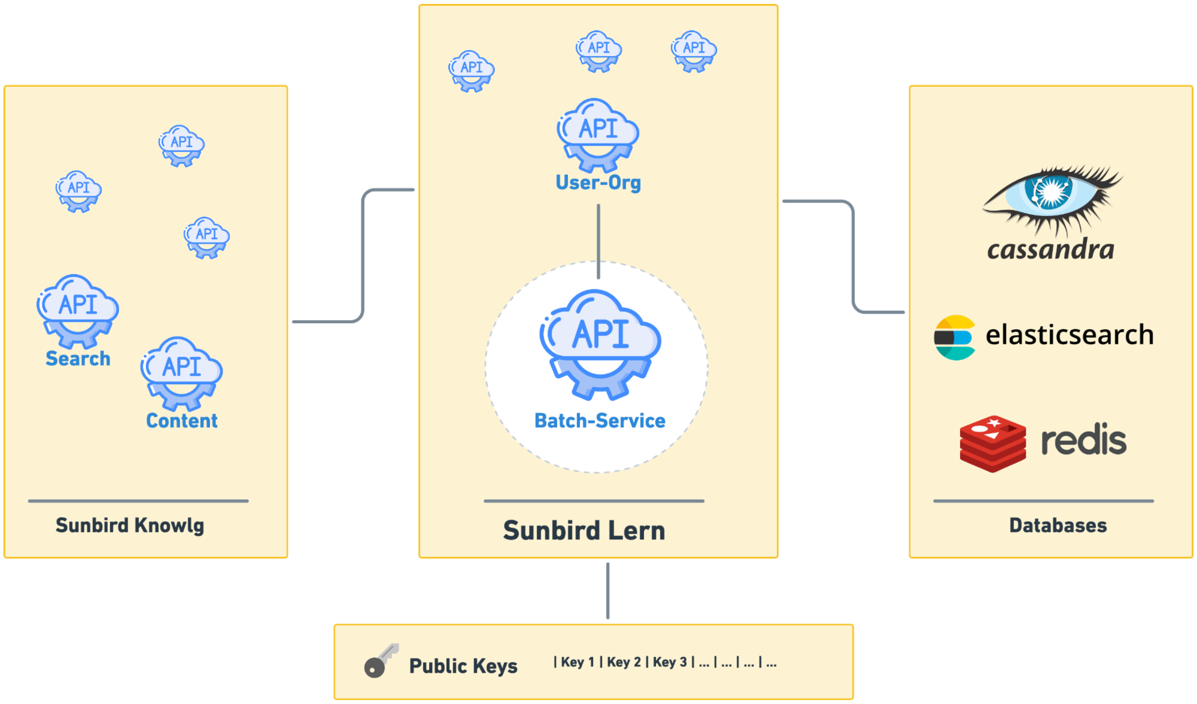
Sunbird ED Developer Bootcamp 2023 (Sunbird Lern)

Context:

This document helps in understanding the local setup, for one of the key microservice of the Sunbird Lern Building Block. This will enable the participants of the bootcamp to choose and work on the problem statements given below.

It has the details about setting up the **batch-service** from the scratch on your local machine.



Prerequisite

- Java 11
- Maven – Latest
- Docker – Latest
- Cassandra 3.11.6
- ES 6.8.11
- IntelliJ Editor
- Postman
- GIT

Validate Prerequisites

Service	Details to validate
Java 11	*java -version*
Maven	*mvn -v*
Docker	*Check the docker process or “ps -ef | grep docker”.*

Prepare your machine

Create a folder structure to organize the data

Command	*mkdir -p ~/sunbird-dbs/cassandra ~/sunbird-dbs/es* *export sunbird_dbs_path=~/sunbird-dbs*
Verification	*echo $sunbird_dbs_path*

Setup Cassandra

Pull Cassandra docker image	*docker pull cassandra:3.11.6*
Create a docker network	*docker network create sunbird_db_network*
Start Cassandra docker container	*docker run -p 9042:9042 --name sunbird_cassandra -v $sunbird_dbs_path/cassandra/data:/var/lib/cassandra -v $sunbird_dbs_path/cassandra/logs:/opt/cassandra/logs -v $sunbird_dbs_path/cassandra/backups:/mnt/backups --network sunbird_db_network -d cassandra:3.11.6*
Validate	*docker ps -a | grep cassandra*

Seed data to Cassandra

Provide permission to the sunbird-dbs/cassandra folder	*chmod -R 777 sunbird-dbs/cassandra*
Copy the sunbird_courses.cql file from github to /sunbird-dbs/cassandra/backups folder.	*wget -O sunbird-dbs/cassandra/backups/sunbird_courses.cql https://raw.githubusercontent.com/Sunbird-Lern/sunbird-course-service/bootcamp/scripts/sunbird_courses.cql*
Start the Cassandra cypher shell	*docker exec -it sunbird_cassandra cqlsh*
Load database schema	*source '/mnt/backups/sunbird_courses.cql'*

Setup Elasticsearch

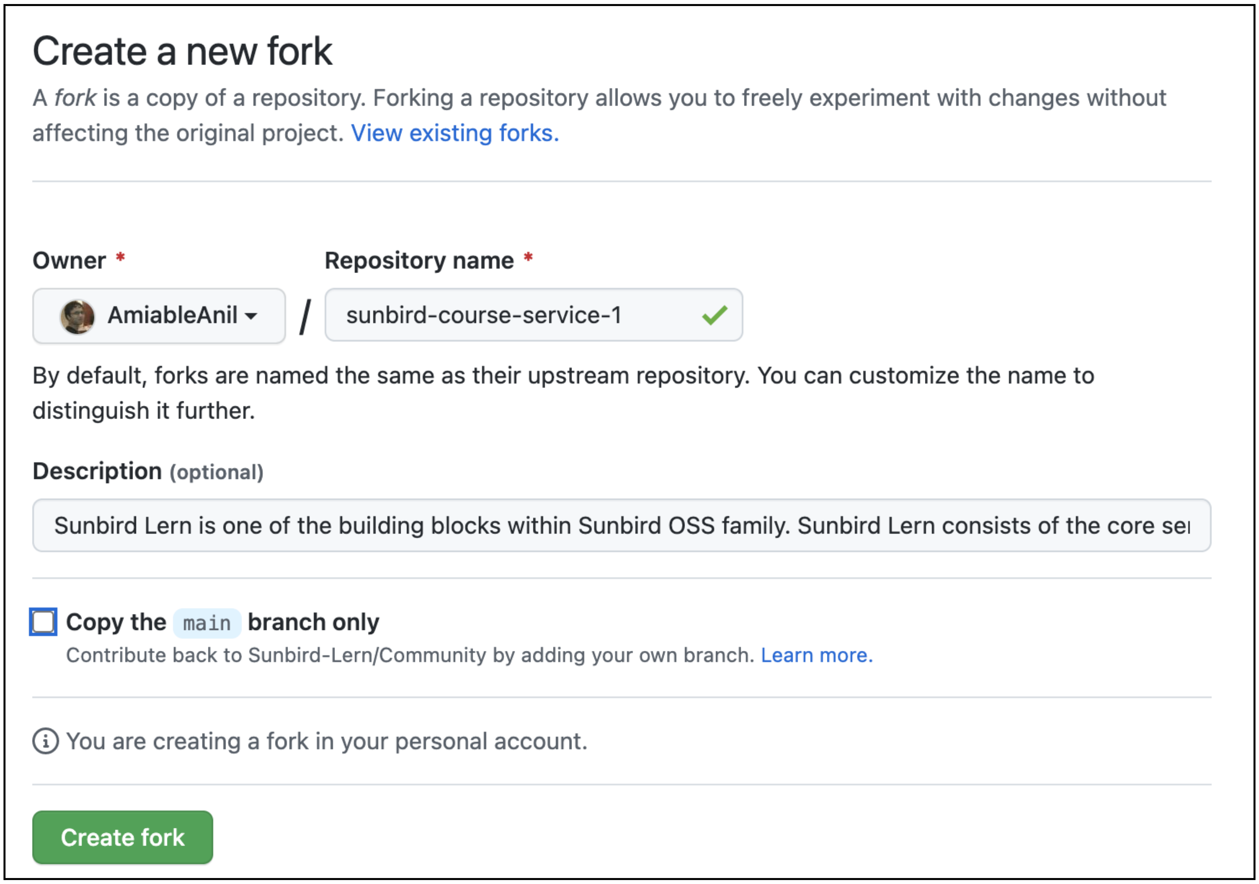
Pull Elasticsearch docker image	*docker pull elasticsearch:6.8.11*
Start Elasticsearch docker container	*docker run -p 9200:9200 --name sunbird_es -v $sunbird_dbs_path/es/data:/usr/share/elasticsearch/data -v $sunbird_dbs_path/es/logs://usr/share/elasticsearch/logs -v $sunbird_dbs_path/es/backups:/opt/elasticsearch/backup -e "discovery.type=single-node" --network sunbird_db_network -d docker.elastic.co/elasticsearch/elasticsearch:6.8.11*
Validate	*docker ps -a | grep es*
Change permissions for the folder	*chmod -R 777 sunbird-dbs/es*
Create “course-batch” index and update the mapping	*Bootcamp postman collection in the link below has the API collection and environment json files, import it in Postman and create index and mapping.* [*https://github.com/Sunbird-Lern/sunbird-course-service/tree/bootcamp/scripts/postman*](https://github.com/Sunbird-Lern/sunbird-course-service/tree/bootcamp/scripts/postman)

Setup Redis

Pull Redis docker image	*docker pull redis:4.0.0*
Start Redis docker container	*docker run --name sunbird_redis -d -p 6379:6379 redis:4.0.0*
Validate	*docker ps -a | grep redis*
SSH to Redis docker container	*docker exec -it sunbird_redis bash*

Batch Service Setup

Please fork the repository to use and make modifications. Below are the steps to fork and clone the repository.



Create a new fork

A *fork* is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. View existing forks.

Owner * **Repository name** *

AmiableAnil / sunbird-course-service-1 ✓

By default, forks are named the same as their upstream repository. You can customize the name to distinguish it further.

Description (optional)

Sunbird Lern is one of the building blocks within Sunbird OSS family. Sunbird Lern consists of the core se

□ **Copy the `main` branch only**

Contribute back to Sunbird-Lern/Community by adding your own branch. Learn more.

ⓘ You are creating a fork in your personal account.

Create fork

Repository	https://github.com/Sunbird-Lern/sunbird-course-service

🗒️ **Note:** Please uncheck the “Copy the main branch only”. This will copy all the branches to the fork.

Clone Repository	*git clone https://github.com/<YOUR_FORK>/sunbird-course-service.git*
Branch	*bootcamp*
Checkout branch	*cd sunbird-course-service* *git remote -v* *git remote remove origin* *git remote add origin git@github.com:reshmi-nair/sunbird-course-service.git* *git remote add upstream git@github.com:sunbird-lern/sunbird-course-service.git* *git checkout -b bootcamp* *git pull upstream bootcamp*
Export the configuration	*Update the **lms-service.sh** file in the **scripts** folder with configuration values to setup environment variables. Copy and run it to export the values*
Verify the configuration	*echo $sunbird_es_host*
Copy Keys for token verification	*Copy the keys folder from below link to local sunbird-course-services root folder* *https://drive.google.com/drive/u/0/folders/1V1CxHKP3IKMHg4s-Q_CSgvpu9bYdugNj*
Build the code base	*cd sunbird-course-service* *mvn clean install -DskipTests*

Run the service	*cd service* *mvn play2:run*
Check the setup using health API	*curl --location --request GET 'http://localhost:9000/health'*

Additional Details:

Please consider using the below user credentials for you to play with the APIs and experience the batch-service capabilities.

Group	Credentials and Other Details
Table – 1	**Creator:** bccreator_table1_mar2023/Password@1 **Reviewer:** bcreviewer_table1_mar2023/Password@1
Table – 2	**Creator:** bccreator_table2_mar2023/Password@1 **Reviewer:** bcreviewer_table2_mar2023/Password@1
Table – 3	**Creator:** bccreator_table3_mar2023/Password@1 **Reviewer:** bcreviewer_table3_mar2023/Password@1
Table – 4	**Creator:** bccreator_table4_mar2023/Password@1 **Reviewer:** bcreviewer_table4_mar2023/Password@1
Table – 5	**Creator:** bccreator_table5_mar2023/Password@1 **Reviewer:** bcreviewer_table5_mar2023/Password@1
Table – 6	**Creator:** bccreator_table6_mar2023/Password@1 **Reviewer:** bcreviewer_table6_mar2023/Password@1
Table – 7	**Creator:** bccreator_table7_mar2023/Password@1 **Reviewer:** bcreviewer_table7_mar2023/Password@1
Table – 8	**Creator:** bccreator_table8_mar2023/Password@1 **Reviewer:** bcreviewer_table8_mar2023/Password@1
Table – 9	**Creator:** bccreator_table9_mar2023/Password@1 **Reviewer:** bcreviewer_table9_mar2023/Password@1
Common Credentials	bcuser1_mar2023/Password@1 bcuser2_mar2023/Password@1

	bcuser3_mar2023/Password@1 bcuser4_mar2023/Password@1 bcuser5_mar2023/Password@1 bcuser6_mar2023/Password@1 bcuser7_mar2023/Password@1 bcuser8_mar2023/Password@1 bcuser9_mar2023/Password@1 bcuser10_mar2023/Password@1

Problem Statements:

Use case 1: LR-374 Compute the number of days to start the batch

Each batch has the metadata to explain the start date, end date, enrolment end date. Please introduce a new property in the batch read API to explain **how many days are left for the users to** consume the course.

- Compute the duration in days.
- The new property name should be “daysToGo”
- The value should be a string.
- If the start date is a future date, “x days to go”.
- If the start date is today, “the batch started today”.
- If the start date is old, “started x days ago”.

Use case 2: LR-375 Limit enrolments for the batch

There is no limitation on the number of enrolments in a batch. Enhance the enrolment feature to control the number of enrolments.

- Define the limit for the batch
  - Option 1: Create a default configuration to control the number of enrolments for the batches.
  - Option 2: Create a new attribute (update the batch table) and accept the limit for each batch as part of create or update API.
- Compute the number of enrolments and control.
  - Each successful enrolment should update the predefined counter.
  - Use the predefined counter and the configuration from the first step to limit the enrolments.
- Enrolment API response.
  - When the enrolment limit exceeds, the API should return HTTP 406 Not Acceptable as response with proper error details.

Use case 3: LR-376 Control enrolment for a specific type of users

For a batch, any user of the sunbird instance can enrol and consume the course. Please enhance the enrolment capability to allow users with specific role or tagging.

The course has “audience” attribute to understand the targeted users. Please consider using the value of the audience (teacher) and map it with the user's metadata (userType) to allow the enrolment.

- Accept the enrolment if the audience (from the course) and userType from the enrolling user metadata matches.
- Use the auth token to extract the user identifier.
- Make the user read API call to fetch the metadata of the user.
- If the enrolment is for an invalid user, the API should return an error code with a message.