1 Virtual Private Cloud

Create a Virtual Private Cloud (VPC) with at least one public subnet. A limitation of EMR running in a VPC is that clusters must be deployed into public subnets of VPCs. An existing VPC can be used if one is already available for use.

EdX recommends that you configure the VPC to have at least one private subnet in addition to the required public subnet. A private subnet is not required. For more information, see the [example configuration scenario](#) in the *Amazon Virtual Private Cloud User Guide* published by AWS.

An example configuration that includes only a [single public subnet](#) can also be found in the *Amazon Virtual Private Cloud User Guide*.

Other Considerations

- To take advantage of price fluctuations in the [spot pricing market](#), you can also deploy several public subnets in different availability zones.
- Consider that the clusters deployed using EMR will need network connectivity to a read replica of the LMS database. EdX recommends that you use the same VPC as the LMS if possible.
- While it is possible to run all of these services outside of a VPC, edX does not recommend doing so.

5.2.2 Identity and Access Management

For Amazon Identity and Access Management (IAM), create an IAM role for use by the EMR cluster. Assign all of the Elastic Compute Cloud (EC2) nodes in the cluster to this role. An option is to consider copying the contents of the default [EC2 role for Amazon EMR](#) used by AWS.

Make sure that an IAM user with administrative privileges is available for use.

5.2.3 SSH Credentials

Generate a secure SSH key that you use to access all AWS resources. For example, run the following command.

```
ssh-keygen -t rsa -C "your_email@example.com"
```

Upload the public key from the SSH key pair to AWS and assign a key pair name to it.

5.2.4 Elastic Compute Cloud

Ensure that at least one Amazon Elastic Compute Cloud (EC2) instance is available to host the various services. Depending on the scale of your deployment, additional instances might be necessary.

Make sure to use the secure key pair name to deploy all of the EC2 instances.

5.2.5 Relational Database Service

Deploy a MySQL 5.6 Relational Database Service (RDS) instance. This RDS instance is the result store.

Connectivity

Ensure that there is connectivity between the EC2 instance that hosts the edX Analytics API and your RDS instance.

Also, ensure that instances deployed into the public subnet of the VPC can connect to your RDS instance.

Users

Create at least the following two users on the RDS instance.

- The “pipeline” user must have permission to create databases, tables, and indexes, and issue arbitrary Data Manipulation Language (DML) commands.
- The “api” user must have read-only access.

Configure passwords for both of these users.

Other Considerations

Configure the RDS instance to use “utf8_bin” collation by default for columns in all databases and tables.

5.2.6 Scheduler Service

Establish an SSH connection to the EC2 instance within the VPC that will run the scheduler service. Then, issue the following commands from a shell running on that instance.

1. Check out the sources files from [edx-analytics-configuration](https://github.com/edx/edx-analytics-configuration).

```
git clone https://github.com/edx/edx-analytics-configuration.git
```

2. Configure the shell to use the AWS credentials of the administrative AWS user.

```
export AWS_ACCESS_KEY_ID=<access key ID goes here>
export AWS_SECRET_ACCESS_KEY=<secret access key goes here>
```

3. Install the AWS Command Line Interface.

```
pip install awscli
```

5.2.7 Simple Storage Solution Buckets

Create a Simple Storage Solution (S3) bucket to hold all Hadoop logs from the EMR cluster.

```
aws s3 mb s3://<your logging bucket name here>
```

Then, create an S3 bucket to hold secure configuration files and initialization scripts.

```
aws s3 mb s3://<your configuration bucket name here>
```

5.2.8 MySQL Connector Library

Download the [MySQL connector library](#) from Oracle. After the download is complete, you then upload it to S3.

```
aws s3 cp /tmp/mysql-connector-java-5.1.*.tar.gz s3://<your configuration bucket name here>/
```

The `edx-analytics-configuration/batch/bootstrap/install-sqoop` script references a specific version of the MySQL connector library. Update this `install-sqoop` script to point to the correct version of the library in the S3 bucket. You must update the script before you continue.

Then, upload the contents of the `edx-analytics-configuration/batch/bootstrap/` directory into your configuration bucket.

```
aws s3 sync edx-analytics-configuration/batch/bootstrap/ s3://<your configuration bucket name here>
```

5.2.9 Cluster Configuration File

Create a cluster configuration file to specify the parameters for the EMR cluster. Review the parameters that follow, and change them to specify your desired configuration. For example, review the `core` and `task` mappings and change the values for `bidprice` and `type` to meet your needs.

Then, save this file to a temporary location such as `/tmp/cluster.yml`.

```
{
  name: <your cluster name here>,
  keypair_name: <your keypair name here>,
  vpc_subnet_id: <your VPC public subnet ID here>,
  log_uri: "s3://<your logging bucket name here>",
  instance_groups: {
    master: {
      num_instances: 1,
      type: m1.medium,
      market: ON_DEMAND,
    },
    core: {
      num_instances: 2,
      type: m1.medium,
      market: SPOT,
    },
  },
}
```

```

        bidprice: 0.03
    },
    task: {
        num_instances: 1,
        type: m1.medium,
        market: SPOT,
        bidprice: 0.03
    }
},
bootstrap_actions: {
    security: {
        path: "s3://<your configuration bucket name here>/security.sh"
    },
    jobtrackerconfig: {
        path: "s3://elasticmapreduce/bootstrap-actions/configure-hadoop",
        args: [
            -m, "mapred.jobtracker.completeuserjobs.maximum=5",
            -m, "mapred.job.tracker.retiredjobs.cache.size=50",
            -m, "mapred.job.shuffle.input.buffer.percent=0.20"
        ]
    }
},
steps: [
    { type: hive_install },
    {
        type: script,
        name: Install Sqoop,
        step_args: [
            "s3://<your configuration bucket name here>/install-sqoop",
            "s3://<your configuration bucket name here>"
        ]
    }
],
user_info: []
}

```

5.2.10 EMR Cluster

Deploy the EMR cluster.

```
EXTRA_VARS="@/tmp/cluster.yml" make provision.emr
```

Example Output

```

pip install -q -r requirements.txt

ansible-playbook --connection local -i 'localhost,' batch/provision.yml -e "$EXTRA_VARS"

PLAY [Provision cluster] *****

TASK: [provision EMR cluster] *****
changed: [localhost]

TASK: [add master to group] *****
ok: [localhost]

```

```
TASK: [display master IP address] *****
ok: [localhost] => {
  "msg": "10.0.1.236"
}

TASK: [display job flow ID] *****
ok: [localhost] => {
  "msg": "j-29UUJVM8P1NPY"
}

PLAY [Configure SSH access to cluster] *****

TASK: [user | debug var=user_info] *****
ok: [10.0.1.236] => {
  "item": "",
  "user_info": []
}

TASK: [user | create the edxadmin group] *****
changed: [10.0.1.236]

TASK: [user | ensure sudoers.d is read] *****
changed: [10.0.1.236]

TASK: [user | grant full sudo access to the edxadmin group] *****
changed: [10.0.1.236]

TASK: [user | create the users] *****
skipping: [10.0.1.236]

TASK: [user | create .ssh directory] *****
skipping: [10.0.1.236]

TASK: [user | assign admin role to admin users] *****
skipping: [10.0.1.236]

TASK: [user | copy github key[s] to .ssh/authorized_keys2] *****
skipping: [10.0.1.236]

TASK: [user | copy additional authorized keys] *****
skipping: [10.0.1.236]

TASK: [user | create bashrc file for normal users] *****
skipping: [10.0.1.236]

TASK: [user | create .profile for all users] *****
skipping: [10.0.1.236]

TASK: [user | modify shell for restricted users] *****
skipping: [10.0.1.236]

TASK: [user | create bashrc file for restricted users] *****
skipping: [10.0.1.236]

TASK: [user | create sudoers file from template] *****
changed: [10.0.1.236]

TASK: [user | change home directory ownership to root for restricted users] ***
```

```
skipping: [10.0.1.236]

TASK: [user | create ~/bin directory] *****
skipping: [10.0.1.236]

TASK: [user | create allowed command links] *****
skipping: [10.0.1.236]

PLAY RECAP *****
10.0.1.236      : ok=0    changed=4    unreachable=0    failed=0
localhost     : ok=4    changed=1    unreachable=0    failed=0
```

Additional Tasks

To complete the EMR configuration, additional configuration and automation procedures are required, including scheduling jobs and automating log duplication. For more information, see the [edX Analytics Installation](#) wiki page.

Work to prepare complete installation procedures for edX Insights is in progress. This section presents the introductory, overview material that is available now.

Adding E-Commerce to the Open edX Platform

EdX uses a Django application called `ecommerce` to provide the platform with ecommerce functionality. This [E-Commerce service](#) extends [Oscar](#), an open source Django ecommerce framework, to manage the edX product catalog and handle orders for those products. The following sections describe how to install and use the E-Commerce service with the Open edX platform.

6.1 Install and Start the E-Commerce Service

To install and start the edX E-Commerce service, you must complete the following steps.

- *Set Up a Virtual Environment*
- *Run Migrations*
- *Configure edX OpenID Connect (OIDC)*
- *Configure a Site, Partner, and Site Configuration*
- *Start the Server*
- *Development Outside Devstack*

Note: These steps assume that you are running Open edX [devstack](#). If you prefer to run the E-Commerce service locally on your computer instead of on the virtual machine (VM) that devstack uses, see [Development Outside Devstack](#).

6.1.1 Set Up a Virtual Environment

1. Create or activate a Python virtual environment. You execute all of the commands described in this section within the `virtualenv` (unless otherwise noted).

For more information, see [Virtual Environments](#).

2. Run the following command to install dependencies.

```
$ make requirements
```

3. (Optional) Create settings overrides that you do not commit to the repository. To do this, create a file named `ecommerce/settings/private.py`. The `ecommerce/settings/local.py` file reads the values in this file, but Git ignores the file.

6.1.2 Run Migrations

To set up the `ecommerce` database, you must run migrations.

Note: Local installations use SQLite by default. If you use another database backend, make sure you update your settings and create the database, if necessary, before you run migrations.

1. To run migrations, execute the following command.

```
$ make migrate
```

When you run migrations, the E-Commerce service adds a default site to your installation.

6.1.3 Configure edX OpenID Connect (OIDC)

The E-Commerce service relies on the edX [OpenID Connect](#) (OIDC) authentication provider for login. OIDC is built on top of OAuth 2.0. Currently, the LMS serves as the authentication provider.

To configure the E-Commerce service to work with OIDC, complete the following procedures.

- *Create and Register a Client*
- *Designate the Client as Trusted*

Create and Register a Client

To create and register a new OIDC client, follow these steps.

1. Start the LMS.
2. In your browser, go to `http://127.0.0.1:8000/admin/oauth2/client/`.
3. Select **Add client**.
4. Leave the **User** field blank.
5. For **Client Name**, enter `E-Commerce Service`.
6. For **URL**, enter `http://localhost:8002/`.
7. For **Redirect URL**, enter `http://127.0.0.1:8002/complete/edx-oidc/`. This is the OIDC client endpoint.

The system automatically generates values in the **Client ID** and **Client Secret** fields.

8. For **Client Type**, select **Confidential (Web applications)**.
9. Select **Save**.

Designate the Client as Trusted

After you create your client, designate it as trusted. Trusted clients bypass the user consent form that usually appears after the system validates the user's credentials.

To designate your client as trusted, follow these steps.

1. In your browser, go to `http://127.0.0.1:8000/admin/oauth2_provider/trustedclient/add/`.

2. In the **OAuth 2.0 clients** list, select the redirect URL for the client that you just created.
3. Select **Save**.

6.1.4 Configure a Site, Partner, and Site Configuration

To finish creating and configuring your OIDC client, you must configure a partner, site, and site configuration for the E-Commerce service to use. The site that you configure is the default site that the E-Commerce service adds when you run migrations. You must update this default site to match the domain that you will use to access the E-Commerce service. You must also set up a site configuration that contains an `oauth_settings` JSON field that stores your OIDC client's settings, as follows.

Setting	Description	Value
<code>SOCIAL_AUTH_EDX_OIDC_KEY</code>	OAuth 2.0 client key	The Client ID field in the <i>Create and Register a Client</i> section.
<code>SOCIAL_AUTH_EDX_OIDC_SECRET</code>	OAuth 2.0 client secret	The value from the Client Secret field in the <i>Create and Register a Client</i> section.
<code>SOCIAL_AUTH_EDX_OIDC_URL</code>	OAuth 2.0 authentication URL	For example, <code>http://127.0.0.1:8000/oauth2</code> .
<code>SOCIAL_AUTH_EDX_OIDC_ID_TOKEN_DECRYPTION_KEY</code>	OIDC ID token decryption key, used to validate the ID token	The same value as <code>SOCIAL_AUTH_EDX_OIDC_SECRET</code> .

To configure your default site, partner, and site configuration, use the appropriate settings module for your environment (`ecommerce.settings.devstack` for Devstack, `ecommerce.settings.production` for Fullstack) to run the following Django management command. This command updates the default site and creates a new partner and site configuration with the specified options.

```
$ sudo su ecommerce
$ python manage.py create_or_update_site --site-id=1 --site-domain=localhost:8002 --partner-code
```

Add Another Site, Partner, and Site Configuration

If you want to add more sites, partners, and site configurations, you can use the `create_or_update_site` command. The following options are available for this command.

Option	Required	Description	Example
<code>--site-id</code>	No	Database ID of a site you want to update.	<code>--site-id=1</code>
<code>--site-domain</code>	Yes	Domain by which you will access the E-Commerce service.	<code>--site-domain=ecommerce.example.com</code>
<code>--site-name</code>	No	Name of the E-Commerce site.	<code>--site-name='Example E-Commerce'</code>
<code>--partner-code</code>	Yes	Short code of the partner.	<code>--partner-code=edX</code>
<code>--partner-name</code>	No	Name of the partner.	<code>--partner-name='Open edX'</code>
<code>--lms-url-root</code>	Yes	URL root of the Open edX LMS instance.	<code>--lms-url-root=https://example.com</code>
<code>--theme-scss-path</code>	No	STATIC_ROOT relative path of the site's SCSS file.	<code>--theme-scss-path=sass/themes/edx.scss</code>
<code>--payment-processors</code>	No	Comma-delimited list of payment processors used on the site.	<code>--payment-processors=cybersource, paypal</code>
<code>--client-id</code>	Yes	OIDC client ID.	<code>--client-id=ecommerce-key</code>
<code>--client-secret</code>	Yes	OIDC client secret.	<code>--client-secret=ecommerce-secret</code>
<code>--from-email</code>	Yes	Address from which email messages are sent.	<code>--from-email=notifications@example.com</code>

To add another site, use the appropriate settings module for your environment (`ecommerce.settings.devstack` for Devstack, `ecommerce.settings.production` for Fullstack) to run the following Django management command. This command creates a new site, partner, and site configuration with the options that you specify.

```
$ sudo su ecommerce
$ python manage.py create_or_update_site --site-domain=[change me] --partner-code=[change me] --
```

6.1.5 Start the Server

To complete the installation and start the E-Commerce service, follow these steps.

Note: Local installations use SQLite by default. If you use another database backend, make sure you update your settings and create the database, if necessary, before you run migrations.

1. (Devstack only) If you are using devstack, switch to the `ecommerce` user and use the `ecommerce.settings.devstack` settings module to run the following commands.

```
$ sudo su ecommerce
$ make serve
```

2. To run the server, execute the following command.

```
$ python manage.py runserver 8002
```

Note: If you use a different port, make sure you update the OIDC client by using the Django administration panel in the LMS. For more information about configuring the OIDC client, see [Configure edX OpenID Connect \(OIDC\)](#).

6.1.6 Development Outside Devstack

If you are running the LMS in `devstack` but would prefer to run the E-Commerce service on your host, set up a reverse port-forward. This reverse port-forward allows the LMS process inside your devstack to use `127.0.0.1:8002` to make calls to the E-Commerce service running on your host. This simplifies LMS URL configuration.

To set up a reverse port-forward, execute the following command when you SSH into your devstack. Make sure that you run this command on the VM host, not the guest.

```
$ vagrant ssh -- -R 8002:127.0.0.1:8002
```

6.2 Manage Static Assets

After you *configure a partner and at least one site* for the E-Commerce system to use, you must compile all static assets and move them to the correct location to be served. The edX E-Commerce service uses `django-compressor` and `RequireJS` to manage static assets.

- `django-compressor` compiles and minifies CSS and JavaScript files, and names files to facilitate cache busting after new file deployment.
- `RequireJS` manages JavaScript dependencies.

Note: The static file directories are set up so that `make static` reads the build output directory of `r.js` before it checks for assets in the `ecommerce/static/` directory. EdX does not recommend that you run `make static` locally. If you run `make static` or `r.js` locally, make sure you delete the `ecommerce/static/build` folder or that you run `make static` before you continue with development. If you do not run `make static` again, `django-compressor` ignores all changes that you make to static files.

6.2.1 Compile and Move Static Assets

To compile and move your static assets and deploy your E-Commerce service, execute the following command locally or on another server.

```
$ make static
```

If you create new pages that have `RequireJS` dependencies, remember to add your new JavaScript modules to the `RequireJS` build file for the project. This is the `build.js` file.

6.3 Create E-Commerce Products

After you *configure a partner and at least one site* for the E-Commerce service to use, and you have compiled and moved your static assets, you can create products. For more information, see the following topics.

6.3.1 Creating Products Overview

The edX platform offers two types of products. You create both products in E-Commerce web pages that are similar to the Django administration panel.

- Course seats represent enrollment types. Each course seat has an associated set of attributes, such as price and certificate availability. The edX code uses course seats to determine how a given enrollment should be handled. For more information, see [Create Course Seats](#).
- Coupons allow users to offer learners a discount, either percentage or fixed amount, off a course enrollment. For more information, see [Create and Manage Coupons](#).

Prepare to Create a Product

Before you create a product, complete the following steps to start the E-Commerce service on your site.

1. In the ecommerce and LMS configuration files (`/edx/etc/ecommerce.yml` and `/edx/app/edxapp/lms.auth.json`, respectively), verify the following settings.

Note: If you are using `devstack`, these values are set correctly for you. However, edX recommends that you verify these values.

- The `EDX_API_KEY` value in the LMS file must be the same as the `EDX_API_KEY` value in the ecommerce file. If the values differ, change the value in the LMS file to match the ecommerce file.
 - The `ECOMMERCE_API_SIGNING_KEY` value in the LMS file must be the same as the `JWT_SECRET_KEY` value in the ecommerce file. If the values differ, change the value in the LMS file to match the ecommerce file.
2. On `devstack`, start the E-Commerce server on port 8002, and start the LMS on port 8000.

6.3.2 Create Course Seats

You create course seats by creating a course on the **Create New Course** page in the course administration tool, which is located at `http://localhost:8002/courses/`. After you create a course, the E-Commerce service creates the course seats that are associated with that course.

To create a course seat, follow these steps.

1. Follow the steps in [Creating Products Overview](#) to start your E-Commerce server.
2. In a browser on your E-Commerce server, go to `http://localhost:8002/courses` to access the **Courses** page.
3. On the **Courses** page, select **Add New Course**.
4. On the **Add New Course** page, enter the information for your course in the following fields.
 - **Course ID**
 - **Course Name**
5. For **Course Type**, select a course type and the options for that course type.
 - If you select **Free (Audit)**, you must specify whether you want to allow honor code learners to earn an honor code certificate. To do this, select **Yes** under **Include Honor Seat**.
 - If you select **Verified**, you must add the following information.
 - **Price (in USD)**
 - **Upgrade Deadline**
 - **Verification Deadline**

- **Include Honor Seat:** This option grants honor code certificates to learners who successfully complete the course.
- If you select **Professional Education**, you must add the following information.
 - **Price (in USD)**
 - **ID Verification Required?**
 - **Upgrade Deadline**
- If you select **Credit**, you must add the following information.
 - **Price (in USD):** The price for a verified certificate.
 - **Upgrade Deadline**
 - **Credit Provider**
 - **Price (USD):** The price for course credit.
 - **Credit Hours**
 - **Upgrade Deadline**
 - **Verification Deadline**
 - **Include Honor Seat:** This option grants honor code certificates to learners who successfully complete the course.

6. Select **Create Course**.

6.3.3 Create and Manage Coupons

This topic covers how to create and distribute coupons and their associated coupon codes. You can use coupons to provide discounted or free course enrollments, also called “course seats”, to your learners.

- *Create a Coupon*
- *Download Coupon Information*
- *Edit a Coupon*
- *Deactivate a Coupon*
- *Distribute Coupon Codes to Learners*
- *View the Invoice for a Coupon*

Create a Coupon

To create a coupon, you create one or more coupon codes that learners use to receive their discounts. You then use your email system to distribute the discount or enrollment codes for that coupon.

Creating a coupon code has several steps.

- *Enter Basic Information*
- *Specify the Coupon Code Type*
- *Specify Courses*
- *Specify Usage Limitations*
- *Specify Invoicing Options*
- *Finish and Review Coupon*

You create coupons and their associated discount or enrollment codes on the **Create New Coupon** page in the coupon administration tool, which is located at <http://localhost:8002/coupons/>. In the tool, you enter basic information and select the options for your coupon.

Enter Basic Information

Each coupon requires some basic information. To enter basic information for your coupon, follow these steps.

1. Follow the steps in [Creating Products Overview](#) to start your E-Commerce server.
2. In a browser on your E-Commerce server, go to <http://localhost:8002/coupons/> to open the coupon administration tool, and then select **Create Coupon**.
3. On the **Create New Coupon** page, enter the following information.
 - **Coupon Name:** The name you want to give the coupon, such as “January 15% Promotion”. The name must have fewer than 30 characters.
 - **Category:** A list of possible classifications for your coupon, such as “Course Promotion” and “Financial Assistance”. Categories can help you keep track of your coupons.
 - **Valid from** and **Valid until:** The dates and times when the discount or enrollment code is valid for use. The time zone is set to Universal Coordinated Time (UTC).
 - **Discount Value:** The discount that you want to apply to the course fee, specified as a percentage between 1% and 99% or a fixed amount in U.S. dollars.
 - **Client:** The name of the organization that you create the codes for. Note that the current system cannot automatically send this organization an invoice for the codes you create. For more information, see [Specify Invoicing Options](#) and [View the Invoice for a Coupon](#).
 - **Note** (optional): Any additional information that you want to add to your coupon, such as why the coupon was created. The note is visible on the coupon page in the coupon administration tool and in the .csv file for the coupon. It is not visible to learners.
 - **Tax Deducted Source (TDS):** This field is not yet active.

After you specify basic information for your coupon, you must specify the coupon code type.

Specify the Coupon Code Type

In addition to entering basic information, you must specify a coupon code type. The coupon code type is either “enrollment code” or “discount code”.

- An enrollment code covers the entire fee for a course seat.
- A discount code offers between 1% and 99% or a fixed dollar amount off the fee for a course seat.

A “course seat” is a single course enrollment in a specific course track.

To specify the coupon code type, follow these steps.

1. For **Code Type**, select **Enrollment Code** if you want the coupon code to cover the entire course fee, or select **Discount Code** if you want to provide a discount for the course.
2. If you selected **Enrollment Code**, locate the **Number of Codes** field, and then enter the number of enrollment codes that you want to create.

If you selected **Discount Code**, the following fields are visible.

- **Discount per Code** (required): Enter the percent or U.S. dollar amount of the discount that you want to offer, then select **Percent** or **Fixed**. Do not add a percent sign or a dollar sign.

- **Code** (optional): If you want to specify your own discount code, such as SCHOLAR15, enter the code that you want in this field.

If you want the system to generate a discount code for you, leave this field empty, and then enter the number of discount codes that you want to create in the **Number of Codes** field.

After you complete this step, you must specify the courses that you want your coupon to apply to.

Specify Courses

In addition to specifying your coupon's code type, you must specify the courses for your coupon. Your coupon can apply to a single course or to many courses.

To specify the courses for your coupon, follow these steps.

1. On the **Create New Coupon** page, select **Single Course** or **Multiple Courses**.
2. Specify the courses for your coupon.

To specify one course, follow these steps.

- (a) For **Course ID**, enter the ID of the course that you want. To find the course ID, open the course administration tool at `http://localhost:8002/courses`, select your course name in the list of courses, and then locate the **Course ID** field on the page for the course.
- (b) For **Seat Type**, select the type of seat for the coupon. A "seat" is a single course enrollment in a specific course track. For example, the seat type might be "verified" or "professional". The options for this field appear after you enter a valid value in the **Course ID** field.

To specify multiple courses, follow these steps.

- (a) In the **Valid for** field, enter a query to retrieve the courses that you want. For more information about creating a query, see [Create a Query for Multiple Courses](#).

To see a preview of the results that your query will return, enter your query in the **Valid for** field, and then select **Preview**. A dialog box opens that lists the courses in your query results.

3. For **Seat Types**, select **Verified**, **Professional**, or both. Note that you can only select both options if your coupon applies to multiple courses.

After you complete these steps, you must *specify usage limitations for your coupon*.

Create a Query for Multiple Courses The coupon administration tool uses queries to return a catalog of courses. The coupons that you create apply to each course in that catalog.

These queries use the following syntax.

```
field_name:search_terms
```

Your query can contain operators such as quotation marks ("), asterisks (*), and hyphens (-).

For example, your query might resemble one of the following queries.

```
org:(Company OR University)
org:(-Company OR -University)
start:[2016-01-01 TO 2016-12-31]
number:6.002x*
```

For more information about queries, including syntax and operators, see [Query string syntax](#).

The following table lists the field names that you can use in your query, along with a description for each field name.

Field Name	Description
announcement	The date the course is announced to the public, in YYYY-MM-DD format.
end	The course run end date, in YYYY-MM-DD format.
enrollment_end	The enrollment end date, in YYYY-MM-DD format.
enrollment_start	The enrollment start date, in YYYY-MM-DD format.
key	The course run key, sometimes also called the course ID.
language	The language in which the course is administered.
max_effort	The estimated maximum number of hours necessary to complete the course.
min_effort	The estimated minimum number of hours necessary to complete the course.
number	The course number (for example, 6.002x).
org	The organization associated with the course (for example, MITx).
pacing_type	The pacing for the course run. Options are <code>instructor_paced</code> or <code>self_paced</code> .
start	The course run start date, in YYYY-MM-DD format.
title	The course title.

Specify Usage Limitations

In addition to specifying courses for your coupon, you must specify usage limitations. Usage limitations control whether one or more customers can use the coupon, as well as the number of times each customer can use the coupon.

To specify usage limitations, follow these steps.

1. On the **Create New Coupon** page, locate **Usage Limitations**.
2. Select one of the following options.

- **Can be used once by one customer**

If you select this option, the **Number of Codes** field is visible. In this field, specify the number of individual discount or enrollment codes you want to create. The value must be 1 or greater. Make sure that you create enough discount or enrollment codes so that each person receives one code.

- **Can be used once by multiple customers** or **Can be used multiple times by multiple customers**

If you select one of these options, the **Maximum Number of Usages** field is visible. In this field, specify the number of times customers can use the coupon code.

For example, if you want to create a coupon code that is available for 10 different customers, and each customer can use the code only one time, enter **10** for **Can be used once by multiple customers**.

If you want to create a coupon code that is available for 10 uses, whether by one customer or multiple customers, enter **10** for **Can be used multiple times by multiple customers**.

After you specify usage limitations, you must specify invoicing options for your coupon.

Specify Invoicing Options

In addition to setting usage limitations for your coupon, you must specify invoicing options. You can send an invoice when you create the coupon or after one or more customers have redeemed the coupon. The invoice can be for the total amount or for part of the total amount.

To specify the way you want to invoice your client, follow these steps.

1. On the **Create New Coupon** page, locate **Invoice Type**.

2. Select one of the following options.

- **Already invoiced:** Select this option if you have already sent an invoice to your client.

If you select this option, you must also enter information in the following fields.

- **Invoice Number:** Your internal invoice number.
- **Invoice Amount:** The amount that you have already invoiced the client.
- **Payment Date:** The date when payment is due from the client.

- **Invoice after redemption:** Select this option if you will send an invoice to your client after one or more coupon codes have been redeemed.
- **N/A:** Select this option if neither of the other options applies to your situation.

3. For **Total to Invoice to Client**, enter the amount to invoice the client for this coupon. You can enter the total amount or a different amount. If you do not want to send an invoice to the client, you can leave this field blank.

Finish and Review Coupon

- After you have *entered all the basic coupon information* and specified the options that you want, select **Create Coupon**.

When you select **Create Coupon**, the system generates one or more discount or enrollment codes as well as the URLs where users can redeem these codes.

When the system has finished generating the coupon, a page for the coupon opens. This page lists the information for your coupon, including all discount or enrollment codes for the coupon, the coupon's status, URLs where users can redeem the codes, dates the coupon is valid, and the course that the coupon applies to. You can also *download a .csv file with the coupon information* from this page.

Download Coupon Information

After you create a coupon, you can download a .csv file that lists the information that you entered when you created the coupon. The .csv file also includes additional information, such as the discount or enrollment codes that are associated with your coupon and the system-generated URL where a user can redeem each code.

1. In your browser, go to `http://localhost:8002/coupons/` to open the coupon administration tool.
2. On the **Coupon Codes** page, locate the coupon that you want in the table, and then select the name of the coupon. The page for the coupon opens.
3. On the page for the coupon, select **Download**. Your .csv file begins downloading automatically.

The .csv file for your coupon lists the following information.

Coupon Name	The name of the coupon.
Code	The 16-digit alphanumeric discount or enrollment code. The .csv file lists at least one entry for each code. The system creates an additional entry for the code each time the status of the code changes (for example, when the status changes from Active to Redeemed).
Maximum Coupon Usage	The number of times that the discount code or enrollment code that is associated with the coupon can be used. For single-use coupons, this value is 1. For multi-use coupons, this is the value that you specified in the Maximum Number of Usages field.
Redemption Count	The number of times the coupon has been redeemed. The initial value is 0, and the value is incremented each time that a discount code or enrollment code for the coupon is redeemed.
Coupon Type	The type of coupon code associated with this coupon. Possible values are Enrollment and Discount .
URL	The URL where the user redeems the coupon code.
Course ID	The ID of the course that the coupon applies to.
Organization	The organization that provides the course.
Client	The organization that purchased the coupon codes.
Category	The value that you selected for the Category field when you created the coupon.
Note	The text, if any, that you entered in the Note field when you created the coupon.
Price	The regular fee for the course.
Invoiced Amount	The text, if any, that you entered in the Total to Invoice to Client field when you created the coupon.
Discount Percentage	The percent discount, if any, that you specified when you created the coupon.
Discount Amount	The dollar amount discount, if any, that you specified when you created the coupon.
Status	The status of the coupon. Possible values are Active , Redeemed , or Expired .
Order Number	The order number associated with the redemption of the enrollment or discount code.
Redeemed By Username	The username of the customer who redeemed the enrollment or discount code.
Created By	The username of the user who created the coupon.
Create Date	The date the coupon was created.
Coupon Start Date	The first date the coupon can be used.
Coupon Expiry Date	The last date the coupon can be used.

Edit a Coupon

You edit a coupon by using the coupon administration tool.

Note: You cannot edit the following fields.

- **Code Type**
- **Course ID**
- **Single course** or **Multiple courses**
- **Seat Type**
- **Usage Limitations**
- **Number of Codes** or **Maximum Number of Usages**

1. In your browser, go to `http://localhost:8002/coupons/` to open the coupon administration tool.
2. On the **Coupon Codes** page, locate the coupon that you want in the table, and then select the name of the coupon. The page for the coupon opens.
3. On the page for the coupon, select **Edit Coupon**.
4. Make the changes that you want.
5. Select **Save Changes**.

Deactivate a Coupon

To deactivate a coupon, change the **Valid from** and **Valid until** date fields so that both dates are in the past. For more information, see [Edit a Coupon](#).

Distribute Coupon Codes to Learners

You can distribute both discount codes and enrollment codes to learners in two ways.

- You provide a coupon code that learners enter on the **Checkout** page for the verified or professional certificate track. If you do this, edX recommends that you also provide the URL for the course About page to make signing up for the course easier.
- You provide a URL for an offer landing page. This automatically generated page presents information about the course, lets the learner know that the coupon code has been applied, and provides the opportunity for the learner to enroll. Learners can access this URL if they do not have an edX account or they are not signed in. However, learners must sign in or create an edX account to redeem the coupon and enroll in the course.

A URL for an offer landing page has the following format.

```
http://localhost:8002/coupons/offer/?code=#####
```

Note: If the coupon code is a discount code, the learner must pay any balance due before enrolling in the course for a verified or professional certificate.

To distribute the coupon code or URL to learners, you determine the coupon code or the URL for the learner to use, and then you create and send an email that includes the coupon code or the URL. For suggestions for email message text, see [Example Email Messages](#).

Find a Coupon Code or URL

You can find coupon codes, whether discount codes or enrollment codes, and their associated URLs in two places: on the page for the coupon in the coupon administration tool, and in a downloadable .csv file. You can use either option to find the coupon code or URL for your learners.

Find a Code or URL on the Coupon Page To find a coupon code or URL on the page for the coupon in the coupon administration tool, follow these steps.

1. In your browser, go to `http://localhost:8002/coupons/` to open the coupon administration tool.
2. On the **Coupon Codes** page, locate the coupon that you want in the table, and then select the name of the coupon. The page for the coupon opens.
3. On the page for the coupon, locate the table under **Codes**.

4. In the table, locate the information that you want.

- For a coupon code that the learner will enter on the **Checkout** page, use the value in the **Code** column.
- For an offer landing page, use the URL in the **Redemption URL** column.

Find a Code or URL in a Downloaded File To find a coupon code or URL in the .csv file for a coupon, follow these steps.

1. *Download a .csv file* that lists the information for your coupon, and then open the .csv file.
2. In the .csv file, locate the information that you want.
 - For a coupon code that the learner will enter on the **Checkout** page, use the value in the **Code** column.
 - For an offer landing page, use the URL in the **URL** column.

Send an Email Message

After you *locate the coupon code or URL* that you want to use, you use your email system to provide that information in a message to potential learners.

Note: When you send a coupon code for a learner to use on the **Checkout** page, edX recommends that you include the About page URL for the course as well as the coupon code to help the learner enroll more easily.

Example Email Messages You can use the following email messages as examples of the communication that you send to your learners.

If Learners Enter a Coupon Code on the Checkout Page

```
Dear learner,

This message includes a discount <or an enrollment> code for edX101: Overview
of Creating an edX Course. For more information about the course, see
https://www.edx.org/course/overview-creating-edx-course-edx-edx101.

To redeem this code, sign up for a verified <or professional> certificate, and
then enter the following coupon code in the **Coupon Code** field on the
**Checkout** page:

ZDPC3AQV3732RQT5

We look forward to learning with you!

The edX101 course team
```

If Learners Visit an Offer Landing Page

```
Dear learner,

This message includes a discount <or an enrollment> code for edX101: Overview
of Creating an edX Course. To redeem this code and enroll in the course, visit
the following URL:

http://localhost:8002/coupons/offer/?code=ZDPC3AQV3732RQT5
```

We look forward to learning with you!

The edX101 course team

View the Invoice for a Coupon

When you create a coupon, the E-Commerce service generates and fulfills an order. The Invoice Payment Processor module in the service then records the fulfilled order. Because the Invoice Payment Processor module assumes that you have sent or will send an invoice to the client who purchased the coupon, the module also records the order in the Invoice table in the Django administration panel so that you can view and reconcile the order.

To view your coupon invoices in the Invoice table, go to `http://localhost:8002/admin/invoice/invoice/`. The table lists all of the invoices for your coupons, along with information such as the client name and the invoice status.

For more information about the way that the E-Commerce service manages orders, see [Manage Orders](#).

6.4 Manage Orders

EdX has created a framework that manages order placement and fulfillment for digital products. Most of the products that edX supports involve modifications to enrollments.

The edX framework allows modules that fulfill enrollment-related products to interact with the [edX Enrollment API](#), which is a part of the LMS. This process can be either synchronous or asynchronous.

6.4.1 Place an Order

To place an order, you need the following information.

- Order number
- User
- Basket
- Shipping method
- Shipping charge
- Billing address
- Order total

To place an order, use the `handle_order_placement()` method that `EdxOrderPlacementMixin` provides. If you modify this code, make sure that the code collects payment before it creates order objects.

6.4.2 Fulfill Orders

To fulfill orders, emit a `post_checkout` signal. An internal fulfillment API then delegates fulfillment of individual order items to the appropriate fulfillment modules.

6.4.3 About Fulfillment Modules

The E-Commerce base fulfillment module has the following interface.

```
fulfill_product(product)

revoke_line(line)
```

Every `ProductType` has a corresponding module that extends this interface and fulfills order items of that particular `ProductType`. To fulfill an order, the system gives each fulfillment module configured in settings (`_oscar.py`) an opportunity to fulfill order lines.

- The `fulfill_product` method fulfills the order. For example, `fulfill_product` might enroll a learner in a course or upgrade the learner to a verified certificate).
- The `revoke_line` method revokes a specific order line. For example, `revoke_line` might unenroll learners from courses or downgrade a learner from a verified seat.

6.4.4 Recover from a Fulfillment Error

If a fulfillment operation fails, the E-Commerce service assigns the order a status indicating the reason for the failure. If you have enabled asynchronous order fulfillment, the service tries again to fulfill the order. You can also manually retry fulfillment of a failed order from the [Oscar](#) order dashboard.

6.5 Test Features

When you create a new feature for the Open edX platform, you must write two kinds of tests for that feature: general tests that evaluate the feature on the Open edX platform, and tests that are specific to the type of feature that you create. For example, if you create a coupon codes feature for use with the edX Ecommerce service, you must write both Open edX tests and Ecommerce tests. This section describes the general tests that you must write for the Open edX platform.

6.5.1 Tests for the Open edX Platform

You must write the following types of tests for all new features that you create for the Open edX platform.

- Django tests. To learn how to test Django code, see the [Testing in Django](#) documentation provided by the Django Software Foundation.
- Acceptance tests. These tests verify behavior that relies on external systems, such as the LMS or payment processors. At a minimum, you must run these tests against a staging environment before you deploy code to production to verify that critical user workflows are functioning as expected. With the right configuration, you can also run the tests locally.

6.6 Test Your E-Commerce Application

To test new applications that you develop for the E-Commerce service, you create and run tests for the Open edX platform first, and then you run a set of tests that are specific to E-Commerce.

For more information about tests for the Open edX platform, see [Test Features](#).

6.6.1 Tests for E-Commerce

When you develop E-Commerce applications, you must run a pre-packaged set of unit tests, Python tests, and E-Commerce acceptance tests.

Run E-Commerce Unit Tests

The E-Commerce unit tests include migrations, the unit test suite, and quality checks. You can run the full unit test, or save time for a local test by disabling the migrations. (You can also run the quality checks independent of the unit test suite.)

- To run the full unit test, including migrations and quality checks, use the following command.

```
$ make validate
```

- To run unit tests with quality checks but without migrations, run the following command.

```
$ DISABLE_MIGRATIONS=1 make validate
```

Note: We recommend that you only run tests without migrations when you run the tests locally.

- To validate code quality independently, run the following command.

```
$ make quality
```

Python Unit Tests

When you create E-Commerce tests, use the `TestCase` class in `ecommerce/tests/testcases.py` to ensure every test has `Site` and `Partner` objects configured. This will help you test any code that relies on these models, which are used for multi-tenancy.

- To run all Python unit tests and quality checks, run the following command.

```
$ make validate_python
```

- To run all Python unit tests and quality checks *in parallel*, run the following command.

```
$ make fast_validate_python
```

- To run the Python unit tests in a specific file, such as `ecommerce/courses/tests/test_utils.py`, run the following command and substitute the desired file path.

```
$ DISABLE_ACCEPTANCE_TESTS=True ./manage.py test
ecommerce.courses.tests.test_utils --settings=ecommerce.settings.test
--with-ignore-docstrings --logging-level=DEBUG
```

Setting the `DISABLE_MIGRATIONS` variable significantly decreases the time needed to run tests by creating the test database directly from Django model definitions as opposed to applying the defined migrations.

```
$ DISABLE_MIGRATIONS=1 DISABLE_ACCEPTANCE_TESTS=True ./manage.py test
ecommerce.courses.tests.test_utils --settings=ecommerce.settings.test
--with-ignore-docstrings --logging-level=DEBUG
```

JavaScript Unit Tests

The E-Commerce project uses [Jasmine](#) for JavaScript unit testing. To create these tests, place your tests in the `ecommerce/static/js/test/specs` directory, and add a `_spec` suffix. For example, your test name may be `ecommerce/static/js/test/specs/course_list_view_spec.js`.

All JavaScript code must adhere to the [edX JavaScript standards](#). These standards are enforced using [JSHint](#) and [jscs](#).

- To run all JavaScript unit tests and linting checks, run the following command.

```
$ make validate_js
```

Run E-Commerce Acceptance Tests

To run specific acceptance tests for the E-Commerce service, you must complete the following procedures.

- *Configure the LMS*
- *Configure E-Commerce*
- *Configure Acceptance Tests*
- *Run Acceptance Tests*

Configure the LMS

To configure the LMS, follow these steps.

1. Verify that the following settings in `lms.env.json` are correct.

```
"ECOMMERCE_API_URL": "http://localhost:8002/api/v2/"
"ECOMMERCE_PUBLIC_URL_ROOT": "http://localhost:8002/"
"JWT_ISSUER": "http://127.0.0.1:8000/oauth2" // Must match the E-Commerce JWT_ISSUER setting
"OAUTH_ENFORCE_SECURE": false
"OAUTH_OIDC_ISSUER": "http://127.0.0.1:8000/oauth2"
```

2. Verify that the following settings in `lms.auth.json` are correct.

```
"EDX_API_KEY": "replace-me" // Must match the E-Commerce EDX_API_KEY setting
"ECOMMERCE_API_SIGNING_KEY": "insecure-secret-key" // Must match the E-Commerce JWT_SECRET_KEY s
```

3. Verify that an LMS account with staff and superuser permissions exists.

By default, most devstack and fullstack LMS instances include a user account that has staff permissions. This account has the username `staff`, the email address `staff@example.com`, and the password `edx`. Run the following commands to grant the account superuser permissions.

```
$ ``./manage.py lms shell --settings=devstack``
>>> from django.contrib.auth.models import User
>>> u = User.objects.get(username='staff')
>>> u.is_superuser = True
>>> u.save()
```

4. In the Django administration panel, verify that an OAuth2 client with the following attributes exists. If one does not already exist, *create a new one*. The client ID and secret must match the values of the E-Commerce `SOCIAL_AUTH_EDX_OIDC_KEY` and `SOCIAL_AUTH_EDX_OIDC_SECRET` settings, respectively.


```
URL: http://localhost:8002/
Redirect URI: http://localhost:8002/complete/edx-oidc/
Client ID: 'replace-me'
Client Secret: 'replace-me'
Client Type: Confidential
```

5. In the Django administration panel, verify that the OAuth2 client referred to above is designated as a trusted client. If this isn't already the case, add the client created above as a new trusted client.
6. In the Django administration panel, create a new access token for the superuser account. Set the client to the OAuth2 client referred to above. Make note of this token; it is required to run the acceptance tests.
7. Make sure that the LMS instance that you will use for testing has at least two courses that learners could enroll in. By default, most LMS instances include the edX demonstration course. Use Studio to create a second course.

Configure E-Commerce

You use the CAT to finish configuring the two courses in your LMS instance.

1. In your browser, go to `http://localhost:8002/courses/` to access the CAT.
2. In the CAT, add both of the courses present on your LMS instance to E-Commerce. Configure one as a “Free (Audit)” course, and the second as a “Verified” course.
3. So that you can test integration with external payment processors, update the contents of the `PAYMENT_PROCESSOR_CONFIG` dictionary found in the settings with valid credentials. To override the default values for development, create a private settings module, `private.py`, and set `PAYMENT_PROCESSOR_CONFIG` inside the module.

Note: If you created a `private.py` file to create settings overrides when you *set up your virtual environment*, you can use that same `private.py` file.

Configure Acceptance Tests

You configure acceptance tests by using the settings in the `ecommerce/blob/master/acceptance_tests/config.py` file. You can use the default values for most settings in this file. However, you must specify values for the following settings by using environment variables.

Variable	Description
<code>ACCESS_TOKEN</code>	The OAuth2 access token used to authenticate requests.
<code>ECOMMERCE_URL_ROOT</code>	The URL root for the E-Commerce service.
<code>LMS_URL_ROOT</code>	The URL root for the LMS.
<code>LMS_USERNAME</code>	A username for any current LMS user, to use during testing.
<code>LMS_EMAIL</code>	The email address used to sign in to the LMS.
<code>LMS_PASSWORD</code>	The password used to sign in to the LMS.

If you test PayPal integration, you must also specify values for the following settings by using environment variables.

Variable	Description
<code>PAYPAL_EMAIL</code>	The email address used to sign in to PayPal during payment.
<code>PAYPAL_PASSWORD</code>	The password used to sign in to PayPal during payment.

Run Acceptance Tests

Run all acceptance tests by executing `make accept`. To run a specific test, execute the following command.

```
$ nosetests -v <path/to/the/test/module>
```

The acceptance tests rely on the *environment variables that you have configured*. For example, when you run the acceptance tests against local instances of E-Commerce and the LMS, you might run the following command, replacing values between angle brackets (<>) with your own values.

```
$ ECOMMERCE_URL_ROOT="http://localhost:8002" LMS_URL_ROOT="http://127.0.0.1:8000" LMS_USERNAME="<username>"
```

When you run the acceptance tests against a production-like staging environment, you might run the following command.

```
$ ECOMMERCE_URL_ROOT="https://ecommerce.stage.edx.org" LMS_URL_ROOT="https://courses.stage.edx.org" LMS_USERNAME="<username>"
```

6.7 Additional E-Commerce Features

After you install the basic features of the E-Commerce service on your instance of the Open edX platform, you can enable or install additional features. You can set up E-Commerce to send email, switch payment processors, or track event data if one of your usual processors is unavailable.

6.7.1 Sending Notifications

The edX E-Commerce service uses the [Communications API](#) that is part of [Oscar](#) to create and send notifications in the form of email messages. To send notifications, you must set up notifications, create one or more email messages, and then send the email messages.

Set Up Notifications

1. Enable the E-Commerce service to send notifications. To do this, change the value of the `ENABLE_NOTIFICATIONS` feature flag to `True`.
2. Define communication type codes to refer to particular types of notification. For example, you might define a communication type code named `COURSE_SEAT_PURCHASED` to correspond to the purchase of a course seat.

Create an Email Message

The E-Commerce service can send both HTML and plain text email messages. To create an email message, create the following three files in the `ecommerce/ecommerce/templates/customer/emails/` folder.

- An HTML template that extends `email_base.html` and includes the body of the email.
- A plain text file that contains the email's subject line.
- A plain text file that contains the body of the email.

Use the following convention to name these files.

```
commtype_{communication type code}_body.html
```

For example, if the communication type code is `course_seat_purchased`, the three files would have the following names.

- `commtype_course_seat_purchased_body.html`
- `commtype_course_seat_purchased_body.txt`
- `commtype_course_seat_purchased_subject.txt`

Note: To add a custom email body, override `block body` in the `email_base.html` file. To add a custom footer, override `block footer` in the `email_base.html` file.

Send Email Messages

To send email messages, use the `send_notification(user, commtype_code, context)` method. This method is implemented in `ecommerce/ecommerce/notifications/notifications.py`.

6.7.2 Changing Payment Processors

Payment processors sometimes experience temporary outages. When these outages occur, you can use Waffle switches to disable the faulty payment processor or processors, then re-enable them after the outage is over.

The names of these switches use prefixes that are the value of the `PAYMENT_PROCESSOR_SWITCH_PREFIX` setting. By default, this value is `payment_processor_active_`. The following table lists valid switches and the payment processors they control.

Payment Processor	Switch Name	Default Value
PayPal	<code>payment_processor_active_paypal</code>	True
CyberSource	<code>payment_processor_active_cybersource</code>	True

In the unlikely event that all payment processors are disabled, the LMS will display an informative error message explaining why payment is not currently possible.

6.7.3 Tracking Data

The E-Commerce service uses [Segment](#) to collect business intelligence data.

To emit events to your Segment project, specify your Segment project's API key as the value of the `SEGMENT_KEY` setting.

6.7.4 Gating E-Commerce Service Features

You can release new E-Commerce service features and functionality behind a feature gate. This project uses the [Waffle](#) library for feature gating.

Types of Feature Gates

Waffle supports the following types of feature gates.

- **Flag:** This gate allows you to enable a feature for specific users, groups, users who meet certain criteria (such as authenticated users or staff), or a certain percentage of visitors.
- **Switch:** This gate is a Boolean that turns a feature on or off for all users.
- **Sample:** This gate allows you to define the probability with which a given feature will be on.

For more information about creating or updating features and feature gates, see the [Waffle documentation](#).

Feature Gates

Waffle offers the following feature gates.

Name	Type	Purpose
user_enrollments_on_dashboard	Switch	Display a user's current enrollments on the dashboard user detail page.
publish_course_modes_to_lms	Switch	Publish prices and SKUs to the LMS after every course modification.
async_order_fulfillment	Sample	Specify what percentage of orders are fulfilled asynchronously.
ENABLE_CREDIT_APP	Switch	Enable the credit checkout page, from which learners can purchase credit in a course.
ENABLE_NOTIFICATIONS	Switch	Enable email notifications for a variety of user actions, such as when an order is placed.
PAYPAL_RETRY_ATTEMPTS	Switch	Enable users to retry unsuccessful PayPal payments.

Enable a Feature Permanently

If you want to make a feature permanent, remove its feature gate from relevant code and tests, and delete the gate from the database.

6.7.5 Maintaining the E-Commerce Service

Most of the time, you do not have to perform maintenance on the E-Commerce service. However, E-Commerce creates **basket** objects to track products that users want to purchase before users place an order. As more baskets and orders are created, the baskets table can grow large. Depending on your database backend, a large table can become difficult to manage and migrate. After an order is placed, you can delete the corresponding basket from the baskets table.

To delete one or more baskets, follow these steps.

Note: Baskets that contain products but that are not used to create orders, such as when a user adds a product to a basket but does not complete the order process, are not deleted. These baskets provide records that users intended to purchase a product.

1. To display the number of baskets that you can delete, run the following command.

```
$ ./manage.py delete_ordered_baskets
```

2. To delete all the baskets that appear after you run the command in step 1, use the `--commit` option.

```
$ ./manage.py delete_ordered_baskets --commit
```

To complete the procedures that this section describes, you use both the Django administration site and the Course Administration Tool (CAT). The CAT is a web app that is included with the E-Commerce service. The CAT enables you to configure and manage products that are associated with the courses on your instance of the Open edX learning management system (LMS).

In addition to these required steps, you can add optional features to the E-Commerce service for your instance of the Open edX platform. For more information, see [Additional E-Commerce Features](#).

Setting Up the Open edX Mobile Applications

This section is intended for those who are building Open edX mobile applications and customizing an Open edX installation to support their use.

- *Accessing the Source Code*
- *Configuring Mobile Application Features*
- *Configuring Video Modules for Mobile*
- *Enabling Push Notifications*

7.1 Accessing the Source Code

There are currently two edX mobile applications, one for iOS and one for Android. You can find the source code and additional documentation for each application in the following repositories.

- iOS: <http://github.com/edx/edx-app-ios>
- Android: <http://github.com/edx/edx-app-android>

7.2 Configuring Mobile Application Features

For the mobile API and authentication to work correctly with the edX mobile applications, you enable them in the `edx/app/edxapp/lms.env.json` file. You then complete the setup for authentication by creating OAuth clients and adding their IDs to the custom configuration file of each mobile application.

7.2.1 Enable Mobile Application Features

Note: Configuration settings added to the `lms.env.json` file are reset to their default values when you use Ansible to update `edx-platform`.

To enable the mobile application features, follow these steps.

1. In the `edx/app/edxapp/lms.env.json` file, add the following lines to the features section.

```
"FEATURES" : {  
  ...  
  "ENABLE_MOBILE_REST_API": true,  
  "ENABLE_OAUTH2_PROVIDER": true,  
  "ENABLE_COMBINED_LOGIN_REGISTRATION": true  
}
```

If you are running in a non-SSL environment, you can set `"OAUTH_ENFORCE_SECURE": false`. This configuration setting should never be set to `false` in anything other than a development environment.

1. Save the `edx/app/edxapp/lms.env.json` file.
2. Restart the server.

7.2.2 Create the OAuth Clients

You create an OAuth client ID and secret that are specific to your installation for use by the mobile applications. The procedure that follows assumes that you create a different client for each of the edX mobile applications.

Before you can create OAuth clients, a superuser must exist on the server. For more information, see [edX Managing the Full Stack](#).

To create your OAuth clients, follow these steps.

1. Log in to the Django administration console for your base URL. For example, `http://{your_URL}/admin`.
2. In the **Oauth2** section, select **Clients**.
3. Select **Add client**. A dialog box opens with the **Client id** and **Client secret** populated for the new client.
4. Enter a **Url** and **Redirect Url** for the first mobile application. For example, `https://{your_URL}/api/mobile/{version}/?app=ios`. While the console requires values for these fields, they are not used by the edX mobile applications. You can enter the same value in both fields.
5. For the **Client type**, select **Public**.
6. Optionally, enter a **User** and an identifying **Name**.
7. Select **Save and add another**.
8. Repeat steps 4-6 for the second mobile application, and then select **Save**.

7.2.3 Configure the Applications with OAuth Client IDs

The procedure that follows assumes that you have a different OAuth client ID for each edX mobile application.

To configure each edX mobile application with its OAuth client ID, you add a setting to its custom configuration `.yaml` file. For information about how to set up custom configuration directories and files, see the GitHub repositories for [iOS](#) and [Android](#).

1. Obtain the OAuth client ID for the first mobile application from your Django administration console. For more information, see [Create the OAuth Clients](#).
2. In your custom GitHub configuration directory, edit the `.yaml` file for the first mobile application. For example, for the edX mobile application for iOS, edit your `ios.yaml` file.
3. Add the following line to the `.yaml` file.

```
OAUTH_CLIENT_ID: '{client_id_for_ios_app}'
```

4. Save the file.
5. Repeat steps 1-4 for the second mobile application.

7.2.4 Configure OAuth Token Expiration

When OAuth tokens expire, learners who use the mobile apps to access your site must log in again.

The `lms/envs/common.py` file includes the default configuration settings for the number of days before OAuth tokens expire for confidential clients and public clients. Instead of modifying the defaults in `lms/envs/common.py`, add the configuration settings to the `lms.env.json` file and set values that will override the default settings.

To configure the number of days before OAuth tokens expire, follow these steps.

1. In the `edx/app/edxapp/lms.env.json` file, add the following lines to specify the number of days that OAuth tokens remain valid for confidential or public clients.

Note: Make sure you add these lines at the top level of the JSON dictionary, and not inside any other declarations.

```
"OAUTH_EXPIRE_CONFIDENTIAL_CLIENT_DAYS" : 365
"OAUTH_EXPIRE_PUBLIC_CLIENT_DAYS" : 30
```

2. Save the `lms.env.json` file, then restart the `edxapp` app.

The values that you defined in `lms.env.json` override the default values defined in `lms/envs/common.py`.

7.3 Configuring Video Modules for Mobile

Course videos must be specifically prepared to ensure that they are in mobile accessible formats. Video modules in mobile-available courses should have low resolution encodings that are readily accessible by mobile devices.

After you obtain a low resolution encoding for a video file and post it online, your course team can add its location to the video module that includes the video in the course structure.

- To configure a video module in edX Studio, you edit the video component. On the **Advanced** tab, in the **Video File URLs** field, enter the URL to the mobile-targeted video as the first URL in the list. For more information, see [Working with Video Components](#) in *Building and Running an Open edX Course*.
- To configure a video module by editing the course XML, enter the URL to the mobile-targeted video as the first URL in the list of `html5_sources`. For more information, see [Video Components](#) in the *edX Open Learning XML Guide*.

7.4 Enabling Push Notifications

You enable push notifications on the server and then configure each of the edX mobile applications. Currently, you can use Parse for notification delivery.

The procedures that follow assume that you have obtained an application ID, REST API key, and client key from Parse.

7.4.1 Enable Push Notification on the Server

To enable the push notification feature, follow these steps.

1. In the `edx/app/edxapp/cms.auth.json` file, add the following lines.

```
PARSE_KEYS = {  
  
    "APPLICATION_ID": "{app_id}",  
  
    "REST_API_KEY": "{API_key}"  
  
}
```

2. Save the `edx/app/edxapp/cms.auth.json` file.
3. Restart the server.
4. Log in to the Django administration console for your Studio URL. For example, `http://studio.{your_URL}/admin`.
5. In the **Contentstore** section, select **Push notification configs**.
6. Select **Add push notification config**.
7. Verify that **Enabled** is selected, and then select **Save**.

7.4.2 Configure Push Notification on the Clients

The procedure that follows assumes that you use a single set of Parse credentials for both of the edX mobile applications.

You add push notification settings to the applicable custom configuration `.yaml` file. For information about how to set up custom configuration directories and files, see the GitHub repositories for [iOS](#) and [Android](#).

1. In your custom GitHub configuration directory, edit the `.yaml` file that stores configuration settings that are common to both of the edX mobile applications. For example, edit your `shared.yaml` file.
2. Add the following lines to the `.yaml` file.

```
PARSE:  
  NOTIFICATIONS_ENABLED: true  
  APPLICATION_ID: {your application id}  
  CLIENT_KEY: {your client key}  
  
PUSH_NOTIFICATIONS: true
```

3. Save the file.

Index of EdX Feature Flags

The following list includes most feature flags that are available in the edX platform.

Name	Description	Default Value
ACCESS_REQUIRE_STAFF_FOR_COURSE	Supported	FALSE
ADVANCED_SECURITY	Supported	TRUE
ALLOW_COURSE_STAFF_GRADE_DOWNLOADS	Supported	FALSE
AUDIT_CERT_CUTOFF_DATE	Supported	None
AUTH_USE_CAS	Supported	TRUE
AUTH_USE_CERTIFICATES	Supported	FALSE
AUTH_USE_OPENID	Supported	FALSE
AUTH_USE_OPENID_PROVIDER	Supported	FALSE
AUTH_USE_SHIB	Supported	FALSE
AUTOMATIC_AUTH_FOR_TESTING	Development	FALSE
AUTOMATIC_VERIFY_STUDENT_IDENTITY_FOR_TESTING	Development	FALSE
CERTIFICATES_HTML_VIEW	Supported	FALSE
CERTIFICATES_INSTRUCTOR_GENERATION	Supported	FALSE
CUSTOM_COURSES_EDX	Supported	FALSE
DEBUG_LEVEL	Deprecated	0
DISABLE_LOGIN_BUTTON	Supported	FALSE
DISPLAY_ANALYTICS_DEMOGRAPHICS	Deprecated	TRUE
DISPLAY_ANALYTICS_ENROLLMENTS	Deprecated	TRUE
DISPLAY_DEBUG_INFO_TO_STAFF	Supported	TRUE
EMBARGO	Supported	FALSE
ENABLE_CORS_HEADERS	Supported	FALSE
ENABLE_COSMETIC_DISPLAY_PRICE	Supported	FALSE
ENABLE_COURSE_DISCOVERY	Supported	FALSE
ENABLE_COURSEWARE_SEARCH	Supported	FALSE
ENABLE_CREDIT_API	Supported	FALSE
ENABLE_DASHBOARD_SEARCH	Supported	FALSE
ENABLE_DISABLING_XBLOCK_TYPES	Supported	TRUE
ENABLE_DISCUSSION_EMAIL_DIGEST	Supported	FALSE
ENABLE_DISCUSSION_HOME_PANEL	Supported	FALSE
ENABLE_DISCUSSION_SERVICE	Supported	TRUE
ENABLE_DJANGO_ADMIN_SITE	Supported	TRUE
ENABLE_EDXNOTES	Supported	FALSE
ENABLE_FEEDBACK_SUBMISSION	Supported	FALSE
ENABLE_INSTRUCTOR_ANALYTICS	Deprecated	FALSE

Continued on next page

Table 8.1 – continued from previous page

Name	Description	Default Value
ENABLE_INSTRUCTOR_BACKGROUND_TASKS	Supported	TRUE
ENABLE_INSTRUCTOR_EMAIL	Supported	TRUE
ENABLE_INSTRUCTOR_LEGACY_DASHBOARD	Deprecated	FALSE
ENABLE_LMS_MIGRATION	Deprecated	FALSE
ENABLE_MANUAL_GIT_RELOAD	Supported	FALSE
ENABLE_MAX_FAILED_LOGIN_ATTEMPTS	Supported	TRUE
ENABLE_MAX_SCORE_CACHE	Supported	TRUE
ENABLE_MKTG_SITE	Supported	FALSE
ENABLE_MOBILE_REST_API	Supported	FALSE
ENABLE_OAUTH2_PROVIDER	Supported	FALSE
ENABLE_ONLOAD_BEACON	Supported	FALSE
ENABLE_OPENBADGES	Supported	FALSE
ENABLE_PAID_COURSE_REGISTRATION	Supported	FALSE
ENABLE_PREREQUISITE_COURSES	Supported	FALSE
ENABLE_PSYCHOMETRICS	Deprecated	FALSE
ENABLE_RENDER_XBLOCK_API	Removed	FALSE
ENABLE_S3_GRADE_DOWNLOADS	Supported	FALSE
ENABLE_SHOPPING_CART	Supported	FALSE
ENABLE_SPECIAL_EXAMS	Supported	False
ENABLE_STUDENT_HISTORY_VIEW	Supported	TRUE
ENABLE_STUDENT_NOTES	Supported	TRUE
ENABLE_TEAMS	Supported	FALSE
ENABLE_TEXTBOOK	Supported	TRUE
ENABLE_THIRD_PARTY_AUTH	Supported	FALSE
ENABLE_VIDEO_BEACON	Supported	FALSE
ENABLE_VIDEO BUMPER	Supported	FALSE
ENABLE_XBLOCK_VIEW_ENDPOINT	Supported	FALSE
ENFORCE_PASSWORD_POLICY	Supported	TRUE
ENTRANCE_EXAMS	Supported	FALSE
FORCE_UNIVERSITY_DOMAIN	Deprecated	FALSE
INDIVIDUAL_DUE_DATES	Supported	FALSE
LICENSING	Supported	FALSE
LOG_POSTPAY_CALLBACKS	Supported	TRUE
MAX_ENROLLMENT_INSTR_BUTTONS	Supported	200
MILESTONES_APP	Supported	FALSE
MODE_CREATION_FOR_TESTING	Development	FALSE
PREVENT_CONCURRENT_LOGINS	Supported	TRUE
REQUIRE_COURSE_EMAIL_AUTH	Supported	TRUE
RUN_AS_ANALYTICS_SERVER_ENABLED	Deprecated	FALSE
SEGMENT_IO_LMS	Deprecated	FALSE
SHIB_DISABLE_TOS	Supported	FALSE
SHOW BUMPER_PERIODICITY	Supported	7 * 24 * 3600
STORE_BILLING_INFO	Supported	FALSE
SUBDOMAIN_BRANDING	Deprecated	FALSE
SUBDOMAIN_COURSE_LISTINGS	Deprecated	FALSE
USE_DJANGO_PIPELINE	Supported	TRUE
USE_MICROSITES	Supported	FALSE

Glossary

A - C - D - E - F - G - H - I - K - L - M - N - O - P - R - S - T - V - W - XYZ

Note: Most of the links to documentation provided in this glossary are to the [Building and Running an edX Course](#) guide, for edX partners. Many of the same topics are available in the Open edX version of this guide, [Building and Running an Open edX Course](#).

9.1 A

A/B Test

See [Content Experiment](#).

About Page

The course page that provides potential learners with a course summary, prerequisites, a course video and image, and important dates.

For more information, see [The Course About Page](#).

Accessible Label

In a problem component, you use special formatting to identify the specific question that learners will answer by selecting options or entering text or numeric responses.

This text is referred to as the accessible label because screen readers read all of the text that you supply for the problem and then repeat the text that is identified with this formatting immediately before reading the answer choices for the problem. This text is also used by reports and Insights to identify each problem.

All problems require accessible labels.

For more information, see [The Simple Editor](#).

Advanced Editor

An XML-only editor in a problem component that allows you to create and edit any type of problem. For more information, see [The Advanced Editor](#).

Assignment Type

The category of graded student work, such as homework, exams, and exercises. For more information, see [Establishing a Grading Policy For Your Course](#).

9.2 C

CAPA Problem

A “Computer Assisted Personalized Approach” (CAPA) problem refers to any of the problem types that are implemented in the edX platform by the `capa_module` XBlock. Examples range from text input, drag and drop, and math expression input problem types to circuit schematic builder, custom JavaScript, and chemical equation problem types.

Other assessment methods are also available, and implemented using other XBlocks. An open response assessment is an example of a non-CAPA problem type.

Certificate

A document issued to an enrolled learner who successfully completes a course with the required passing grade. Not all edX courses offer certificates, and not all learners enroll as certificate candidates. For information about setting up certificates for your course, see [Setting Up Course Certificates](#).

Chapter

See [Section](#).

Checkbox Problem

A problem that prompts learners to select one or more options from a list of possible answers. For more information, see [Checkbox Problem](#).

Chemical Equation Response Problem

A problem that allows learners to enter chemical equations as answers. For more information, see [Chemical Equation Problem](#).

Circuit Schematic Builder Problem

A problem that allows learners to construct a schematic answer (such as an electronics circuit) on an interactive grid. For more information, see [Circuit Schematic Builder Problem](#).

Closed Captions

The spoken part of the transcript for a video file, which is overlaid on the video as it plays. To show or hide closed captions, you select the CC icon. You can move closed captions to different areas on the video screen by dragging and dropping them.

For more information, see [Watching Videos on the edX Video Player](#).

Cohort

A group of learners who participate in a class together. Learners who are in the same cohort can communicate and share experiences in private discussions.

Cohorts are an optional feature of courses on the edX platform. For information about how you enable the cohort feature, set up cohorts, and assign learners to them, see [Using Cohorts in Your Courses](#).

Component

The part of a unit that contains your actual course content. A unit can contain one or more components. For more information, see [Developing Course Components](#).

Content Experiment

You can define alternative course content to be delivered to different, randomly assigned groups of learners. Also known as A/B or split testing, you use content experiments to compare the performance of learners who have been exposed to different versions of the content. For more information, see [Overview of Content Experiments](#).

Content Library

See [Library](#).

Content-Specific Discussion Topic

A category within the course discussion that appears at a defined point in the course to encourage questions and conversations. To add a content-specific discussion topic to your course, you add a discussion component to a unit. Learners cannot contribute to a content-specific discussion topic until the release date of the section that contains it. Content-specific discussion topics can be divided by cohort, so that learners only see and respond to posts and responses by other members of the cohort that they are in.

For more information, see [Working with Discussion Components](#). For information about making content-specific discussion topics divided by cohort, see [Setting up Discussions in Courses with Cohorts](#).

Course Catalog

The page that lists all courses offered in the edX learning management system.

Course Handouts

Course handouts are files you make available to learners on the **Home** page. For more information, see [Adding Course Updates and Handouts](#).

Course Navigation Pane

The navigation frame that appears at one side of the **Course** page in the LMS. The course navigation pane shows the sections in the course. When you select a section, the section expands to show subsections. When you select a subsection, the first unit in that subsection appears on the course page.

See also [Unit Navigation Bar](#).

Course Run

The term or time frame in which a specific offering of your course takes place. You set the course run when you create your course. For more information, see [Creating a Course](#).

Course Page

The page where learners access the primary instructional materials for your course. Sections, subsections, units, and components are all accessed from the **Course** page. This page was formerly called the **Courseware** page.

Courseware

In Open Learning XML (OLX) and in data packages, “courseware” refers to the main content of your course, consisting mainly of lessons and assessments. Courseware is organized into sections, subsections, units, and components. Courseware does not include handouts, the syllabus, or other course materials.

Note that the **Course** page was formerly called the **Courseware** page.

Course-Wide Discussion Topic

Optional discussion categories that you create to guide how learners find and share information in the course discussion. Course-wide discussion topics are accessed from the **Discussion** page in your course. Examples of course-wide discussion topics include Announcements and Frequently Asked Questions. Learners can contribute to these topics as soon as your course starts. For more information, see [Managing Course Discussions](#) and [Create Course-Wide Discussion Topics](#).

If you use cohorts in your course, you can divide course-wide discussion topics by cohort, so that although all learners see the same topics, they only see and respond to posts and responses by other members of the cohort that they are in. For information about configuring discussion topics in courses that use cohorts, see [Setting up Discussions in Courses with Cohorts](#).

Custom Response Problem

A custom response problem evaluates text responses from learners using an embedded Python script. These problems are also called “write-your-own- grader” problems. For more information, see [Write-Your-Own-Grader Problem](#).

9.3 D

Data Czar

A data czar is the single representative at a partner institution who is responsible for receiving course data from edX, and transferring it securely to researchers and other interested parties after it is received.

For more information, see the [EdX Research Guide](#).

Discussion

The set of topics defined to promote course-wide or unit-specific dialog. Learners use the discussion topics to communicate with each other and the course team in threaded exchanges. For more information, see [Managing Course Discussions](#).

Discussion Component

Discussion topics that course teams add directly to units. For example, a video component can be followed by a discussion component so that learners can discuss the video content without having to leave the page. When you add a discussion component to a unit, you create a content-specific discussion topic. See also [Content Specific Discussion Topic](#).

For more information, see [Working with Discussion Components](#).

Discussion Thread List

The navigation frame that appears at one side of the **Discussion** page in the LMS. The discussion thread list shows the discussion categories and subcategories in the course. When you select a category, the list shows all of the posts in that category. When you select a subcategory, the list shows all of the posts in that subcategory. Select a post to read it and its responses and comments, if any.

Dropdown Problem

A problem that asks learners to choose from a collection of answer options, presented as a drop-down list. For more information, see [Dropdown Problem](#).

9.4 E

edX101

An online course about how to create online courses. The intended audience for [edX101](#) is faculty and university administrators.

edX Edge

[edX Edge](#) is a less restricted site than edX.org. While only edX employees and consortium members can create and post content on edX.org, any users with course creator permissions for Edge can create courses with Studio on [studio.edge.edx.org](#), then view the courses on the learning management system at [edge.edx.org](#).

edX Studio

The edX tool that you use to build your courses. For more information, see [Getting Started with Studio](#).

Embargo

An embargo is an official ban on trade or commercial activity with a particular country. For example, due to U.S. federal regulations, edX cannot offer certain courses (for example, particular advanced STEM courses) on the edx.org website to learners in embargoed countries. Learners cannot access restricted courses from an embargoed country. In some cases, depending on the terms of the embargo, learners cannot access any edX courses at all.

Exercises

Practice or practical problems that are interspersed in edX course content to keep learners engaged. Exercises are also an important measure of teaching effectiveness and learner comprehension. For more information, see [Adding Exercises and Tools](#).

Export

A tool in edX Studio that you use to export your course or library for backup purposes, or so that you can edit the course or library directly in XML format. See also [Import](#).

For more information, see [Export a Course](#) or [Export a Library](#).

9.5 F

Forum

See [Discussion](#).

9.6 G

Grade Range

Thresholds that specify how numerical scores are associated with grades, and the score that learners must obtain to pass a course.

For more information, see [Set the Grade Range](#).

Grading Rubric

See [Rubric](#).

9.7 H

Home Page

The page that opens first every time learners access your course. You can post announcements on the **Home** page. The handout navigation sidebar appears at the side of this page. This page was formerly called the **Course Info** page.

HTML Component

A type of component that you can use to add and format text for your course. An HTML component can contain text, lists, links, and images. For more information, see [Working with HTML Components](#).

9.8 I

Image Mapped Input Problem

A problem that presents an image and accepts clicks on the image as an answer. For more information, see [Image Mapped Input Problem](#).

Import

A tool in edX Studio that you use to load a course or library in XML format into your existing course or library. When you use the Import tool, Studio replaces all of your existing course or library content with the content from the imported course or library. See also [Export](#).

For more information, see [Import a Course](#) or [Import a Library](#).

9.9 K

Keyword

A variable in a bulk email message. When you send the message, a value that is specific to the each recipient is substituted for the keyword.

9.10 L

Label

See [Accessible Label](#).

LaTeX

A document markup language and document preparation system for the TeX typesetting program. In edX Studio, you can [Import LaTeX Code into an HTML Component](#).

Learning Management System (LMS)

The platform that learners use to view courses, and that course team members use to manage learner enrollment, assign team member privileges, moderate discussions, and access data while the course is running.

Learning Sequence

See [Unit Navigation Bar](#).

Left Pane

See [Course Navigation Pane](#).

Library

A pool of components for use in randomized assignments that can be shared across multiple courses from your organization. Course teams configure randomized content blocks in course outlines to reference a specific library of components, and randomly provide a specified number of problems from that content library to each learner.

For more information, see [Working with Content Libraries](#) and [Randomized Content Blocks](#).

Live Mode

A view that allows the course team to review all published units as learners see them, regardless of the release dates of the section and subsection that contain the units. For more information, see [View Your Live Course](#).

LON-CAPA

The LearningOnline Network with Computer-Assisted Personalized Approach e-learning platform. The structure of CAPA problem types in the edX platform is based on the [LON-CAPA](#) assessment system, although they are not compatible.

See also [CAPA Problems](#).

9.11 M

Math Expression Input Problem

A problem that requires learners to enter a mathematical expression as text, such as $e=mc^2$.

For more information, see [Entering Mathematical and Scientific Expressions](#) in the *EdX Learner's Guide*.

MathJax

A LaTeX-like language that you use to write equations. Studio uses MathJax to render text input such as x^2 and $\sqrt{x^2-4}$ as “beautiful math.”

For more information, see [Using MathJax for Mathematics](#).

Module

An item of course content, created in an XBlock, that appears on the **Course** page in the edX learning management system. Examples of modules include videos, HTML-formatted text, and problems.

Module is also used to refer to the structural components that organize course content. Sections, subsections, and units are modules; in fact, the course itself is a top-level module that contains all of the other course content as children.

Multiple Choice Problem

A problem that asks learners to select one answer from a list of options. For more information, see [Multiple Choice Problem](#).

9.12 N

Numerical Input Problem

A problem that asks learners to enter numbers or specific and relatively simple mathematical expressions. For more information, see [Numerical Input Problem](#).

9.13 O

Open Response Assessment

A type of assignment that allows learners to answer with text, such as a short essay and, optionally, an image or other file. Learners then evaluate each others' work by comparing each response to a [rubric](#) created by the course team.

These assignments can also include a self assessment, in which learners compare their own responses to the rubric, or a staff assessment, in which members of course staff evaluate learner responses using the same rubric.

For more information, see [Open Response Assessments](#).

9.14 P

Pages

Pages organize course materials into categories that learners select in the learning management system. Pages provide access to the course content and to tools and uploaded files that supplement the course. Links to each page appear in the course material navigation bar.

For more information, see [Adding Pages to a Course](#).

Partner Manager

Each EdX partner institution has an edX partner manager. The partner manager is the primary contact for the institution's course teams.

Pre-Roll Video

A short video file that plays before the video component selected by the learner. Pre-roll videos play automatically, on an infrequent schedule.

For more information, see [Adding a Pre-Roll Video to Your edX Course](#).

Preview Mode

A view that allows you to see all the units of your course as learners see them, regardless of the unit status and regardless of whether the release dates have passed.

For more information, see [Preview Course Content](#).

Problem Component

A component that allows you to add interactive, automatically graded exercises to your course content. You can create many different types of problems.

For more information, see [Working with Problem Components and Adding Exercises and Tools](#).

Progress Page

The page in the learning management system that shows learners their scores on graded assignments in the course. For more information, see [Checking Your Progress in a Course](#) in the *EdX Learner's Guide*.

9.15 Q

Question

A question is a type of post that you or a learner can add to a course discussion topic to bring attention to an issue that the discussion moderation team or learners can resolve.

For more information, see [Managing Course Discussions](#).

9.16 R

Research Data Exchange (RDX)

An edX program that allows participating partner institutions to request data for completed edx.org courses to further approved educational research projects. Only partner institutions that choose to participate in RDX contribute data to the program, and only researchers at those institutions can request data from the program.

For more information, see [Research Data Exchange](#).

Rubric

A list of the items that a learner's response should cover in an open response assessment. For more information, see the [Rubric](#) topic in [Open Response Assessments](#).

See also *Open Response Assessment*.

9.17 S

Section

The topmost category in your course outline. A section can represent a time period or another organizing principle for course content. A section contains one or more subsections.

For more information, see [Developing Course Sections](#).

Sequential

See *Subsection*.

Short Course Description

The description of your course that appears on the edX [Course List](#) page.

For more information, see [Describe Your Course](#).

Simple Editor

The graphical user interface in a problem component that contains formatting buttons and is available for some problem types. For more information, see [Editing a Problem in Studio](#).

Split Test

See *Content Experiment*.

Subsection

A division in the course outline that represents a topic in your course, such as a lesson or another organizing principle. Subsections are defined inside sections and contain units.

For more information, see [Developing Course Subsections](#).

9.18 T

Text Input Problem

A problem that asks learners to enter a line of text, which is then checked against a specified expected answer.

For more information, see [Text Input Problem](#).

Transcript

A text version of the content of a video. You can make video transcripts available to learners.

For more information, see [Step 2. Create or Obtain a Video Transcript in Working with Video Components](#).

9.19 U

Unit

A unit is a division in the course outline that represents a lesson. Learners view all of the content in a unit on a single page.

For more information, see [Developing Course Units](#).

Unit Navigation Bar

The horizontal control that appears at the top of the **Course** page in the LMS. The unit navigation bar contains an icon for each unit in the selected subsection. When you move your pointer over one of these icons, the name of the unit appears. If you have bookmarked a unit, the unit navigation bar includes an identifying flag above that unit's icon.

See also [Course Navigation Pane](#).

9.20 V

Vertical

See [Unit](#).

Video Component

A component that you can use to add recorded videos to your course.

For more information, see [Working with Video Components](#).

9.21 W

Whitelist

In edX courses, a whitelist is a list of learners who are being provided with a particular privilege. For example, whitelisted learners can be specified as being eligible to receive a certificate in a course, regardless of whether they would otherwise have qualified based on their grade.

In the grade report for a course, whitelisted learners have a value of “Yes” in the **Certificate Eligible** column, regardless of the grades they attained. For information about the grade report, see [Interpreting the Grade Report](#).

Wiki

The page in each edX course that allows both learners and members of the course team to add, modify, or delete content. Learners can use the wiki to share links, notes, and other helpful information with each other. For more information, see [Using the Course Wiki](#).

9.22 XYZ

XBlock

EdX's component architecture for writing course components: XBlocks are the components that deliver course content to learners.

Third parties can create components as web applications that can run within the edX learning management system. For more information, see [Open edX XBlock Tutorial](#).

XSeries

A set of related courses in a specific subject. Learners qualify for an XSeries certificate when they pass all of the courses in the XSeries. For more information, see [XSeries Programs](#).