**Task Checklist**

|  |  |
| --- | --- |
| **Contain** | **Page No.** |
| 1. Tasks on machines 2. Machine Requirements 3. Pre-requisites 4. Pre-Checking 5. Before Porting 6. Actual Porting on the Hawthorn Single Instance Machine 7. Actual Porting using ansible commands 8. After porting 9. Verifying the Migration 10. Dump and Restore   Link: <https://docs.google.com/document/d/1VoqHxcVuKoYp_x-Qd1GnK7JTN-RqRMfQQyBTOXL2sxE/edit>   1. Install the "Custom field for the registration page" Version.1    1. Remove old migration data from the MySQL database    2. Install the "Custom field for the registration page"   Link:  <https://docs.google.com/document/d/1YUfqAOB61gnxX-XV2oy7ncAoMqdLC5KB/edit#heading=h.gjdgxs>   * 1. Migrate the custom-field tables data from the Ginkgo release to the Ironwood release.  1. Appendix    1. Extra checking done in the last migration: | 1  2  3  3  4  9  13  14  16  18  20  20  23  24 |

# Tasks on machines:

**Task on machinesFig : *Tasks on machines***

Number represents the sequence of processes. The 6th step has two steps that can run parallely. Sequence of steps will change in the next round. It can be changed as per the distributed databases in the production environment.

# Machine Requirements:

Below are the machines required for the process.

* A **Ginkgo.2** machine with live production data dump. It must have the same configurations and settings as in the live production Gingko machine. Also, the **Custom Fields for the registration page** (Version.0) application must be installed in the Gingko.2 machine.
* A **Hawthorn.2** new machine with the same configurations and settings as it will be in the live production machine with the live data of the Ginkgo.2(MySQL and Mongo) machine. The **Custom fields for the registration page** application should not be installed.
* An **Ironwood.master** new machine with the same configurations and settings as it will be in the live production machine. The **Custom fields for the registration page** application should not be installed.
* A **remote machine** that has all the privileges to access the above machines and their databases(MySQL and Mongo)

# Pre-requisites:

All services of Open edX machines should stop.

Add the SECRET\_KEY in the configuration files vi /edx/app/edxapp/lms.auth.json, vi /edx/app/edxapp/cms.auth.json and vi my-passwords.yml etc.

# Pre-Checking:

Collect the credential of all distributed databases, and plan as per distributed databases.(Possibility of the distributed databases: Mysql databases are distributed on the two servers.)

|  |  |  |  |
| --- | --- | --- | --- |
| **About** | **IP** | **Username** | **Password** |
| **Hawthorn.2** | | | |
| MySQL all | 10.80.0.100 | mig | mig |
| ~~MySQL edxapp\_csmh~~ | ~~10.80.0.100~~ |  | ~~mig~~ |
| Mongodb | 10.80.0.100 | admin | password |
| ~~Ansible machine~~ | ~~10.80.0.100~~ |  |  |
| **Ironwood.master** | | | |
| MySQL all(exclude edxapp\_csmh) | 10.10.3.151 | mig | password |
| MySQL edxapp\_csmh | 10.10.3.153 | mig | password |
| Mongodb | 10.10.3.145 | admin | a01b@iitb. |
| ~~Ansible machine~~ | ~~10.~~ |  |  |

**Confirm Is the SECRET\_KEY added in the configuration files vi /edx/app/edxapp/lms.auth.json, vi /edx/app/edxapp/cms.auth.json and vi my-passwords.yml etc.?**

# Before Porting:

1. Clone the git repository on the remote machine:

***# Make a folder named migration on the remote machine***

***mkdir migration***

***cd migration***

***git clone http://saritat@gitlab.cse.iitb.ac.in/IITBombayX-Utilities/iitbombayxdbupgrade.git***

***cd iitbombayxdbupgrade***

***git checkout Ginkgo2ToIronwood***

1. Stop the services of the Hawthorn.2 and Ironwood.master machines and verify if all the services are stopped.

*# To stop the services*

***sudo /edx/bin/supervisorctl stop all***

*# To Confirm the status of the services*

***sudo /edx/bin/supervisorctl status all***

1. To check the MySQL database(s) privileges of the **old release** assigned to the remote machine:

***mysql -u<mysql\_user\_name> -p -h<mysql-ip-address>***

*# To display list of databases*

***show databases;***

***use information\_schema;***

*# To* Check the **character set** of all databases, all tables and all columns

# **Expecte**: all user’s databases, all user’s tables and all user’s columns should have utf8 character set.

*# To*Check the character set of all databases.

***select \* from SCHEMATA;***

*# To*Check the character set of all tables of all databases.

***SELECT count(\*), table\_collation, table\_schema***

***FROM information\_schema.`TABLES`***

***group by table\_collation, table\_schema***

***order by table\_schema;***

*# To*Check the character set of all columns of all tables of all databases.

***SELECT count(\*), character\_set\_name, TABLE\_NAME, table\_schema***

***FROM information\_schema.`COLUMNS`***

***group by character\_set\_name, TABLE\_NAME, table\_schema***

***order by table\_schema, TABLE\_NAME;***

*# To*Check No. of user tables present in each database.

***SELECT COUNT(\*), TABLE\_SCHEMA***

***FROM `information\_schema`.`TABLES`***

***WHERE TABLE\_SCHEMA***

***NOT IN ('mysql','information\_schema','performance\_schema')***

***group by TABLE\_SCHEMA ;***

***#*** *Exit from MySQL*

1. To check the mongodb database(s) privileges of the **old release** assigned to the remote machine:

***mongo -u <mongo\_user\_name> -p <mongo\_password> <new\_mongo\_ip-address>:27017/edxapp --authenticationDatabase admin***

***show dbs***

*# Users of the Mongo database*

***show users***

*# List of collections and indexes in the database*

***use <database\_name>***

*# List of collections*

***show collections***

*# Count of record and indexes in the collections* ***db.getCollectionNames().forEach(function(collname) {***

***var c=db[collname].count();***

***print("Number of documents in the",collname," collection:",c);***

***var indexes= db[collname].getIndexes();***

***var i=indexes.length;***

***print("Number of indexes on the",collname,"collection:",i);***

***})***

*# Users of the Mongo database*

***show users***

*#Exit from Mongo*

1. To check the MySQL database(s) privileges of the **new release** assigned to the remote machine:

***mysql -u<mysql\_user\_name> -p -h<mysql-ip-address>***

*# To display list of databases*

***show databases;***

***use information\_schema;***

*# To* Check the **character set** of all databases, all tables and all columns

# **Expecte**: all user’s databases, all user’s tables and all user’s columns should have utf8 character set.

*# To*Check the character set of all databases.

***select \* from SCHEMATA;***

*# To*Check the character set of all tables of all databases.

***SELECT count(\*), table\_collation, table\_schema***

***FROM information\_schema.`TABLES`***

***group by table\_collation, table\_schema***

***order by table\_schema;***

*# To*Check the character set of all columns of all tables of all databases.

***SELECT count(\*), character\_set\_name, TABLE\_NAME, table\_schema***

***FROM information\_schema.`COLUMNS`***

***group by character\_set\_name, TABLE\_NAME, table\_schema***

***order by table\_schema, TABLE\_NAME;***

*# To*Check No. of user tables present in each database.

***SELECT COUNT(\*), TABLE\_SCHEMA***

***FROM `information\_schema`.`TABLES`***

***WHERE TABLE\_SCHEMA***

***NOT IN ('mysql','information\_schema','performance\_schema')***

***group by TABLE\_SCHEMA ;***

***#*** *Exit from MySQL*

1. To check the mongodb database(s) privileges of the **new release** assigned to the remote machine:

***mongo -u <mongo\_user\_name> -p <mongo\_password> <new\_mongo\_ip-address>:27017/edxapp --authenticationDatabase admin***

***show dbs***

*# Users of the Mongo database*

***show users***

*# List of collections and indexes in the database*

***use <database\_name>***

*# List of collections*

***show collections***

*# Count of record and indexes in the collections* ***db.getCollectionNames().forEach(function(collname) {***

***var c=db[collname].count();***

***print("Number of documents in the",collname," collection:",c);***

***var indexes= db[collname].getIndexes();***

***var i=indexes.length;***

***print("Number of indexes on the",collname,"collection:",i);***

***})***

***#Exit from Mongo***

1. **Data Report Script:**
   1. Make copies “***Script/ReportScript”*** *according to the designed distributed architecture of the database server.*
      1. *Copy command*

***cp -r Script/ReportScript Script/ReportScript\_csmh***

***Note:*** Repeat following steps(b) for each copy of the ***ReportScript.***

* 1. Run the **data report script** for generating MySQL and Mongo database reports by following the steps below.
     1. *Change directory*

***cd Script/ReportScript<\_csmh>***

* + 1. *Set the parameters*

*# Set the parameters in* ***parameter.config*** *file and run the script as per the instructions given in README.md file,*

***vi parameter.config***

For the data report from **G**inkgo to **H**awthorn, set *UpgradingType="****GH****"*, result path, MySQL server detail(*IP, port and user credential*) and Mongo server detail(*IP, port and user credential*) of the old release machine as per the instruction in the ***parameter.config*** file.

For the **copy of ReportScript,** make sure the final **result path** of each server data report should be **unique(Change** a value of variable **mig\_Round="Round0/csmh")**.

After changing the parameters, save and exit.

* + 1. Run the **Data Report Script**

***bash data\_report.sh***

* + 1. Copy log of the **data report script.** Search for **“doesn't exist”,** if it is find reason; otherwise it is OK.

**Note: “doesn't exist” in the log** means that particular table doesn’t exist in the machine.

**----------------------------------what are you checking here ????**

1. Run the **Record Count Script** for generating MySQL and Mongo database reports by following the steps below.(**Repeat for each server, if required**)
   1. *Change directory*

***cd Script/RecordCountScript***

* 1. *Set the parameters*

*# Set the parameters in* ***parameter.config*** *file and run the script as per the instructions given in README.md file,*

***vi parameter.config***

For the record count report, set the result path, MySQL server detail(*IP, port and user credential*) and Mongo server detail(*IP, port and user credential*) of the old release machine as per the instruction in the ***parameter.config*** file.

After changing the parameters, save and exit.

* 1. Run the **Record Count Script,** it is used to form the observation in excel format.

**------------------ what does DataReport script do? I am confused with the RecordCount script and DataReport script? Please clarify**

**Yes, both are different. I need the output of RecordCount script for making report in excel format like**

**https://drive.google.com/file/d/1YqLFVzyEIbbXb3Be6AGhlOAXr75lh8j2/view?usp=sharing**

***bash count\_rows.sh***

# 6. Actual Porting on the Hawthorn Single Instance Machine

1. Verify the character set of all user databases and all user tables and all user columns. It should have an **“utf8”** character set. While upgrading the discovery database errors were received due to the character set and some missing tables in the discovery database on the Hawthon.2.

Discovery database has actually 103 tables of “utf8” character set for the Ginkgo release, but we found 96 tables with "latin1" character set in the production setup. It didn't have user data so we replaced it with a fresh discovery database for the Ginkgo.2 release.

The solutions mentioned above are explained below

Replace the discovery database:

*#To restore the* discovery *tables, Copy the file “iitbombayxdbupgrade/Dump/Fresh/Ginkgo2/G27/mysql-structure-data-discovery-20190906T113755.sql” from the git directory of the remote machine, to the Hawthorn machine and run following commands*

***scp -r <user\_remote\_machine>@<IP\_remote\_machine>:<path\_of\_discovery\_dump\_file> .***

***mysql -uroot -p < mysql-structure-data-discovery-20190906T113755.sql***

1. To migrate data from old release (Ginkgo) to new release (Hawthorn), we need to drop the database tables used by djcelery. These tables should be empty in the Ginkgo 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. While dropping the database tables used by djcelery using the management command, errors were received due to the foreign key constraint and some missing tables on the Hawthon.2 and Ironwood.master machine in the development environment. To avoid the errors, we tried a solution which is described below
   1. Restore all djcelery tables
   2. Set the foreign key constraint OFF of MySQL database.
   3. Run the management command provided by OpenedX to drop the database tables used by djcelery.
   4. Set the foreign key constraint ON of MySQL database.

The solutions mentioned above are explained below

1. Restore all the djcelery tables:

*#To restore all djcelery tables, Copy the file “iitbombayxdbupgrade/Dump/Data/celery\_tables.sql” from the git directory of the remote machine, to Gingko machine and run following commands*

***scp -r <user\_remote\_machine>@<IP\_remote\_machine>:<path\_of\_celery\_tables\_file> .***

***mysql -uroot -p < celery\_tables.sql***

The below steps are used for migrating the restore data for the Single-Instance machine in the development environment. For migrating to the live-production- data in a distributed instance in the production environment refer to the points mentioned at the end of this subsection.

1. Before dropping djcelery tables, set FOREIGN\_KEY\_CHECKS to OFF in MySQL.

***mysql -uroot -p***

***select @@global.FOREIGN\_KEY\_CHECKS;***

***select @@FOREIGN\_KEY\_CHECKS;***

***SET global FOREIGN\_KEY\_CHECKS=0;***

***SET FOREIGN\_KEY\_CHECKS=0;***

***select @@global.FOREIGN\_KEY\_CHECKS;***

***select @@FOREIGN\_KEY\_CHECKS;***

***exit***

1. 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***

1. Reset the foreign key constraint of MySQL database.

***mysql -uroot -p***

***SET global FOREIGN\_KEY\_CHECKS=1;***

***SET FOREIGN\_KEY\_CHECKS=1;***

***select @@global.FOREIGN\_KEY\_CHECKS;***

***select @@FOREIGN\_KEY\_CHECKS;***

***exit***

1. Verify the openedx version in ‘/var/tmp/configuration’ directory: output should be “**\* (HEAD detached at open-release/hawthorn.2)**”.
   1. Go to the directory ‘/var/tmp/configuration’

***cd /var/tmp/configuration***

* 1. Check the git branch

***git branch***

# *If it is not \* (HEAD detached at open-release/hawthorn.2), then*

***git checkout open-release/hawthorn.2***

* 1. Check ‘git diff’ in “/var/tmp/configuration" directory. There should be no difference.

***git diff***

* 1. Go to the home directory

***cd***

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

***lynx internet.iitb.ac.in***

1. Set permission to /edx/app/edx\_ansible/edx\_ansible/util/install/native.sh file.

# assign all permission to the file

***sudo chmod 777 /edx/app/edx\_ansible/edx\_ansible/util/install/native.sh***

1. Set an environment variable. it should be as git branch of “/var/tmp/configuration” git repository :

***export OPENEDX\_RELEASE=open-release/hawthorn.2***

# check by printing variable

***echo $OPENEDX\_RELEASE***

1. Set an environment variable **OPENEDX\_RELEASE=open-release/hawthorn.2** before line “CONFIGURATION\_VERSION=${CONFIGURATION\_VERSION-$OPENEDX\_RELEASE}” in /edx/app/edx\_ansible/edx\_ansible/util/install/native.sh file.

***sudo vi /edx/app/edx\_ansible/edx\_ansible/util/install/native.sh***

***OPENEDX\_RELEASE=open-release/hawthorn.2***

***echo ${OPENEDX\_RELEASE}***

***CONFIGURATION\_VERSION=${CONFIGURATION\_VERSION-$OPENEDX\_RELEASE}***

1. Run the new release(Hawthorn) migration script, which will upgrade old release(Ginkgo) data to be valid for new release(Hawthorn):

***/edx/app/edx\_ansible/edx\_ansible/util/install/native.sh --tags migrate***

It may ask for following :

* **Mail configuration**: Select **no configuration** option.
* **Configuring grub-pc**: Select **no configuration** option.
* **Some package updated version:** Select **currently-installed version** option.
* Password for edx user

# 

# 7. Actual Porting using ansible commands

Run the Ironwood migrations, which will update Hawthorn data to be valid for Ironwood. **Upgradation Process** for the **live-production-data** for the distributed instance in the production environment using ansible command. Ansible commands are run by the sysadmin team. Below steps to be followed.

1. Confirm the git branch of the repository used.
2. Enable logs in the “ansible” for analysis before the migration process starts.
3. Drop the database tables used by djcelery.
4. As per a single instance machine in the development environment, the sequence of tasks to upgrade data using the open edx “***/edx/app/edx\_ansible/edx\_ansible/util/install/native.sh***” script is given below and the same can be followed for a distributed environment.~~----------------- location of the file????~~

**For Ironwood Machine**

* 1. Edxapp (lms and cms)
  2. Ecommerce
  3. Analytics\_api
  4. Insights
  5. Edx\_django\_service
  6. Forum
  7. Xqueue

#### Sample Ansible commands

In the production environment, the actual migration was done by the sysad team. They had used the following commands(It is not exact copy of command that was run in the actual migration) on the “(edx\_ansible) edx@Iron-Ansible:/var/tmp/configuration/playbooks”

~~ansible-playbook ./edxapp.yml -u edx -i inventory.ini -k -K -c ssh --tags migrate~~

~~ansible-playbook ./ecommerce.yml -u edx -i inventory.ini -k -K -c ssh --tags migrate~~

~~ansible-playbook ./analyticsapi.yml -u edx -i inventory.ini -k -K -c ssh --tags migrate~~

~~ansible-playbook ./insights.yml -u edx -i inventory.ini -k -K -c ssh --tags migrate~~

~~ansible-playbook ./discovery.yml -u edx -i inventory.ini -k -K -c ssh --tags migrate~~

~~ansible-playbook ./forum.yml -u edx -i inventory.ini -k -K -c ssh --tags migrate~~

~~ansible-playbook ./xqueue.yml -u edx -i inventory.ini -k -K -c ssh --tags migrate~~

**Actual ansible commands:**

ansible-playbook -c ssh edxapp.yml -i inventory.ini --limit edxapp --tags "migrate" -e@server\_vars.yml -u edx -k -K -vvv

ansible-playbook -c ssh xqueue.yml -i inventory.ini --limit xqueue --tags "migrate" -e@server\_vars.yml -u edx -k -K

ansible-playbook -c ssh forum.yml -i inventory.ini --limit forum --tags "migrate" -e@server\_vars.yml -u edx -k -K

ansible-playbook -c ssh discovery.yml -i inventory.ini --limit discovery --tags "migrate" -e@server\_vars.yml -u edx -k -K

ansible-playbook -c ssh ecommerce.yml -i inventory.ini --limit ecommerce --tags "migrate" -e@server\_vars.yml -u edx -k -K

# 7. After porting

1. To check the MySQL database(s) privileges of the **new release** assigned to the remote machine:

***mysql -u<mysql\_user\_name> -p -h<mysql-ip-address>***

*# To display list of databases*

***show databases;***

***use information\_schema;***

*# To* Check the **character set** of all databases, all tables and all columns

# **Expecte**: all user’s databases, all user’s tables and all user’s columns should have utf8 character set.

*# To*Check the character set of all databases.

***select \* from SCHEMATA;***

*# To*Check the character set of all tables of all databases.

***SELECT count(\*), table\_collation, table\_schema***

***FROM information\_schema.`TABLES`***

***group by table\_collation, table\_schema***

***order by table\_schema;***

*# To*Check the character set of all columns of all tables of all databases.

***SELECT count(\*), character\_set\_name, TABLE\_NAME, table\_schema***

***FROM information\_schema.`COLUMNS`***

***group by character\_set\_name, TABLE\_NAME, table\_schema***

***order by table\_schema, TABLE\_NAME;***

*# To*Check No. of user tables present in each database.

***SELECT COUNT(\*), TABLE\_SCHEMA***

***FROM `information\_schema`.`TABLES`***

***WHERE TABLE\_SCHEMA***

***NOT IN ('mysql','information\_schema','performance\_schema')***

***group by TABLE\_SCHEMA ;***

***#*** *Exit from MySQL*

1. To check the mongodb database(s) privileges of the **new release** assigned to the remote machine:

***mongo -u <mongo\_user\_name> -p <mongo\_password> <new\_mongo\_ip-address>:27017/edxapp --authenticationDatabase admin***

***show dbs***

*# Users of the Mongo database*

***show users***

*# List of collections and indexes in the database*

***use <database\_name>***

*# List of collections*

***show collections***

*# Count of record and indexes in the collections* ***db.getCollectionNames().forEach(function(collname) {***

***var c=db[collname].count();***

***print("Number of documents in the",collname," collection:",c);***

***var indexes= db[collname].getIndexes();***

***var i=indexes.length;***

***print("Number of indexes on the",collname,"collection:",i);***

***})***

***#Exit from Mongo***

1. **Data Report Script:**
   1. Make copies “***Script/ReportScript”*** *according to the designed distributed architecture of the database server.*
      1. *Copy command*

***cp -r Script/ReportScript Script/ReportScript\_csmh***

***Note:*** Repeat following steps(b) for each copy of the ***ReportScript.***

* 1. Run the **data report script** for generating MySQL and Mongo database reports by following the steps below.
     1. *Change directory*

***cd Script/ReportScript<\_csmh>***

* + 1. *Set the parameters*

*# Set the parameters in* ***parameter.config*** *file and run the script as per the instructions given in README.md file,*

***vi parameter.config***

For the data report from **G**inkgo to **H**awthorn, set *UpgradingType="****GH****"*, result path, MySQL server detail(*IP, port and user credential*) and Mongo server detail(*IP, port and user credential*) of the old release machine as per the instruction in the ***parameter.config*** file.

For the **copy of ReportScript,** make sure the final **result path** of each server data report should be **unique(Change** a value of variable **mig\_Round="Round0/csmh")**.

After changing the parameters, save and exit.

* + 1. Run the **Data Report Script**

***bash data\_report.sh***

* + 1. Copy log of the **data report script.** Search for **“doesn't exist”,** if it exists find reason; otherwise it is OK.

1. Run the **Record Count Script** for generating MySQL and Mongo database reports by following the steps below.(**Repeat for each server, if required**)
   1. *Change directory*

***cd Script/RecordCountScript***

* 1. *Set the parameters*

*# Set the parameters in* ***parameter.config*** *file and run the script as per the instructions given in README.md file,*

***vi parameter.config***

For the record count report, set the result path, MySQL server detail(*IP, port and user credential*) and Mongo server detail(*IP, port and user credential*) of the old release machine as per the instruction in the ***parameter.config*** file.

After changing the parameters, save and exit.

* 1. Run the **Record Count Script**

***bash count\_rows.sh***

# 8. Verifying the Migration:

1. Copy Report Results: Go to the git repository directory copied in the remote machine. Check if the data report results(“ReportResult**GH/**Before” and “ReportResult**GH/**After”) are available in the result folder(It is set in parameter.config file) which was created in the section 4.1 or the section 4.3.

**Data Report Result:** Result will be stored on the "$rel\_result\_path/$mig\_Type/$mig\_Round/ReportResult$UpgradingType" path using the ***Scripts/ReportScript/parameter.config*** *file* variables value.

**Run script:** Run the **findDiff script** to get the difference between the database report results available in the “Before" folder and in the “After" folder.

***cd Scripts/ReportScript***

To compare the data report from **G**inkgo to **H**awthorn, set *UpgradingType="****GH****"*and result path as per the instruction in the ***parameter.config*** file.

# *Set the parameter in the parameter.config file as per instructions given in the "README.md" file,*

***vi parameter.config***

*Save and exit, and run the script.*

***bash findDiff.sh***

***Read the output log of the script.***

**Output:** After running this script, the “Conclusion” folder is created in the result path which is set in the “parameter.config” file. The Conclusion folder has the result of the comparison.

Extra: Steps 13 to 16 to know more.

1. To **Compare the schema** of MySQL database: Use software “Meld Diff Viewer”or the diff terminal command as

***diff -wy --suppress-common-lines <path\_of\_before\_schema> <path\_of\_after\_schema>***

.----- Was this software good? I remember, some issue was there

It is a graphical interface. it is slow as compared to the diff command. It works well for small files. The findDiff script covers it through the diff command. it is for us to confirm everything is fine.

1. Check for the files **Notdeleted\_tables**\_<database\_name>.txt if exist in the “../ReportResult**GH**/After/mysqlReports<\_ip\_time>/” result folder.
2. Check for the files new\_rows\_<database\_name>.txt if exist in the “../ReportResult**GH**/After/mysqlReports<\_ip\_time>/” result folder.
3. Note down the row count in **django\_migrations** table of edxapp\_csmh database on dedicated Ginkgo.2 Ported Machine for edxapp\_csmh.

# 9. Dump and Restore

1. To take dump of MySQL and Mongo Database and restore it use following link

Link: <https://docs.google.com/document/d/1VoqHxcVuKoYp_x-Qd1GnK7JTN-RqRMfQQyBTOXL2sxE/edit>

**10. Install the "Custom field for the registration page" Version.1**

1. **Remove old migration data from the MySQL database**
2. **Remove old migration** data from the MySQL database

*Go to the git repository directory copied on the remote machine*

***cd Script/Reg\_custom\_field\_data\_mig***

*# Set the parameters ( IP, MySQL database port and user credential of the Ironwood.master machine) in the parameter.config file as per instructions given in README.md file.*

***vi parameter.config***

*Save and exit.*

*#Run the script*

***bash remove\_G2\_custom\_reg\_data.sh***

1. **Install the "Custom field for the registration page"**
2. **Install the "Custom field for the registration page" Version.1** application on the Ironwood.master machine using “Custom fields for Registration page” Document.

Reference: [Instructions to Setup Custom Fields in IITBombayX for Ironwood.master version.1.docx](https://docs.google.com/document/d/1YUfqAOB61gnxX-XV2oy7ncAoMqdLC5KB/edit#heading=h.gjdgxs)

1. Clone the application from the git repository using the following command:

***git clone {git repository} {example AppDirectory}***

Use the following command:

***git clone http://<username>@gitlab.cse.iitb.ac.in/IITBombayXCustomRegForm/IITBomabayXCustomRegForm.git***

1. Go to the application directory

***cd IITBomabayXCustomRegForm/***

1. Checkout the git branch “*IITBombayX”* using the following command

***git checkout IITBomabayX***

1. Change the ownership of the application directory “*IITBomabayXCustomRegForm*” to the “*edxapp*” using the following command

***sudo chown -R edxapp:edxapp ../IITBomabayXCustomRegForm***

1. **Install the application:**
   1. Make sure the Internet connection is proper using the following command.

***lynx internet.iitb.ac.in***

* 1. Go to the directory “*/home/edx*”

Make sure the current working directory is “*/home/edx*” using the following command.

***Pwd***

If it is not “*/home/edx*”, change it to this directory using the following command.

***cd***

* 1. Change the user “edxapp” by executing the following command on the shell:

***sudo -H -u edxapp bash***

* 1. Activate edx platform virtual environment “*edxapp\_env” using the following command*

***source /edx/app/edxapp/edxapp\_env***

* 1. Now, Install this application using following command:

***pip install -e* IITBomabayXCustomRegForm*/***

* 1. Exit from the “*edxapp*” user shell using the following command:

***exit***

1. Open the “ **lms.env.json**” file located in the **“/edx/app/edxapp/”** directory (you may have to create it if it doesn't exist.)

***sudo vi /edx/app/edxapp/lms.env.json***

* 1. Add *"custom\_reg\_form"* in the "*ADDL\_INSTALLED\_APPS*" array. After adding, it should look like this

***"ADDL\_INSTALLED\_APPS": ["custom\_reg\_form"],***

* 1. Set a "*REGISTRATION\_EXTENSION\_FORM*" to the "*custom\_reg\_form.forms.ExtraInfoForm*". After setting the above parameter, it should look like this.

***"REGISTRATION\_EXTENSION\_FORM": "custom\_reg\_form.forms.ExtraInfoForm",***

* 1. Make sure the value of “***city****”* is “***hidden***” and “***state*” must be removed** ( should not be present) in the "*REGISTRATION\_EXTRA\_FIELDS*" dictionary in the “**lms.env.json**” file,. It should look like this

***"REGISTRATION\_EXTRA\_FIELDS": {***

***"first\_name": "required",***

***"city": "hidden",***

***"confirm\_email": "hidden",***

***"country": "required",***

***.***

***.***

***.***

***},***

* 1. Save and exit from the file.

1. In the same directory “**/edx/app/edxapp/**”, open the “**cms.env.json**” file (you may have to create it if it doesn't exist.)

***sudo vi /edx/app/edxapp/cms.env.json***

and set the “*ENABLE\_COMBINED\_LOGIN\_REGISTRATION*” feature flag to **“true**”. After setting the flag it should look like this.

***"ENABLE\_COMBINED\_LOGIN\_REGISTRATION": true,***

Save and exit from the file.

1. **Run migrations:**
   1. Make sure the Internet connection is proper using the following command.

***lynx internet.iitb.ac.in***

* 1. Go to the directory “*/home/edx*”

Make sure the current working directory is “*/home/edx*” using the following command.

***Pwd***

If it is not “*/home/edx*”, change it to this directory using the following command.

***cd***

* 1. Change the user *“edxapp*” by executing the following command on the shell:

***sudo -H -u edxapp bash***

* 1. Activate edx platform virtual environment “*edxapp\_env”* using the following command

***source /edx/app/edxapp/edxapp\_env***

* 1. Go to the “*edx-platform*” directory using following command:

***cd /edx/app/edxapp/edx-platform/***

* 1. Run Migrations using following command:

***./manage.py lms migrate custom\_reg\_form --settings=aws***

* 1. Exit from the “*edxapp*” user shell using the following command:

***exit***

1. **~~Migration process done. Start/restart~~** ~~the LMS service using the following command.~~

***~~sudo /edx/bin/supervisorctl restart lms~~***

1. **Migrate the custom-field tables data from the Ginkgo release to the Ironwood release**
2. **Migrate the custom-field tables** data from the Ginkgo release to the Ironwood release.

*# Go to the git repository directory copied on the* ***remote machine***

***cd Script/Reg\_custom\_field\_data\_mig***

# *Run the script*

***bash mig\_custom\_reg\_data\_GtoI.sh***

**11. Appendix**

1. **Extra checking done in the last migration:**

**Note:**It will not be required in this migration but keep for the reference.

USE edxapp\_csmh;

select count(\*) from django\_migrations;

# Expected count should be 296

select app, name, applied from django\_migrations order by app, name;

If unexpected diff found in diff\_diff\_UGC\_ecommerce\_auth\_permission.txt file as result of “findDiff.sh” script()diff is due to swapping in 131 and 128). To verify, Run following queries on MySQL of ported Ginkgo.2 machine:

USE ecommerce;

select \* from django\_content\_type where id=131 OR ID =128;

+-----+---------------+----------------+

| id | app\_label | model |

+-----+---------------+----------------+

| 128 | social\_django | partial |

| 131 | social\_django | usersocialauth |

USE edxapp\_csmh;

select count(\*) from django\_migrations;

# Expected count should be 296

1. select app, name, applied from django\_migrations order by app, name;

+-----+---------------+----------------+

If result is as above then it is OK.

1. “waffle\_switch” table of discovery database:

USE discovery;

select count(\*) from waffle\_switch;

select \* from waffle\_switch;

diff\_diff\_UGC\_analytics-api\_auth\_permission.txt: This database not consider.

diff\_diff\_UGC\_analytics-api\_django\_content\_type.txt: This database not consider.

diff\_diff\_UGC\_dashboard\_auth\_permission.txt: This database not consider.

diff\_diff\_UGC\_dashboard\_django\_content\_type.txt: This database not consider.

diff\_diff\_UGC\_dashboard\_django\_migrations.txt: This database not consider.

diff\_diff\_UGC\_ecommerce\_auth\_permission.txt:

"django\_content\_type" of ecommerce is master table of auth\_permission of ecommerce. There is change the swap the id if content which has id 131 and 128.Thus there are 6 changes in diff\_diff\_UGC\_ecommerce\_auth\_permission file, where 3 places swap 2 values.

diff\_diff\_UGC\_ecommerce\_django\_migrations.txt:

While Row count in django\_migrations table of ecommerce database: 198 is matched, but the sequence is change.

diff\_diff\_UGC\_edxapp\_csmh\_django\_migrations.txt:

diff\_diff\_upgraded\_rows\_analytics-api.txt: This database not consider.

diff\_diff\_upgraded\_rows\_dashboard.txt : This database not consider.

diff\_diff\_upgraded\_rows\_edxapp\_csmh.txt

.............................................................................

mysql> use discovery;

mysql> select \* from waffle\_switch;

+----+---------------------------------------+--------+------+----------------------------+----------------------------+

| id | name | active | note | created | modified |

+----+---------------------------------------+--------+------+----------------------------+----------------------------+

| 1 | use\_company\_name\_as\_utm\_source\_value | 0 | | 2018-02-07 08:22:45.087069 | 2018-02-07 08:22:45.087090 |

| 2 | threaded\_metadata\_write | 0 | | 2018-02-07 08:23:02.732896 | 2018-02-07 08:23:02.732910 |

| 3 | parallel\_refresh\_pipeline | 0 | | 2018-02-07 08:23:02.735020 | 2018-02-07 08:23:02.735029 |

| 4 | publish\_course\_runs\_to\_marketing\_site | 0 | | 2018-02-07 08:24:01.681434 | 2018-02-07 08:24:01.681455 |

| 5 | log\_course\_search\_queries | 0 | | 2018-02-07 08:24:03.942503 | 2018-02-07 08:24:03.942517 |

| 6 | auto\_course\_about\_page\_creation | 0 | | 2018-02-07 08:24:07.964722 | 2018-02-07 08:24:07.964738 |

| 7 | enable\_publisher\_email\_notifications | 0 | | 2018-02-07 08:26:02.153035 | 2018-02-07 08:26:02.153051 |

| 8 | publisher\_hide\_features\_for\_pilot | 0 | | 2018-02-07 08:26:21.523905 | 2018-02-07 08:26:21.523919 |

| 9 | publisher\_add\_instructor\_feature | 0 | | 2018-02-07 08:26:22.683585 | 2018-02-07 08:26:22.683601 |select \* from waffle\_switch;

| 10 | publisher\_comment\_widget\_featselect \* from waffle\_switch;ure | 0 | | 2018-02-07 08:26:30.890393 | 2018-02-07 08:26:30.890416 |

| 11 | publisher\_history\_widget\_feature | 0 | | 2018-02-07 08:26:31.108341 | 2018-02-07 08:26:31.108353 |

| 12 | publisher\_approval\_widget\_feature | 0 | | 2018-02-07 08:26:31.298306 | 2018-02-07 08:26:31.298319 |

| 13 | publish\_person\_to\_marketing\_site | 0 | | 2018-02-07 08:26:41.099838 | 2018-02-07 08:26:41.099898 |

+----+---------------------------------------+--------+------+----------------------------+----------------------------+

13 rows in set (0.03 sec)

on pRod Ginkgo.2 Migrated Machine 10.80.0.81:

waffle\_switch table contents of discovery database

select \* from waffle\_switch;

mysql> select \* from waffle\_switch;

+----+---------------------------------------+--------+------+----------------------------+----------------------------+

| id | name | active | note | created | modified |

+----+---------------------------------------+--------+------+----------------------------+----------------------------+

| 1 | threaded\_metadata\_write | 0 | | 2019-05-21 12:02:41.492430 |2019-05-21 12:02:41.492448 |

| 2 | parallel\_refresh\_pipeline | 0 | |2019-05-21 12:02:41.495565 | 2019-05-21 12:02:41.495575 |

| 3 |publish\_course\_runs\_to\_marketing\_site | 0 | | 2019-05-21 12:02:55.435871| 2019-05-21 12:02:55.435884 |

| 4 | log\_course\_search\_queries | 0 | |2019-05-21 12:02:56.153416 | 2019-05-21 12:02:56.153429 |

| 5 |enable\_publisher\_email\_notifications | 0 | | 2019-05-21 12:03:18.902812| 2019-05-21 12:03:18.902826 |

| 6 | publisher\_hide\_features\_for\_pilot |0 | | 2019-05-21 12:03:25.085765 | 2019-05-21 12:03:25.085782 |

| 7 |publisher\_add\_instructor\_feature | 0 | | 2019-05-21 12:03:25.725366 |2019-05-21 12:03:25.725380 |

| 8 | publisher\_comment\_widget\_feature | 0| | 2019-05-21 12:03:27.891832 | 2019-05-21 12:03:27.891864 |

| 9 |publisher\_history\_widget\_featurselect \* from waffle\_switch;e | 0 | | 2019-05-21 12:03:28.055026 |2019-05-21 12:03:28.055039 |

| 10 | publisher\_approval\_widget\_feature |0 | | 2019-05-21 12:03:28.344788 | 2019-05-21 12:03:28.344805 |

| 11 |publish\_person\_to\_marketing\_site | 0 | | 2019-05-21 12:03:33.728559 |2019-05-21 12:03:33.728574|

+----+---------------------------------------+--------+------+----------------------------+----------------------------+

11 rows in set (0.52 sec)

==========================================================================================================

dev

mysql> select \* from django\_content\_type where id=131 OR ID =128;

+-----+---------------+----------------+

| id | app\_label | model |

+-----+---------------+----------------+

| 128 | social\_django | usersocialauth |

| 131 | social\_django | partial |

+-----+---------------+----------------+

2 rows in set (0.00 sec)

.............................................................................

prod

mysql> USE ecommerce;select \* from django\_content\_type where id=131 OR ID =128;

Reading table information for completion of table and column names

You can turn off this feature to get a quicker startup with -A

Database changed

select \* from waffle\_switch;+-----+---------------+----------------+

| id | app\_label | model |

+-----+---------------+----------------+

| 128 | social\_django | partial |

| 131 | social\_django | usersocialauth |

+-----+---------------+----------------+

2 rows in set (0.00 sec)

mysql>

ie.Ok

chk the mongo conclusion folder:

if there are four files then finddiff.sh result ok for mongo

it there are two files the befor after same.

Task to do:

Note:

ReportScript available in branch Roun1Report is Final but that is not abailable in master branch.(To take schema of each database separately.

ADD NEW TABLE LIST IN DISCOVERY 96 WHY 73.

size diff of mongo files not zero condition will also handled as per diff\_upgraded\_rows\_\*

remove a size filter on outer level and apply in internal level