Import DB to local system:

Enable mysql terminal:

$ cd /usr/local/mysql/bin

$ ./mysql -u root -p

(then enter your mysql pswd, ex: root)

rergerg—

Import DB:

mysql> use KPLMS\_CLOUD

mysql> source /Users/getit/Documents/KPGerritHub/phase2dev/KPLMS\_CLOUD.sql

(source <.sql file path>)

Screen Name: Company registration

URL path: /cloudadmin/register

HTML file: registrationCompany.html

js controller file: companyRegisterController.js

APIs -

1. To get Company’s Industry List:

We use this API to predict the Industry type for the user input field in Company registration screen

in KPConsumer -

js function name: getCompanyIndustryListDetails()

constantService.js URL Constant: getCompanyIndustryList:'/kp/industry/list'

java controller file: IndustryController.java - GET\_INDUSTRY\_LIST\_WITHOUT\_LOGIN

in KPApi -

API URL: http://localhost:8080/KPApi/industry/get/list

java controller file: IndustryController.java – GET\_INDUSTRY\_LIST

in DB -

Tables Used: kp\_industry

2. To Check Organization name exists:

We use this API to check if there are any organization already exists with the same name in DB

in KPConsumer -

js function name: checkOrgExist()

constantService.js URL Constant: checkOrgExit :'/kp/org/checkexistedorg'

java controller file: KPOrganizationController.java - CHECK\_EXISTED\_ORG

in KPApi -

API URL: http://localhost:8080/KPApi/org/check/existedorg

java controller file: KPOrganizationController.java – CHECK\_EXISTED\_ORG

in DB -

Tables Used: kp\_organization

Screen Name: Course Details

URL path: /pod/details

HTML file:

js controller file: cloudUserCourseDetailsControllers.js

Sub-module: Participants List

HTML file: participantList.html

js file: cloudUserCourseDetailsControllers.js

1. To get participants list and Ongoing Course Progress:

We use this API to get list of participants of ongoing course and their progress

in KPConsumer -

js function name: getOngoingpodsParticipantList

js file : podService.js

constantService.js URL Constant: fetchEnrolledUserOngoingPods:'/kp/advance/get/enrolled/user/ongoing/'

java controller file: CloudUserAdvanceController.java - GET\_ONGOING\_USER\_LIST

in KPApi -

API URL:

http://localhost:8080/KPApi/advanceuser/get/enrolled/user/ongoing/579/3/5/1? sortBy=asc&fname=&mname=&lname=

java controller file: CloudUserAdvanceController.java – GET\_ONGOING\_USER\_LIST

in DB -

Tables Used: kp\_user, kp\_r\_org\_site\_course\_counter

2. To save Course Progress On completion of each chapter:

We use this API to save course progress in DB

in KPConsumer -

js function name: nextChapter()

js file : classroomService.js

constantService.js URL Constant: fetchEnrolledUserOngoingPods:'/kp/advance/get/enrolled/user/ongoing/'

java controller file: CloudUserCourseController.java - ADD\_ENROLLED\_POD\_PROGRESS\_DETAILS

in KPApi -

API URL:

http://localhost:8080/KPApi/

java controller file: CloudUserCourseController.java – ADD\_ENROLLED\_POD\_PROGRESS\_DETAILS

in DB -

Tables Used: kp\_user, kp\_r\_org\_site\_course\_counter

steps:

1. check chapter is already completed - isChapterCompleted(long userId, long courseId, long contentId) – kp\_course\_progress\_details

2. if no records found in kp\_course\_progress\_details table -

2.1 add course progress details – kp\_course\_progress\_details

2.1.1 addCourseProgressPercentage

2.1.1.1 check isOrgCourseAutoCertified – kp\_org\_course

3. Fetch User List to enroll to course:

We use this API to get list of eligible participants to enroll to the course

in KPConsumer -

HTML file: cloudUserViewCourseParticipateList.html

js controller file: enrollUserController.js

js function name: fetchEnrollUserList()

constantService.js URL Constant: getUserListToEnroll : '/kp/enrolltocourse/get/user/toenroll/',

java controller file: CourseEnrollmentController.java - GET\_USER\_TO\_ENROLL

in KPApi -

API URL: http://localhost:8080/KPApi/enrolltocourse/get/user/toenroll/594/3/5/1

java controller file: CourseEnrollmentController.java – GET\_ONGOING\_USER\_LIST

in DB -

Tables Used: kp\_user

3. Login:

Login to KP

in KPConsumer -

HTML file: login.html

js controller file: loginController.js

js function name: loginSubmit()

constantService.js URL Constant: getUserListToEnroll : '/kp/enrolltocourse/get/user/toenroll/',

java controller file: CourseEnrollmentController.java - GET\_USER\_TO\_ENROLL

in KPApi -

API URL: http://localhost:8080/KPApi/enrolltocourse/get/user/toenroll/594/3/5/1

java controller file: CourseEnrollmentController.java – GET\_ONGOING\_USER\_LIST

in DB -

Tables Used: kp\_user

3. Pod progress details:/pod/progress

Login to KP

in KPConsumer -

HTML file: userProgressDetails.html

js controller file: userProgressDetailsController.js

js function name: fetchUserProgressDetails()

constantService.js URL Constant: fetchUserProgress : '/kp/usercourse/get/progress/percent/',

java controller file: CloudUserCourseController.java – GET\_USER\_PROGRESS\_OF\_COURSE

in KPApi -

API URL: http://localhost:8080/KPApi/enrolltocourse/get/user/toenroll/594/3/5/1

java controller file: CloudUserCourseController.java – GET\_USER\_PROGRESS\_OF\_COURSE

in DB -

Tables Used:

Pod Progress Details -

Ongoing / Completed Pods

Course Payment:

To display Payment info before pay

in KPConsumer -

HTML file: paymentIntegration.html

js controller file: paymentController.js

methods -

1. fetchPaymentInfo()

js function name: loginSubmit()

constantService.js URL Constant: getUserListToEnroll : '/kp/enrolltocourse/get/user/toenroll/',

java controller file: CourseEnrollmentController.java - GET\_USER\_TO\_ENROLL

in KPApi -

API URL: http://localhost:8080/KPApi/enrolltocourse/get/user/toenroll/594/3/5/1

java controller file: CourseEnrollmentController.java – GET\_ONGOING\_USER\_LIST

in DB -

Tables Used: kp\_user

Pay now:

paymentController: requestPayment()

paymentService: requestPayment()

requestPayment : '/kp/payment/request'

PaymentController: REQUEST\_PAYMENT

0243104000169073