IITB Ekalavya Summer Internship Programme

2019 Project Report

IITBombayX Platform Upgrade

9 July, 2019

UNDER THE GUIDANCE OF

Professor. S Sudarshan

Contributors:

Mr. Harsh Soni

Ms. Prachi Pandey

Ms. Shreya Kumari

Mentors:

Ms. Aparna Pansare

Mrs. Sarita Kadhao

Acknowledgement

We would like to express our deepest gratitude to Prof. S Sudarshan and IIT-B for providing us with this opportunity and having faith in our abilities. We would like to thank our mentors Ms. Aparna Pansare, Mrs. Sarita Kadhao for providing constant support and giving us a broader picture of the whole scenario without which the completion of the project would not have been possible. Their constant endeavor and guidance have helped us complete the project within the stipulated time. We express our sincere gratitude to all the system administrators in MOOC Lab, Old Computer Science Department for their constant help. It would be very difficult to not mention Mr. Rahul Kharat for all the administrative help and making our working environment comfortable. Last but not the least, we would like to thank our fellow interns for helping us out with all the problems that we faced and making our internship experience an enjoyable and memorable one.

Summer Internship 2019

Project Approval Certificate
Department of Computer Science and Engineering

Indian Institute of Technology Bombay

The project entitled **IITBombayX Platform Upgrade** submitted by Mr. Harsh Soni, Ms. Prachi Pandey, Ms. Shreya Kumari is approved for Summer Internship 2019 Eklavya programme from 19th May 2019 to 9th July 2019, at Department of Computer Science and Engineering, IIT Bombay.

.....

Prof. S Sudarshan
प्राध्यापक/Professor
संगणक विज्ञान एवं अभियांत्रिकी विभाग
Department of Computer Science Engineering
भारतीय प्रौद्योगिकी संस्थान, मुंबई
Indian Institute of Technology, Bombay
पवई/Powai, मुंबई-७६/Mumbai-76.

Place: IIT Bombay, Mumbai
Date: 9 July, 2019

.....

Ms. Aparna Pansare
(Mentor in charge)

Declaration

I declare that this written submission represents my ideas in my own words and whenever others ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Mr. Harsh Soni

NIT Raipur

Ms. Prachi Pandey

NIT Uttarakhand

Ms. Shreya Kumari

NIT Uttarakhand

Abstract

The aim of this project is to upgrade the IITBombayX platform from Ginkgo to Ironwood version. Along with the migration, we also need to verify that the data migration was successful this was verified though shell scripting and the features were tested using selenium automation. The migration is done in two steps i.e. from Ginkgo to Hawthorn and then from Hawthorn to Ironwood. Each tool is equipped to analyze the correctness of the performance and participation of the various group members.

Contents

1 Introduction

1.1 Overview:

IITBombayX is an online platform developed by IIT Bombay, to offer Massive Open Online Courses (MOOCs) for individuals from varying backgrounds. It specializes in Hybrid MOOCs which captures the benefits of flipped classrooms, online lectures, and live interactions with the IITBombayX course instructors.

Open edX is a massive open online course (MOOC) provider. It hosts online university-level courses in a wide range of disciplines to a worldwide student body, including some courses at no charge. It also conducts research into learning based on how people use its platform.

Open edX Platform Releases:

- **Open edX** Ironwood **Release**
- **Open edX** Hawthorn **Release**
- **Open edX** Ginkgo **Release**
- **Open edX** Ficus **Release**
- **Open edX** Eucalyptus **Release**
- **Open edX** Dogwood **Release**
- **Open edX** Cypress **Release**
- **Open edX** Birch **Release**

Data migration is the process of selecting, preparing, extracting, and transforming data and permanently transferring it from one computer storage system to another. Additionally, the validation of migrated data for completeness and the decommissioning of legacy data storage are considered part of the entire data migration process. Data migration is a key consideration for any system implementation, upgrade, or consolidation, and it is typically performed in such a way as to be as automated as possible, freeing up human resources from tedious tasks. Data migration occurs for a variety of reasons, including server or storage equipment replacements, maintenance or upgrades, application migration, website consolidation, disaster recovery, and data center relocation.

1.2 Purpose

Our project aimed to upgrade the IITBombayX platform from Ginkgo to Ironwood version via Hawthorn.
We don't want to lose valuable data about customers in the old system and start with the empty new system, so we need to migrate the data successfully.
We dealt with the following aspects:

Planning:

The data, applications, etc. that will be migrated are selected based on technical requirements and dependencies. Feasible migration and back-out scenarios are developed, as well as the associated tests, automation scripts, mappings, and procedures.

Migration:

All Software requirements are validated, and migration procedures are customized as necessary. Some sort of pre-validation testing may also occur to ensure requirements and customized settings function as expected. If all is deemed well, migration begins, including the primary acts of data extraction, where data is read from the old system, and data loading where data is written to the new system.

Post-migration:

After data migration, results are subjected to data verification to determine whether data was accurately translated, is complete, and supports processes in the new system. During verification, there may be a need for a parallel run of both systems to identify areas of disparity and forestall erroneous data loss. Additional documentation and reporting of the migration project is conducted, and once the migration is validated complete, legacy systems may also be decommissioned.

1.3 Scope

A **shell script** is a computer program designed to be run by the Unix shell, a command-line interpreter. The various dialects of shell scripts are considered to be scripting languages. Typical operations performed by shell scripts include file manipulation, program execution, and printing text.

The scripts used here to perform migration from Ginkgo to Hawthorn and then from Hawthorn to Ironwood release are suitable for performing migration from any Open edX platform to any other.

Also, the analyzing part remains the same.

2 Objective

2.1 Idea

The main aim is to perform data migration and test whether data is migrated successfully. Further we need to analyse the differences in the features of old release (Ginkgo) and new release(s) (Hawthorn and Ironwood).

Advantages of Data Migration:

Following lists the advantages of migration,

1. Safeguard against data corruption and loss during tape-to-tape copying.
2. Leverage established policies and procedures, with minimal manual effort.
3. Your data will be better organized, and you'll have more resources to spend on other.

2.2 Getting Started

For getting started with the process of migration one needs to perform following steps,

1) Taking backup of the data present in the old release and new release (mysql and mongo dump).
2) After the dump has been taken, analyse the tables and columns of both releases.
3) The above analysis can be done easily by the help of Information Schema (present in MySQL), diff command and MS Excel.
4) Restore the dump from old release to the new release and the old release schema-data will be restored in the new release.
5) Perform the Migration steps as per the new release to modify the schema-data according to the new release.
6) After the Migration step is completed, you need to ensure your migration was as according to the release and that all features are properly working on the new release.
7) For this you need to analyse the Before and After report result of your migration.
8) After ensuring the safe migration of data from old release to the new release, testing of features is to be done on the new release.
9) Write Selenium scripts for the automated testing of features on the new release.
10) The Process is completed.

3 Description

3.1 Platform

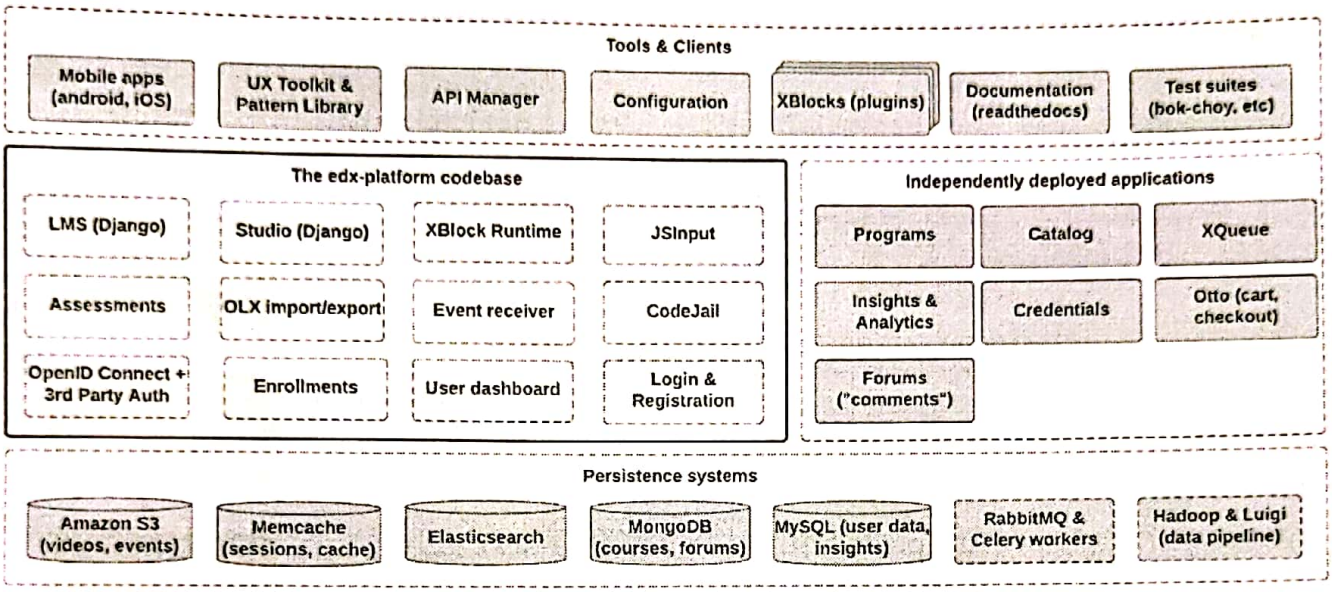

Fig 1- edX-Platform Description

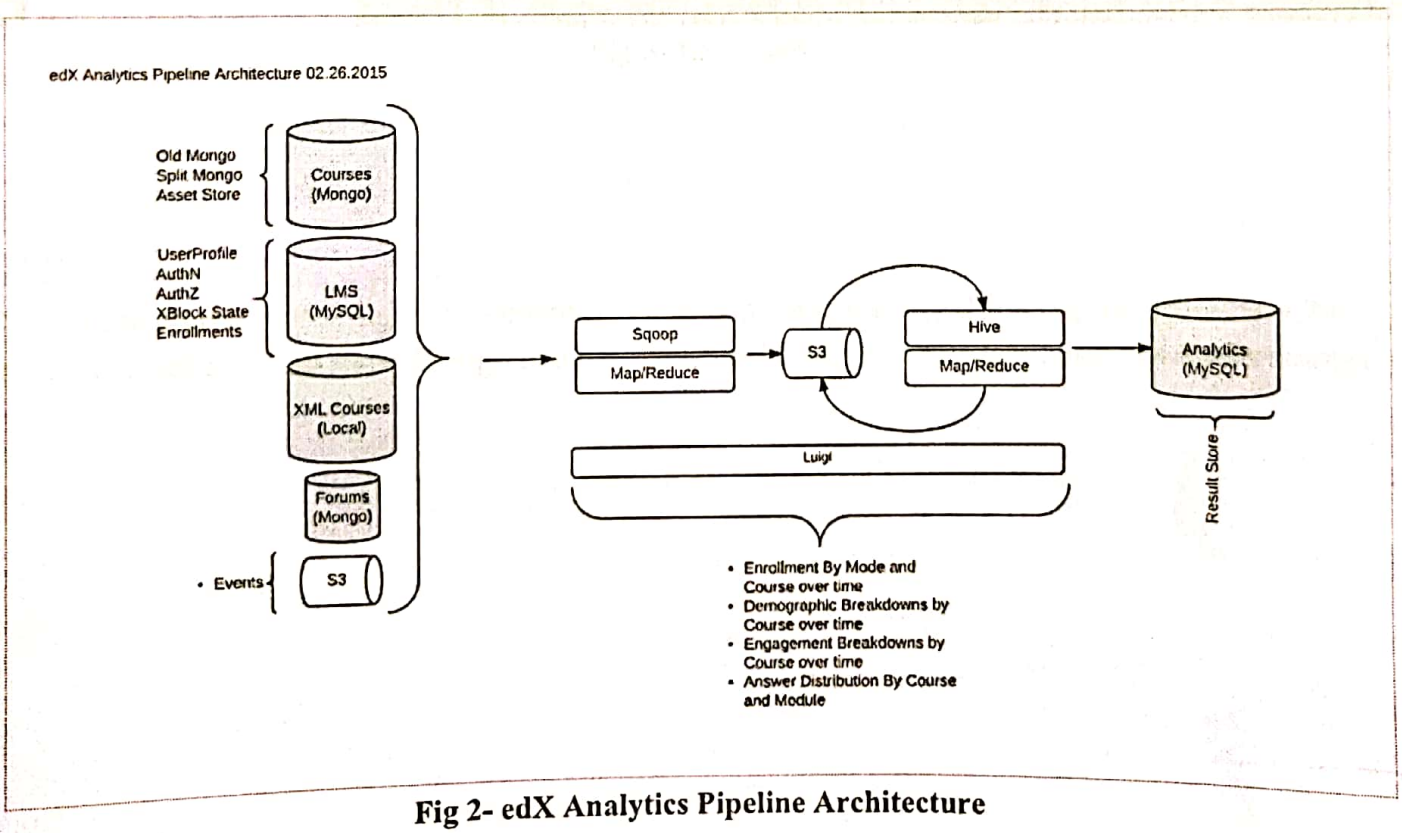

Fig 2- edX Analytics Pipeline Architecture

LMS :

The LMS is the most visible part of the Open edX project. Learners take courses using the LMS. The LMS also provides an instructor dashboard that users who have the Admin or Staff role can access by selecting Instructor.

The LMS uses a number of data stores. Courses are stored in MongoDB, with videos served from YouTube or Amazon S3. Per-learner data is stored in MySQL.

As learners move through courses and interact with them, events are published to the analytics pipeline for collection, analysis, and reporting.

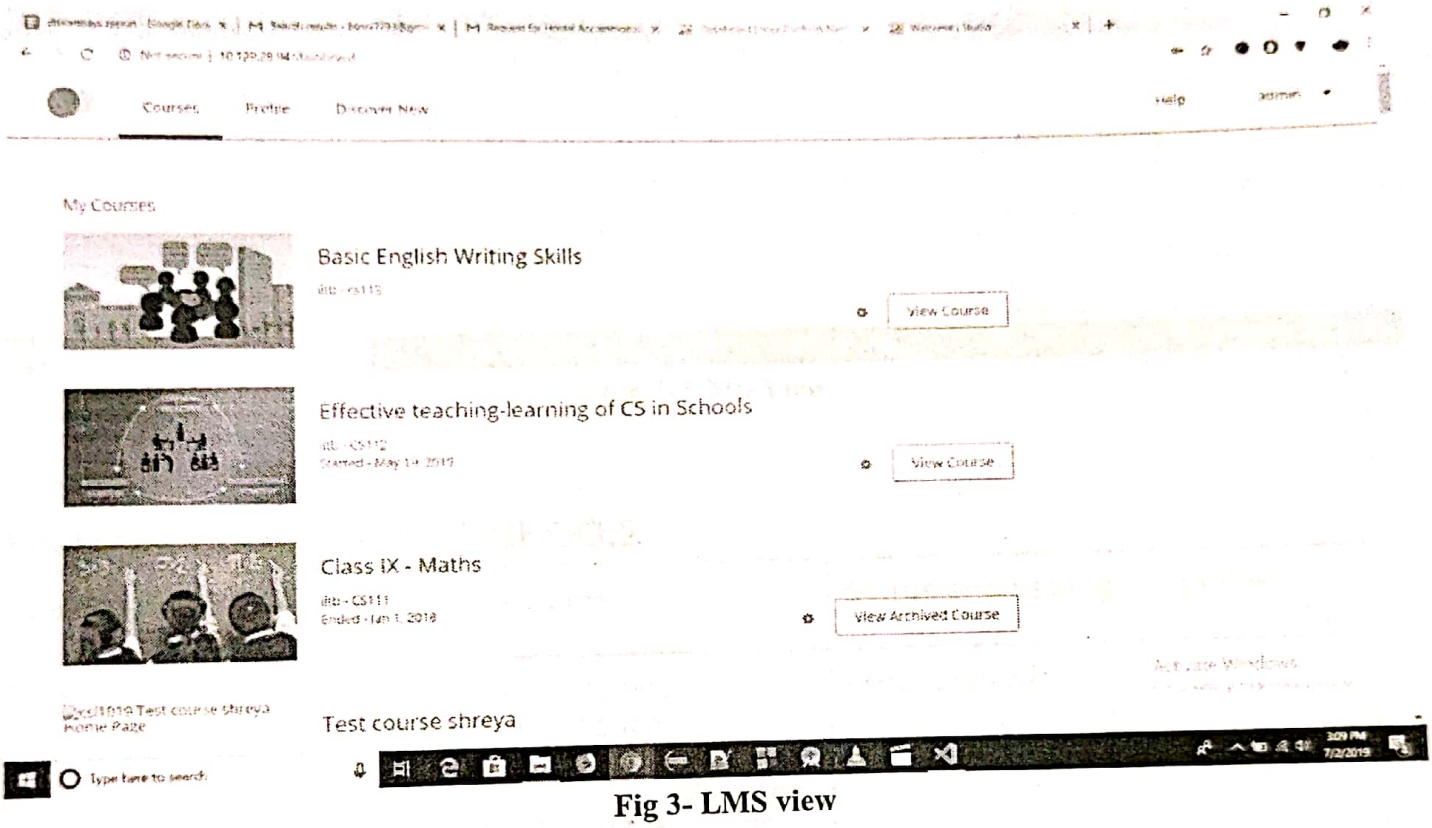

Fig 3- LMS view

CMS :

CMS is the course authoring environment. Course teams use it to create and update courses. CMS writes its courses to the same Mongo database that the LMS uses. It uses mongo DB for its storage.

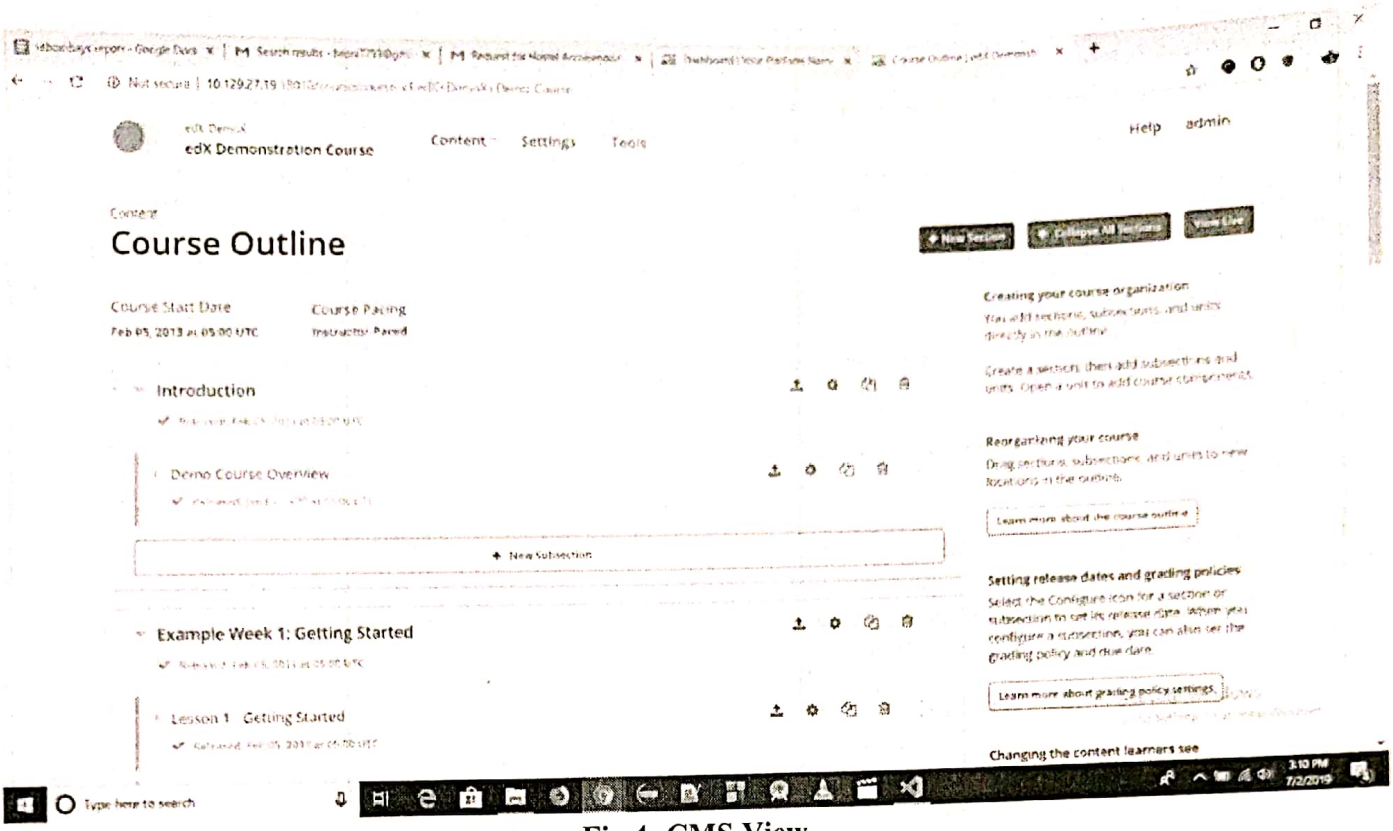

Fig 4- CMS View

Features of Open edX GINKGO.2

Features	About Features	Status and setting for enable
1.1. Course Navigation	Lists all sections and subsections at once http://10.129.103.95/courses/course-v1:<COURSE NAME>course/	- Pre Enabled
1.2. HLS video playback	HTML Live stream, used when YouTube videos are not available. http://10.129.103.95/dashboard	- Pre Enabled

1.3. Search (Outline)	Search functionality is available on the Outline page on the top right corner.
	
	Feature Available URL:
	
	Refer:https://edx.readthedocs.io/projects/edx-installing-configuring-and-running/en/latest/configuration/edx_search.html#install-edx-search
		Setting for enabling the features:
		File: *edx/app/edxapp/edx-platform/lms/envs/common.py*
		`# Dashboard search feature`
		`'ENABLE_DASHBOARD_SEARCH': True,`
		
		`# Course discovery feature`
		`'ENABLE_COURSE_DISCOVERY': True,`
		
		`# Use None for the default search engine`

		SEARCH_ENGINE = "search.elastic.ElasticSearchEngine" # Use LMS specific search initializer SEARCH_INITIALIZER = "lms.lib.courseware_search.lms_search_initializer.LmsSearchInitializer" # Use the LMS specific result processor SEARCH_RESULT_PROCESSOR = "lms.lib.courseware_search.lms_result_processor.LmsSearchResultProcessor" # Use the LMS specific filter generator SEARCH_FILTER_GENERATOR = "lms.lib.courseware_search.lms_filter_generator.LmsSearchFilterGenerator" # Override to skip enrollment start date filtering in course search SEARCH_SKIP_ENROLLMENT_START_DATE_FILTERING = False **File:** *edx/app/edxapp/edx-platform/cms/envs/common.py* 'ENABLE_COURSEWARE_INDEX': True, 'ENABLE_LIBRARY_INDEX': True, SEARCH_ENGINE = "search.elastic.ElasticSearchEngine"
1.4. Bulk Email	Course staff can now send emails to learners based on their enrollment track, so you can reach all verified or audit track learners at one time. http://10.129.103.95/courses/course-v1:<COURSE NAME>/instructor#view-send_email	Access to CMS

1.5. Drag and Drop Questions	Drag and Drop problems can be rescored http://10.129.103.95:18010/course/course-v1:<COURSE_NAME>	Access to CMS
1.6. Admin approval	Only admin can allot courses to the studio users	Access to CMS
1.7. Video ID field	Now you can provide **ID** to your video	Access to CMS
1.8. Open Response Assessment	Problem component	Access to CMS

Follow this link for more info about Ginkgo : https://www.appsembler.com/blog/open-edx-ginkgo-whats-new/

Features of Open edX HAWTHORN.2

Features	About Features	Status and Setting for enable
2.1. Improved learner experience through **Completion API.**	Green completion checkmarks appear on the course outline view(LMS). **Feature Available URL:** /courses/course-v1:INT-IIT+CSI1+19-20/course/	Settings in admin panel: Need to enable a feature flag in the Django admin. In the LMS administration, navigate to Waffle->Switches and add a new switch **"completion.enable_completion_tracking"** Refer to:https://www.appsembler.com/blog/open-edx-hawthorn/
2.2. Optimized video quality with adaptive video streaming	Both video streaming and adaptive streaming options are available.	Refer to:https://www.appsembler.com/blog/open-edx-hawthorn/
2.3. Easy File uploads	Drag and Drop option available to insert new files. Previous files preview available with the feature to directly upload them again,if required.It is available in the CMS section. **Feature Available URL:** /assets/course-v1:nituk+cs113+2019_21/ From the Content drop down menu,choose Files & Upload	Refer to:https://www.appsembler.com/blog/open-edx-hawthorn/

	option.	
2.4. Weekly highlights of Email	Few highlights are specified for each section and Open edX will send out a weekly email message listing these highlights to learners.	Setting in admin panel: Need to add a switch in Waffle->Switches in LMS Django admin, **`dynamic_pacing.studio_course_update`** Refer to: https://github.com/edx/edx-platform/tree/master/openedx/core/djangoapps/schedules#configuring-highlights-ui-in-studio https://www.appsembler.com/blog/open-edx-hawthorn/
2.5. Improved Learners profile	The Learner Profile page now shows the date the learner joined the platform and any course credentials they've received. It also includes links to their social media accounts if they've opted to share those. **Feature Available URL:** /account/settings To view the profile: In the LMS,From the left side drop down menu ,choose profile option.	To know how to add the social links refer to the following link: https://www.appsembler.com/blog/open-edx-hawthorn/
2.6. Improved Discussion Forums	The first time a learner's post receives a comment, they will recieve an email message which contains the comment and a link back to the course discussion post.	Settings in admin panel: Need to enable forum notifications in Django admin in **Site Configuration->Site_configurations** and set the value to { **`"COURSE_CATALOG_API_URL":"http://edx.devstack.discovery:18381/api/v1/","enable_forum_notifications":true}`** Refer to: https://www.appsembler.com/blog/open-edx-hawthorn/

Follow this link for more info about hawthorn :
https://www.appsembler.com/blog/open-edx-hawthorn/

Features of Open edX IRONWOOD.1

Features	About Features	Status and setting for enable
3.1. Studio Login via the LMS	You can't directly log in to the studio. First, you need to log in the LMS and then you will directly be logged in to the studio as well. **Feature Available URL:** /login?next=%2F	- Pre Enabled.
3.2. Course Launch & Best Practices Checklist	Checklists are summarized on the course outline showing the current number of items successfully completed out of the total remaining. **Feature Available URL:** /course/course-v1:iitb+cs113+2019_21 (The url may change depending upon the course)	- Pre Enabled.
3.3.Video Status Indicators	The status of each video is indicated and also a video gets resumed after what we have watched so far. **Feature Available URL:** courses/course-v1:iitb+CS112+2019_21/course/ (The url may change depending upon the course)	- Pre Enabled.
3.4.Gradebook Application	View and manage grades for all learners in a course. Also, In Studio we can find specific learners and override assignment-specific grades. **Feature Available URL:** courses/course-v1:iitb+CS112+2019_21/instructor#view-data_download (The url may change depending upon the course)	- Pre Enabled.
3.5. Learner Data for a Specific Problem	The downloadable Student State report received readability enhancements and now contains new columns for single problems or for all problems in a unit, subsection, or section, as well as for all problems in the course. **Feature Available URL:** courses/course-v1:iitb+CS112+2019_21/instructor#view-data_download (The url may change depending upon the course)	- No new columns were found in Student grade card relative to Hawthorn but columns for single problem in a unit, subsection, or section are present.

3.6. Password Complexity Requirements	Password cant be similar to username. Also must have more than 3 characters. **Feature Available URL:** /register?next=%2F	- Pre Enabled.
3.7. Public courses	Part of a course that are public allow access to its contents to visitors that have not yet registered or accessed the platform with a username and password.Currently, it is possible to use this new configuration only for HTML and video content sections. It is expected that in the future the support will be expanded to other types of content. **Feature Available URL:** **http://10.129.28.94/**	- The first step to configure a course as public is to add a specific waffle flag for the course in the edxapp LMS Django sysadmin console at: https://yourdomain.com/admin/waffle_utils/waffleflagcourseoverridemodel/ The name of the flag to be added and enabled is seo.enable_anonymous_courseware_access. After this, a new setting item will appear in Studio advanced configurations page, to configure the way this feature will be enabled for the desired course. **http://10.129.28.94:18010/settings/advanced/course-v1:iitb+cs113+2019_21**

Follow this link for info about Ironwood :
https://edx.readthedocs.io/projects/open-edx-release-notes/en/latest/ironwood.html

4 Migration

4.1 Steps for Migration

Introduction

We have upgraded data from current Open edX release (referred to as "old release") to next release (referred to as "new release"). We have performed migration from Gingko to Hawthorn and from Hawthorn to Ironwood version of Open edX.

Section I : Run script to generate reports for old release. These reports will be used for verification to confirm upgrade the data from old release to new release done Successfully.

Section II : Steps for upgrade the data from old release to new release.

Section III : Run script to generate reports for new release. These reports will be used for verification to confirm upgrade the data from old release to new release done Successfully.

Section IV : Run script to find the differences between generated files in section I on old release machine and files in section III on new release machine to confirm that the data has been upgraded from old version to new version.

Section I

Generate data reports of Old release.

In this step, database reports of data existing on old release is created. This will be stored and used to compare with similar reports generated after porting to the new release in subsequent sections. This report can be run from any remote machine.

Pre-requisites:

1. A machine with MySQL server that has database dump of old release restored on it.
2. MySQL database user having all privileges to all databases on this server for remote machine.
3. To check privileges assigned to remote database

```
mysql -u<user_name> -p -h<host-ip-address>  #Exit from MySQL
```

4. A machine with mongo database server that has dump of old release restored on it.
5. Mongo database user having all privileges to all databases on this server.
6. Make sure in Mongo there is no any extra blank database in Mongo.
7. Stop services of old release machine and Confirm "Did the services stop?"

```
sudo /edx/bin/supervisorctl stop all
sudo /edx/bin/supervisorctl status all
```

Steps:

Steps for generating reports on remote machine are as following

1. Clone the repository

Make folder dataportingreport on local machine

```
mkdir  dataportingreport
cd dataportingreport
git clone http://<username>@10.105.24.86/Data-porting/OpenedXDataPorting.git
cd OpenedXDataPorting
git checkout branch_name
```

2. Run report script.:

(a) Run script for generating MySQL and Mongo reports

```
cd scripts/ReportScript
```

#Set parameter in parameter.config file as per instructions given in README.md file

```
bash count_rows.sh
```

Note: Parameter required to run script.

1. "Select process type. Is this process running Before/After the migration?"

"Enter (A/B) Default Before:" b

2. "In MySQL, There are above extra databases or missing databases has on host_server_ip machine. Do you want to continue with common databases only?"

"Enter (y/n) Default Yes:"

Output: After running this script, "Before/ReportResult" folder is created at OpenedXDataPorting/scripts/.

3. Run row count script:

(a) Run script for generating MySQL and Mongo record count

```
cd ../../analysis/record_count/
```

Set parameter in parameter.config file as per instructions given in README.md file, run script

```
bash count_rows.sh
```

Output: After running this script, two folders are created for MySQL and Mongo.

Section II

Steps for upgrading the data from "old release" to "new release". This section can run independently.

Pre-requisites:

1. A new fresh new release installed machine. Make a fresh installation of the new release on a new machine.
2. MySQL user with all privileges on all MySQL databases on new machine. Conform by login in mysql using following command:

```
mysql -u<user_name> -p -h<127.0.0.1>    #Exit from MySQL
```

3. Mongo user with all privileges on all mongo databases on new machine.

Steps:

1. Stop all services accessing the database on old release machine and Confirm "Did the services stop?"

```
sudo /edx/bin/supervisorctl stop all
sudo /edx/bin/supervisorctl status all
```

2. Check disk space availability for taking the dump.

```
$ df
```

3. Take dump of the data (mysql and mongo) on the old release machine.

Make folder Dump

```
mkdir Dump
cd Dump
```

Fetch the file from here:

```
http://10.105.24.86/Data-porting/OpenedXDataPorting/blob/OldToNew/FreshData/Fic.4/F96-FreshBackup/shellscript/dump.sh
```

Verify the name of mongo discussion forum database in the variable "mongo_common_dbs" and correct it if required.

```
bash dump.sh
```

Output: Make sure current working directory has the following things.

- A folder is created which name starts with mongo dump like mongo-dump-20180814T164638. It contains files of dump of mongo databases.
- .sql files which is mysql dump like mysql-structure-data-20180814T164638.sql and
- A .tgz file like openedx-data-20180814T164638.tgz, which has dump of MySQL and Mongo databases.

4. Copy the generated .tgz data file from old release machine to the folder 'DumpforMigrate' on new release machine.

5. Stop all services on the new release machine and Confirm "Did the services stop?"

```
sudo /edx/bin/supervisorctl stop all
sudo /edx/bin/supervisorctl status all
```

6. To restore the old release data into the new release machine use the following commands as an example

```
tar -xvf openedx-data-20180814T164638.tgz
mysql -uroot -p < mysql-structure-data-20180814T164638.sql
mongorestore -u admin -p -h localhost --authenticationDatabase admin --drop
-d edxapp mongo-dump-20180814T164638/edxapp
mongorestore -u admin -p -h localhost --authenticationDatabase admin --drop
-d                    cs_comments_service                    mongo-dump-
20180814T164638/cs_comments_service_development
```

Note: After restore delete the unzip files to release the disk space.

Example:

```
rm mysql-structure-data-20180814T164638.sql
rm -r mongo-dump-20180814T164638/
```

7. Set character set of databases from latin1 to utf8

```
mysql -u root -e "alter database edxapp character set utf8";
mysql -u root -e "alter database ecommerce character set utf8";
mysql -u root -e "alter database edxapp_csmh character set utf8";
mysql -u root -e "alter database xqueue character set utf8";
```

8. To migrate data from old release to new release, first drop the database tables used by djcelery. These tables should be empty in your old release data, so it is safe to drop them. The edx-platform application has a management command to check that they are empty and drop them:

```
sudo su - -s /bin/bash edxapp
. edxapp_env
cd edx-platform/
python manage.py lms drop_djcelery_tables --settings=aws
exit
```

9. Confirm the command 'git branch' in /var/tmp/configuration directory: output should be "* (HEAD detached at open-release/ginkgo.2)"

If it is not, then "git checkout open-release/ginkgo.2"

```
cd /var/tmp/configuration
git branch
# If it is not * (HEAD detached at open-release/ginkgo.2), then
git checkout open-release/ginkgo.2
```

10. Check 'git diff' in /var/tmp/configuration directory. It should have no diff.

```
git diff
```

11. Be sure internet service is start, to check login on internet proxy server as

```
lynx internet.iitb.ac.in
```

12. Set permission to /edx/app/edx_ansible/edx_ansible/util/install/sandbox.sh file.

```
# assign all permission to the file (/edx/app/edx_ansible/edx_ansible/util/install/sandbox.sh)
sudo chmod 777 /edx/app/edx_ansible/edx_ansible/util/install/sandbox.sh
```

13. Set environment variable if not as: OPENEDX_RELEASE=open-release/ginkgo.2 in /edx/app/edx_ansible/edx_ansible/util/install/sandbox.sh file before variable CONFIGURATION_VERSION=${CONFIGURATION_VERSION-${OPENEDX_RELEASE-master}}.

```
sudo vi /edx/app/edx_ansible/edx_ansible/util/install/sandbox.sh
OPENEDX_RELEASE=open-release/ginkgo.2
echo ${OPENEDX_RELEASE-master}
CONFIGURATION_VERSION=${CONFIGURATION_VERSION-${OPENEDX_RELEASE-master}}
```

14. Run the new release migration script, which will upgrade old release data to be valid for new release.

```
# Run sandbox.sh file for upgrade old release data to be valid for new release data
/edx/app/edx_ansible/edx_ansible/util/install/sandbox.sh --tags migrate
```

It may ask for the following :

a) Mail configuration: no configuration?

b) Password for edx user

15. Restart all services if no need to run report script otherwise go for section III and skip above.

```
sudo /edx/bin/supervisorctl start all
```

Section III

Generate data reports for "new release".

In this step, database reports of migrated data for new release will be created. This will be stored and used to compare with reports generated before porting for old release. This report can be run from any remote machine. This section can be run independently.

Pre-requisites:

1. A Ginkgo.2 machine with MySQL server having ported database dump of old release(in this case, Ficus.4) restored on it.
2. MySQL database user having all privileges to all databases on this server for remote machine.
3. To check privileges assigned to remote database

```
$ mysql -u<user_name> -p -h<host-ip-address>
```

4. A machine with mongo database server having database dump of new release restored on it.

5. Mongo database user having all privileges to all databases on this server.

6. Make sure in Mongo there is no any extra blank database in Mongo.

7. If you have to verify result of porting "SUCCESSFUL" or "NOT", don't start services and make confirm after porting no any service is ON.

Stop services of new release machine and confirm "Did the services stop?"

```
$ sudo /edx/bin/supervisorctl stop all
$ sudo /edx/bin/supervisorctl status all
```

Steps:

Steps for generating reports on remote machine are as following

1. Clone the repository

Note: If you have clone of repository already then go in repository directory which is created in section-I and skip this step.

Make folder dataportingreport on local machine

```
mkdir dataportingreport
cd dataportingreport
git clone http://saritat@10.105.24.86/Data-porting/OpenedXDataPorting.git
cd OpenedXDataPorting
git checkout branch_name
```

2. Run report script:

(a) Run script for generating MySQL and Mongo reports

```
cd scripts/ReportScript
```

Set parameter in parameter.config file as per instructions given in README.md file, run

script

```
bash count_rows.sh
```

Note: Parameter required to running script

1. "Select process type. Is this process running Before/After the migration?"

"Enter (A/B) Default Before:" a

2. "In MySQL, There are above extra databases or missing databases has on host_server_ip machine. Do you want to continue with common databases only?"

"Enter (y/n) Default Yes:"

Output: After running this script, "After/ReportResult" folder is created at OpenedXDataPorting/scripts/.

3. Run row count script:

(a) Run script for generating MySQL and Mongo record count

```
cd ../../analysis/record_count/
```

Set parameter in parameter.config file as per instructions given in README.md file, run script

```
bash count_rows.sh
```

Output: After running this script, two folders are created for MySQL and Mongo.

Section IV

Find differences between reports generated for "old machine" in Section-I and "new machine" in Section-III.

In this step, find the differences between generated report files in Section I for old release machine and report files in Section III for Ginkgo machine to confirm upgrade the data from old release to new release done successfully.

This section dependent on report generated in Section I and Section III.

Steps:

Steps for find difference between before report result and after report result on remote machine are as following

1. Clone the repository:

Note: If you have clone of repository already then go in repository directory which is created in section-I and skip this step.

Make folder dataportingreport on local machine

```
mkdir dataportingreport
```

```
cd dataportingreport
git clone http://prachip@10.105.24.86/Data-porting/OpenedXDataPorting.git
cd OpenedXDataPorting
git checkout FicusToGinkgo2
```

2. Copy Report Result:

Note: If you have clone of repository already and report result in folder "scripts/ReportResult/" then go in repository directory which is created in section-I and skip this step, Otherwise

i. Create folder ReportResult in script folder

```
$ cd script
$ mkdir ReportResult
```

ii. copy report result folder in "Before" of old release in folder scripts/ReportResult/

iii. copy report result folder in "After" of new release in folder scripts/ReportResult/

3. Run script:

Run script for generating diff:

```
$ cd scripts/ReportScript
```

Set parameter in parameter.config file as per instructions given in README.md file, run script

```
$ bash findDiff.sh
```

Output: After running this script, "ReportConclusion" folder is created at `OpenedXDataPorting/scripts/.`

Following link can be referred : https://edx.readthedocs.io/projects/edx-installing-configuring-and-running/en/latest/platform_releases/ginkgo.html

4.2 Automated Testing Using Selenium

Selenium is a free (open source) automated testing suite for web applications across different browsers and platforms. Selenium focuses on automating web-based applications. Testing done using Selenium tool is usually referred as Selenium Testing.

Selenium is not just a single tool but a suite of software's, each catering to different testing needs of an organization. **It has four components.**

- Selenium Integrated Development Environment (IDE)

- Selenium Remote Control (RC)
- WebDriver
- Selenium Grid

Selenium Python bindings provides a simple API to write functional/acceptance tests using Selenium WebDriver. Through Selenium Python API you can access all functionalities of Selenium WebDriver in an intuitive way.
Selenium Python bindings provide a convenient API to access Selenium WebDrivers like Firefox, IE, Chrome, Remote etc. The current supported Python versions are 2.7, 3.5 and above.

Selenium Scripts for testing features:

1. Checklist feature in Ironwood

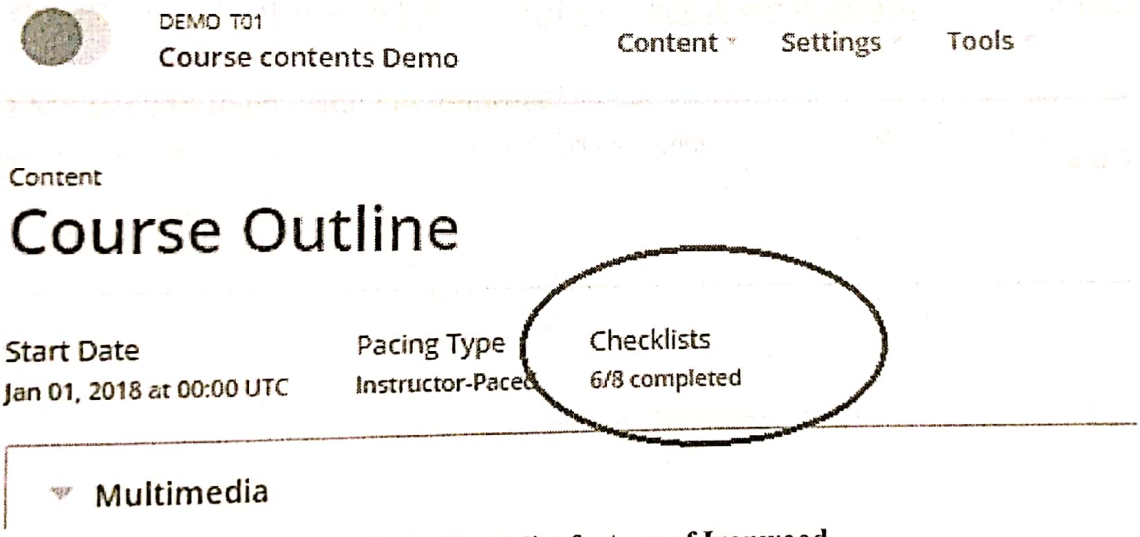

Fig 5- Checklist feature of Ironwood

CODE: https://github.com/harsoni/selenium/blob/master/ironwood_checklist.py

2. Public access feature in Ironwood

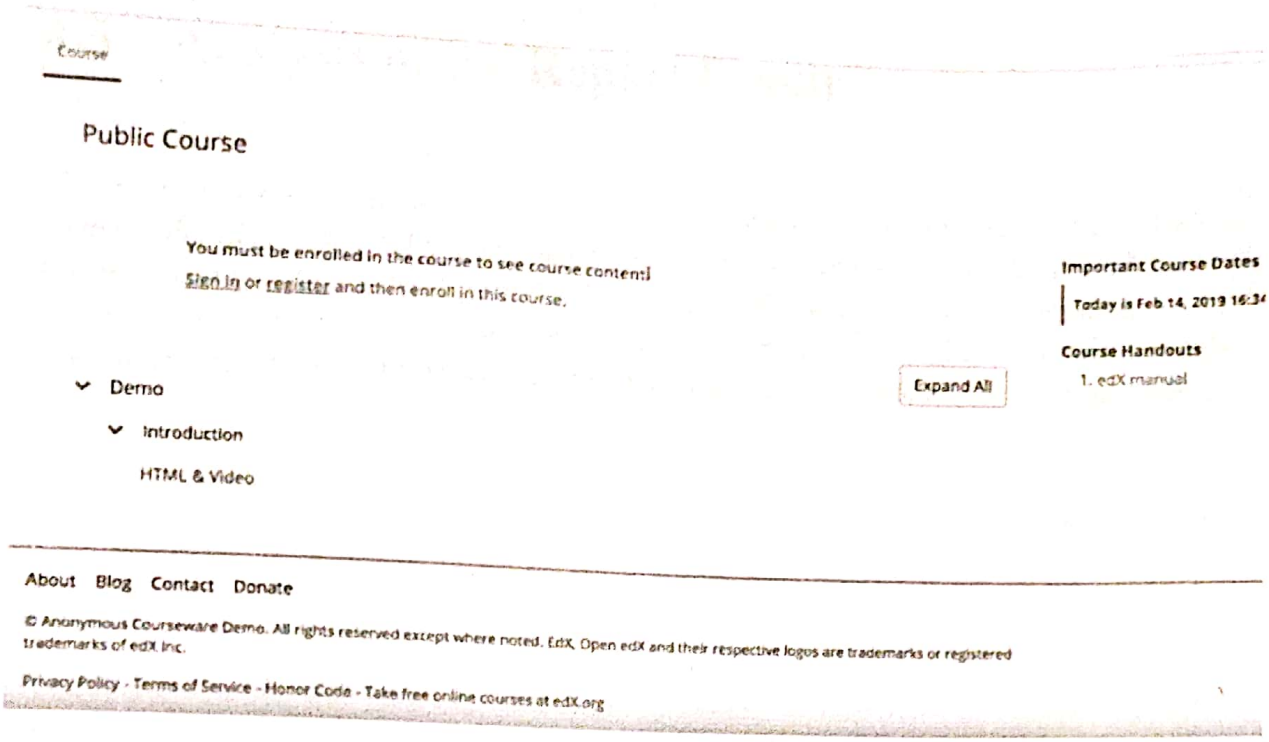

Fig 6- Public Course feature in Ironwood

CODE: https://github.com/harsoni/selenium/blob/master/ironwood_publicaccess.py

3. LMS to CMS login feature in Ironwood

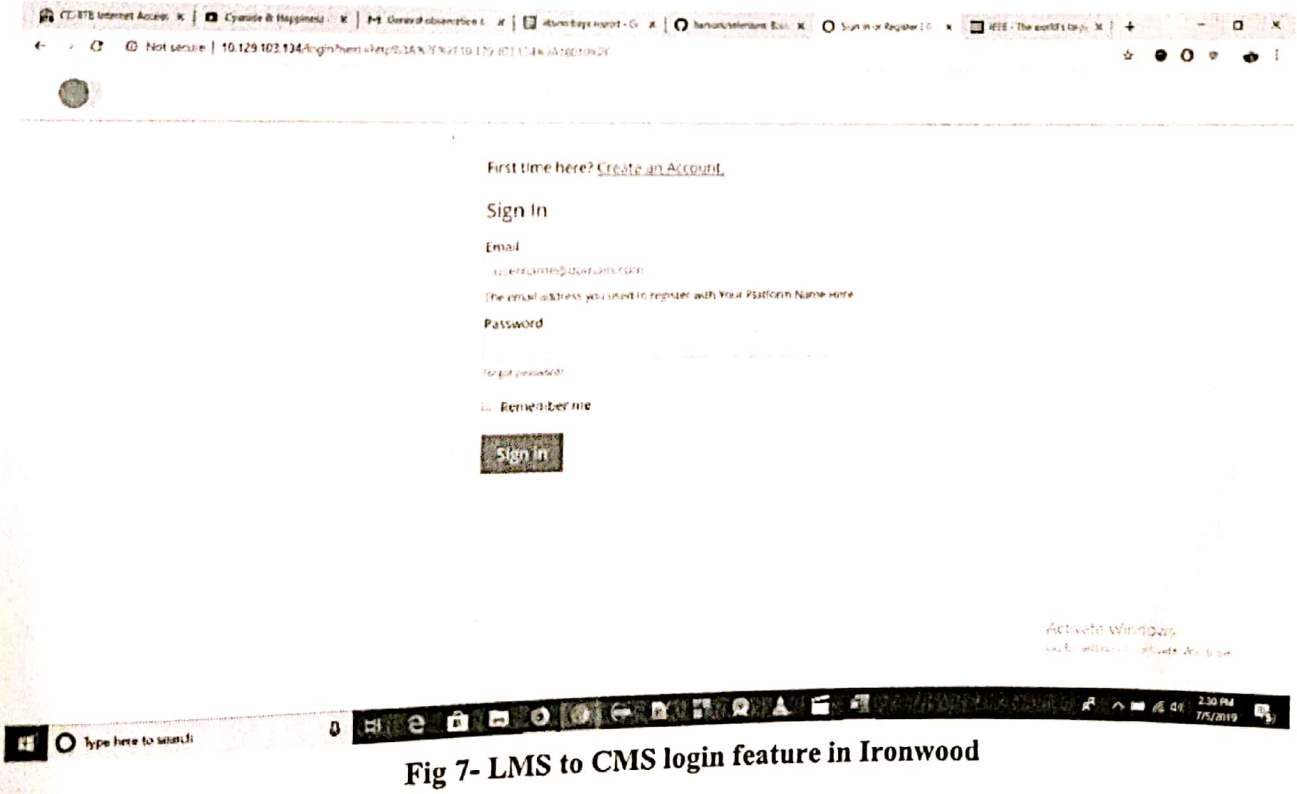

Fig 7- LMS to CMS login feature in Ironwood

CODE: https://github.com/harsoni/selenium/blob/master/ironwood_lms_to_cms.py

4.3 Analysis of the Report Result

Migration was done and two folders were created namely "Before" and "After". Before contains the files of the old version and after of the new version. The difference of the two folders is compared with the expected result. Kindly refer section IV of the migration process for the scripts of folder creation and finding the differences between the actual result and the expected result. If these differences are zero that implies that migration has been successful.

5 Results

5.1 Conclusion

The Migration process was done successfully from Ginkgo to Hawthorn and from Hawthorn to Ironwood. We performed 3 migrations during the process, first without production data and the last two with dummy production data. Reports were compared with the expected output and features were tested using Selenium to ensure that migration process was successful. Further changes can be done in the script to analyse the result better and make better analysis.

6 References

1) https://open.edx.org/

2) https://docs.edx.org/

3) https://edx.readthedocs.io/projects/edx-installing-configuring-and-running/en/latest/platform_releases/ginkgo.html

4) https://edx.readthedocs.io/projects/edx-installing-configuring-and-running/en/latest/platform_releases/hawthorn.html

5) https://edx.readthedocs.io/projects/edx-installing-configuring-and-running/en/latest/platform_releases/ironwood.html

6) https://www.edunext.co/articles/discover-open-edx-ironwood/

7) https://open.edx.org/announcements/release-announcement/ironwood-is-here/

8) https://openedx.atlassian.net/wiki/spaces/AC/pages/23003228/Everything+About+Database+Migrations

9) https://edx.readthedocs.io/projects/edx-developer-guide/en/latest/architecture.html

Appendix

Appendix 1: Technical Specifications

At first, we should know the databases used by IITBomabayX, hence basic knowledge of MySQL and MongoDB is necessary for the first phase of project.

MySQL, pronounced either "My S-Q-L" or "My Sequel," is an open source relational database management system.
MongoDB is a cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with schema.

Then after analyzing part, comes the script writing phase:
A shell script is a computer program designed to be run by the Unix/Linux shell which could be:

Shell :

- The Bourne Shell
- The C Shell
- The Korn Shell
- The GNU Bourne-Again Shell

A shell is a command-line interpreter and typical operations performed by shell scripts include file manipulation, program execution, and printing text.

Migrations are Django's way of propagating changes you make to your models (adding a field, deleting a model, etc.) into your database schema. They're designed to be mostly automatic, but you'll need to know when to make migrations, when to run them, and the common problems you might run into.

Django:

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source.

MVC Structure of Django:

Model: This handles your data representation, it serves as an interface to the data stored in the

database itself, and also allows you to interact with your data without having to get perturbed with all the complexities of the underlying database.

View: As the name implies, it represents what you see while on your browser for a web application or In the UI for a desktop application.

Controller: provides the logic to either handle presentation flow in the view or update the model's data i.e it uses programmed logic to figure out what is pulled from the database through the model and passed to the view,also gets information from the user through the view and implements the given logic by either changing the view or updating the data via the model , To make it more simpler, see it as the engine room.

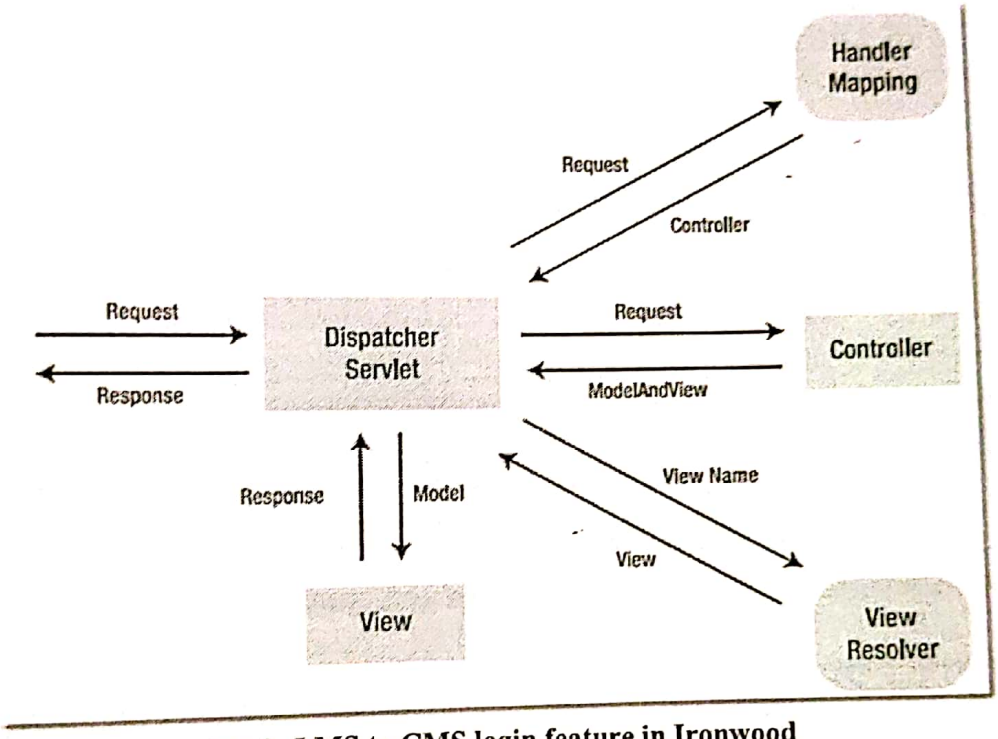

Fig 8- LMS to CMS login feature in Ironwood

Commands:

There are several commands which you will use to interact with migrations and Django's handling of database schema:

- **makemigrations**, which is responsible for creating new migrations based on the changes you have made to your models.

Few examples:

```
python manage.py makemigrations <name_of_app>
python manage.py makemigrations --name=<name_of_migration>
```

```
python manage.py makemigrations library --name=add_publisher
```

- **migrate**, which is responsible for applying and rolling back migrations. Few examples:

```
python manage.py migrate <name_of_app>
python manage.py migrate <name_of_app> <rollback_migration_name>
```

- **sqlmigrate**, which displays the SQL statements for a migration.

- **showmigrations**, which lists a project's migrations and their status.

Appendix 2: Technologies used

MongoDB :

MongoDB is a document database with the scalability and flexibility that you want with the querying and indexing that you need

- MongoDB stores data in flexible, JSON-like documents, meaning fields can vary from document to document and data structure can be changed over time
- The document model maps to the objects in your application code, making data easy to work with
- Ad hoc queries, indexing, and real time aggregation provide powerful ways to access and analyze your data
- MongoDB is a distributed database at its core, so high availability, horizontal scaling, and geographic distribution are built in and easy to use
- MongoDB is free to use.

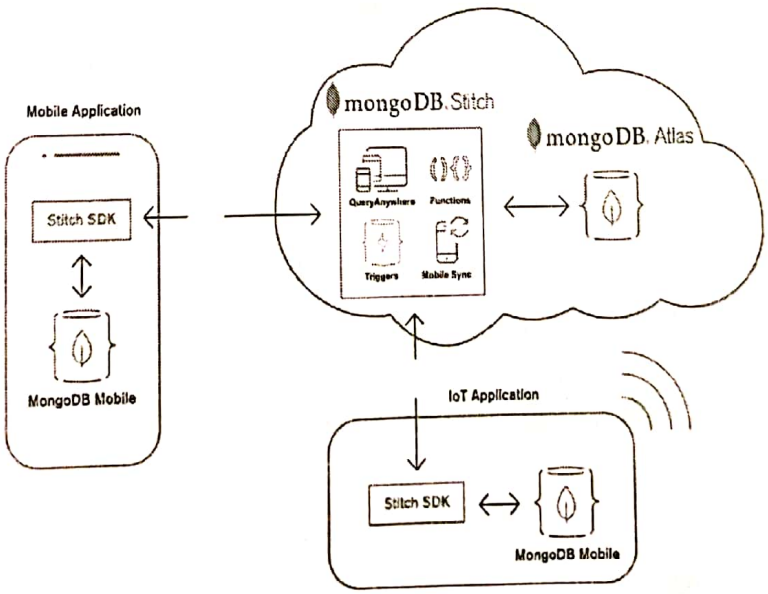

Fig 9-MongoDB Architecture

MySQL:

It is the world's most used open source relational database management that runs as a server providing multi-user access to a number of databases. MySQL is an Oracle-backed open source relational database management system (RDBMS) based on Structured Query Language (SQL). MySQL runs on virtually all platforms, including Linux, UNIX and Windows. MySQL is an important component of an open source enterprise stack called LAMP.

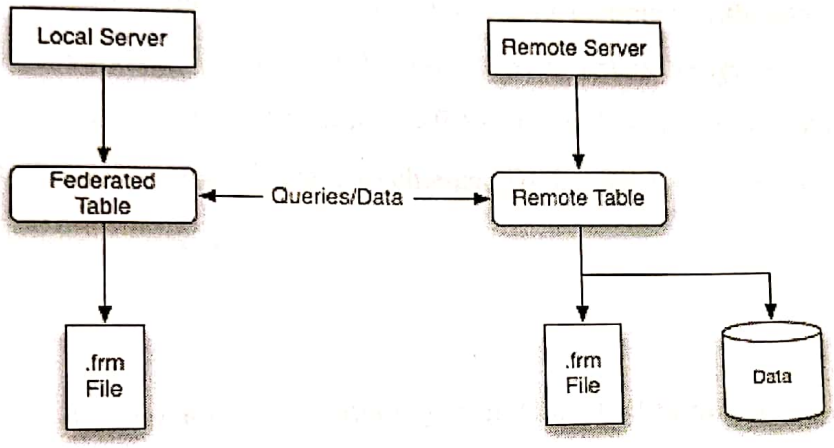

Fig 10- MySQL Architecture

Selenium (using python) :

Selenium is an open-source web-based automation tool. Python language is used with Selenium for testing. It has far less verbose and easy to use than any other programming language. The Python APIs empower you to connect with the browser through Selenium.

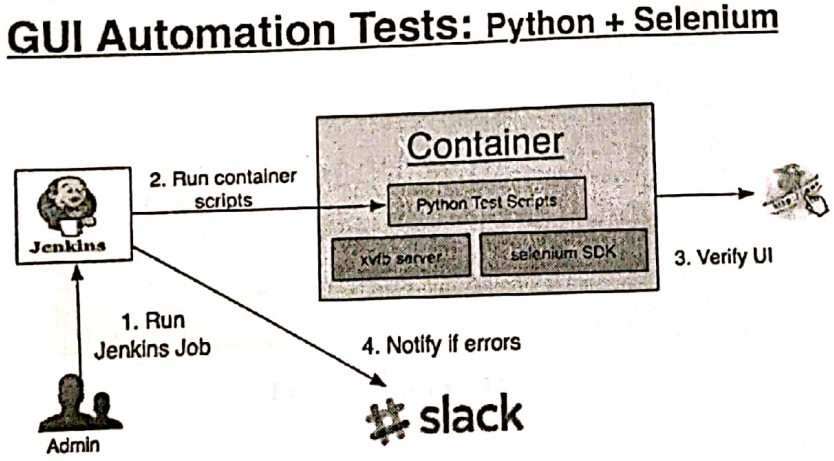

Fig 11- Selenium Python

We have used selenium to find the new features in Ironwood. These features are :

1. LMS to CMS login
2. Public Courses
3. Checklist in any specified course

MSExcel :

Microsoft Excel is a spreadsheet program included in the Microsoft Office suite of applications. Spreadsheets will provide you with the values arranged in rows and columns that can be changed mathematically using both basic and complex arithmetic operations. In addition to the standard spreadsheet features, Excel offers programming support via Microsoft's Visual Basic for Applications (VBA), the ability to access data from external sources via Microsoft's Dynamic Data Exchange (DDE). Microsoft Excel is an Electronic Spreadsheet Computer Program.

Shell Scripting :

Shell Scripting is an open-source operating system. Our Shell Scripting tutorial includes all topics of Scripting executing scripting, loops, scripting parameters, shift through parameters, sourcing, getopts, case, eval, let etc.

Ubuntu :

Ubuntu is an open source software operating system that runs from the desktop to the cloud, to all your internet connected things.

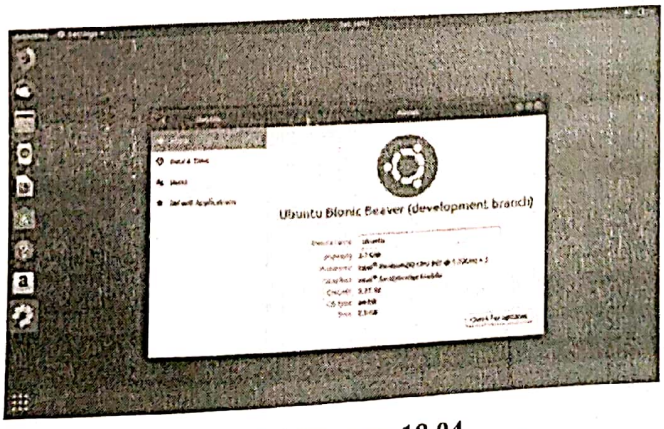

Fig 12- Ubuntu 18.04

List of Figures