**Environment Setup:**

**Cloning Repository:**

To clone repository please follow these steps

* git clone <https://gitlab.os/hranalytics/Tara.git>
* git config --global http.postBuffer 524288000
* git config --global http.sslVerify false
* git checkout dev
* git pull origin dev

**Compile and Build:**

* mvn clean install

**PIO event server commands:**

* **To create an app:**

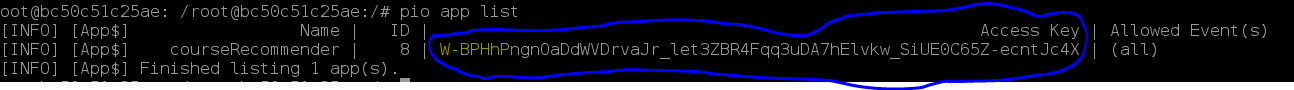
pio app new nameOfApp

* **To delete an app:**

pio app delete nameOfApp

* **To list all available app**:

pio app list (sample output is following)



This key will be used in future for ingesting anything into event server against this app. (This key is the reference of this app)

* **To clear all the data present in this application:**

pio app data-delete nameOfApp

**Configuring Universal Recommender:**

Add all the events in the engine.json against eventNames field.

Events supported right now are following

* courseTaken
* memberOf
* ownProject
* relatedTo

Increase "spark.executor.memory" to "6g". Configured engine can be seen in Tara/universalRecommender/engine.json

**Mysql Setup:**

* step1
* cd Tara/
* git pull origin dev
* mvn clean install
* mysql
* create database courserecommender;
* use courserecommender;
* DROP TABLE IF EXISTS `recommendations`;
* CREATE TABLE `recommendations` (

`userId` varchar(500) NOT NULL,

`courseId` varchar(500) NOT NULL,

`date` bigint(20) NOT NULL,

`taken` tinyint(1) NOT NULL,

`recommendationId` int(11) NOT NULL AUTO\_INCREMENT,

`feedback` varchar(500) DEFAULT NULL,

`rating` float DEFAULT NULL,

PRIMARY KEY (`recommendationId`)

) ENGINE=MyISAM AUTO\_INCREMENT=86 DEFAULT CHARSET=latin1;

* DROP TABLE IF EXISTS `courses`;
* CREATE TABLE `courses` (

`title` varchar(1000) NOT NULL,

`currentId` varchar(10000) DEFAULT NULL,

`allIds` varchar(10000) DEFAULT NULL,

`updateCount` int(11) NOT NULL,

PRIMARY KEY (`title`)

) ENGINE=MyISAM DEFAULT CHARSET=latin1;

**Tomcat Setups:**

**Cors setting for Tomcat:**

Add the following lines in **$TOMCAT\_HOME/conf/web.xml**

<filter>

<filter-name>CorsFilter</filter-name>

<filter-class>org.apache.catalina.filters.CorsFilter</filter-class>

<init-param>

<param-name>cors.allowed.origins</param-name>

<param-value>\*</param-value>

</init-param>

<init-param>

<param-name>cors.allowed.methods</param-name>

<param-value>GET,POST,HEAD,OPTIONS,PUT</param-value>

</init-param>

<init-param>

<param-name>cors.allowed.headers</param-name>

<param-value>Content-Type,X-Requested-With,accept,Origin,Access-Control-Request-Method,Access-Control-Request-Headers</param-value>

</init-param>

<init-param>

<param-name>cors.exposed.headers</param-name>

<param-value>Access-Control-Allow-Origin,Access-Control-Allow-Credentials</param-value>

</init-param>

<init-param>

<param-name>cors.support.credentials</param-name>

<param-value>true</param-value>

</init-param>

<init-param>

<param-name>cors.preflight.maxage</param-name>

<param-value>10</param-value>

</init-param>

</filter>

<filter-mapping>

<filter-name>CorsFilter</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

**Web Services Deployment:**

* cp /Tara/we-services/target/TaraServices.war $TOMCAT\_HOME/webapps/
* cp /Tara/email.sh $TOMCAT\_HOME/bin/
* cp /Tara/rest.sh $TOMCAT\_HOME/bin/
* start tomcat

**Training of the model:**

* cd Tara/
* pio app data-delete courseRecommender (clear the application)
* bash preparer.sh
* bash ingestor.sh
* cd universalRecommender
* pio build
* pio train (if fails then try to free some ram using the command “sudo echo 3 > /proc/sys/vm/drop\_caches”)
* pio deploy &

Here are the details of the contents of ingestor.sh

**Amazon Ingestor:**

java -cp /Tara/InputValidator/target/InputValidator-0.0.1-SNAPSHOT-jar-with-dependencies.jar com.tara.dataIngestor.AmazonReportIngestor http://localhost:7070 W-BPHhPngnOaDdWVDrvaJr\_let3ZBR4Fqq3uDA7hElvkw\_SiUE0C65Z-ecntJc4X amazon "\"\n\"" "\",\"" 37 null @# /temp/courses 60 true <http://172.30.89.210:8081>

**Arguments Details:**

* appurl = args[0]; //url of event store
* accessKey = args[1]; //app\_id returned from pio app list
* inputFile = args[2]; //path of input file
* lineSplitter = args[3]; //line separator
* recordSplitter = args[4]; //record separator
* inputLength = args[5]; //length of input records, any record with less/more than this count will be treated as malformed
* oldDataPath = args[6]; //path of already ingested courses, null if ingesting first time
* oldDataSplitter = args[7]; //splitter to use for temp courses to be saved
* newDataPath = args[8]; //path where the new records of courses will be saved so that next time these records will be used as oldDataPath, ideally its value should be equal to oldDataPath
* fuzzyScoreThreshold = args[9]; //threshold for fuzzy matcher
* updateDb = args[10]; //value should always be true ..it was added for testing
* tomcatPrefix = args[11]; //prefix of tomcat for api calling

**Historic Ingestion:**

This above command do the historic ingestion for courses and stores the ingested courses at path=”/temp/courses”. The format of file is id oldDataSplitter title. In the above example sample record of this file will be “ITC@#intro to computer science”.

**UserInfoPlaza Ingestor:**

java -cp /Tara/InputValidator/target/InputValidator-0.0.1-SNAPSHOT-jar-with-dependencies.jar com.tara.dataIngestor.UipIngestor http://localhost:7070 W-BPHhPngnOaDdWVDrvaJr\_let3ZBR4Fqq3uDA7hElvkw\_SiUE0C65Z-ecntJc4X uip "\",\"" 11

**Arguments Details:**

* appurl = args[0];
* accessKey = args[1];
* inputFile = args[2];
* recordSplitter = args[3];
* inputLength = args[4];

**Projects Ingestor:**

java -cp /Tara/InputValidator/target/InputValidator-0.0.1-SNAPSHOT-jar-with-dependencies.jar com.tara.dataIngestor.ProjectIngestor http://localhost:7070 W-BPHhPngnOaDdWVDrvaJr\_let3ZBR4Fqq3uDA7hElvkw\_SiUE0C65Z-ecntJc4X proj "\",\"" 7

**Arguments Details:**

* appurl = args[0];
* accessKey = args[1];
* inputFile = args[2];
* fileSplitter = args[3];

**User relations ingestor:**

java -cp /Tara/InputValidator/target/InputValidator-0.0.1-SNAPSHOT-jar-with-dependencies.jar com.tara.dataIngestor.UserRelationIngestor http://localhost:7070 W-BPHhPngnOaDdWVDrvaJr\_let3ZBR4Fqq3uDA7hElvkw\_SiUE0C65Z-ecntJc4X ur "\",\"" 6

**Arguments Details:**

* appurl = args[0];
* accessKey = args[1];
* inputFile = args[2];
* fileSplitter = args[3];

**Social Group Ingestor:**

java -cp /Tara/InputValidator/target/InputValidator-0.0.1-SNAPSHOT-jar-with-dependencies.jar com.tara.dataIngestor.SocialGroupIngestor http://localhost:7070 W-BPHhPngnOaDdWVDrvaJr\_let3ZBR4Fqq3uDA7hElvkw\_SiUE0C65Z-ecntJc4X sg "\",\""

**Arguments Details:**

* appurl = args[0];
* accessKey = args[1];
* inputFile = args[2];
* fileSplitter = args[3];

**Group Members Ingestor:**

java -cp /Tara/InputValidator/target/InputValidator-0.0.1-SNAPSHOT-jar-with-dependencies.jar com.tara.dataIngestor.GroupMemberIngestor http://localhost:7070 W-BPHhPngnOaDdWVDrvaJr\_let3ZBR4Fqq3uDA7hElvkw\_SiUE0C65Z-ecntJc4X gm "\",\""

**Arguments Details:**

* appurl = args[0];
* accessKey = args[1];
* inputFile = args[2];
* fileSplitter = args[3];

**Incremental Loading of Data:**

* cd /Tara
* bash incrementalIngestor.sh incrementalAmazon(name of input file)
* pio build
* pio train
* pio deploy &

**Course Update Logic:**

For course update logic we have implemented fuzzyMatcher (com.tara.common.FuzzyHelper) and if title of new course matches with any previously ingested course then we generate a courseUpdate API call to update course detail in mysql database. Here is a flow of course update.

New course(intro to cs,ITC) added into database the record will be like

|  |  |  |  |
| --- | --- | --- | --- |
| title | currentId | allIds | updateCount |
| intro to cs | ITC | ITC | 0 |

If a course (intro to cs\_new,ITC\_NEW) is added because this course’s title is passing 60% threshold with previously added course so now course update request will be generated and the db after ingestion will be

|  |  |  |  |
| --- | --- | --- | --- |
| title | currentId | allIds | updateCount |
| intro to cs\_new | ITC\_NEW | ITC\_NEW,ITC | 1 |

Whenever recommendation Taken request will be generated then all the recommendations for all matching Ids will be updated. I you want to implement your own implementation then you will need to modify this class **com.tara.common.FuzzyHelper** which implements interface **com.tara.common.FuzzyMatcher**.

**API Documentation:**

Use the following APIs

API Endpoint: <https://octopus.orangesv.us/tara/TaraServices/rest/tara/>

APIs:

1. GET generateRecommendations/{employeeID}/{count}

**ExampleCall:**

<https://octopus.orangesv.us/tara/TaraServices/rest/tara/generateRecommendations/ZQCP1460/4>

1. GET sendEmail/{employeeID}/{count}

**ExampleCall:**

<https://octopus.orangesv.us/tara/TaraServices/rest/tara/sendEmail/ZQCP1460/4>

1. GET insertRecommendation/{userId}/{courseId}

**ExampleCall:**

<https://octopus.orangesv.us/tara/TaraServices/rest/tara/insertRecommendation/ZQCP1460/DSLBL2CT-1214-PostTest>

1. GET insertCourse/{id}/{title}

**ExampleCall:**

<https://octopus.orangesv.us/tara/TaraServices/rest/tara/insertCourse/DSLBL2CT-1214-PostTest/intro%20to%20cs>

1. GET updateCourse/{id}/{oldTitle}/{newTitle}

**ExampleCall:**

<https://octopus.orangesv.us/tara/TaraServices/rest/tara/updateCourse/DSLBL2CT-1214-PostTest_NEW/intro%20to%20cs/intro%20to%20com%20sci>

1. GET markRecommendationTaken/{userId}/{courseId}

**ExampleCall:**

<https://octopus.orangesv.us/tara/TaraServices/rest/tara/markRecommendationTaken/ZQCP1460/DSLBL2CT-1214-PostTest>

1. GET getRecommendations/{startDate}/{endDate}/{includeDateInFilter}/{state}

ExampleCall:

<https://octopus.orangesv.us/tara/TaraServices/rest/tara/getRecommendations/0/0/false/TAKEN>

<https://octopus.orangesv.us/tara/TaraServices/rest/tara/getRecommendations/0/0/false/ALL>

1. GET getRecommendationsByUser/{startDate}/{endDate}/{includeDate}/{state}/{count}

**ExampleCall:**

[**https://octopus.orangesv.us/tara/TaraServices/rest/tara/getRecommendationsByUser/0/0/false/TAKEN/10**](https://octopus.orangesv.us/tara/TaraServices/rest/tara/getRecommendationsByUser/0/0/false/TAKEN/10)

1. GET

getDetailedRecommendationsByUser/{startDate}/{endDate}/{includeDate}/{state}/{count}

**ExampleCall:**

<https://octopus.orangesv.us/tara/TaraServices/rest/tara/getDetailedRecommendationsByUser/0/0/false/ALL/10>

**Dashboard:**

To see the dashboard please go inside the cloned repo folder and open the page “Tara/startbootstrap-sb-admin-2-gh-pages/pages/index.html” in web browser.