Vedantu: Institute System Integration APIs

Vedantu Innovations Pvt. Ltd.

16th December, 2013

Version 1.3

Last updated: 9th January, 2014

TABLE OF CONTENTS

Introduction

Vedantu is a self-sustained platform that allows an Institute to manage their organization's:

1. Academic structure
2. People (Students, Teachers, Admins etc.)
3. Courses
4. Content
5. Devices

from its web-applications.

For Institutes that have their own User Management Systems Vedantu supports integration for:

1. User Registration
2. User Authentication
3. Enroll User to Class
4. Usage Data Upload

Institute EndPoint URL Registration

In order to enable an integration with the Institute's User Management System it must register the corresponding EndPoint URL with the Vedantu system. This can be done at the time of submitting Institute creation request or subsequently using the Vedantu Admin Tool.

Note:

1. All communication will be done over secure HTTP (HTTPS). So, the Institute EndPoint must be a HTTPS URL.
2. There are suggestive URLs specified for various APIs. The Institute may override the URL to something of their choice, including the name of the API.
   eg. [https://<institute-endpoint-url>/instiAuth] instead of the suggested
   [https://<institute-endpoint-url>**/authenticate**]
3. The Institute may specify set of pre-defined headers or parameters (key-value pairs) to be passed with each request. This may be required for validating the authenticity of the API call at their end.

Institute System based User Registration Process

Refer the following figure to understand the call sequence for the User Registration process.

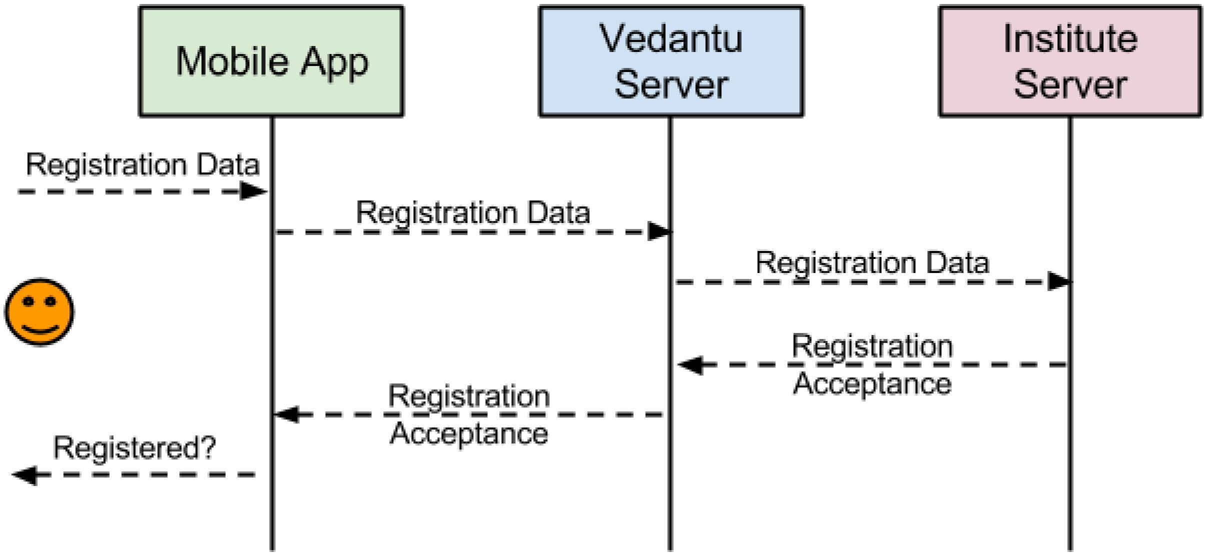

Figure 1. Sequence Diagram for User Registration Process

Note: Vedantu Server simply acts as a proxy for the User Registration Process. It does not log or store the user credentials in its database. However, there might be supplementary registration data (like class mappings) that Vedantu might have to store in its Server or Application databases.

Note: At present the user registration is supported only for STUDENT role.

User Registration API

https://<institute-endpoint-url>**/register**

Registers a new User.

HTTP Method	POST

Parameters

Parameter	Example	Description

username	anthonyg	**required** Username to be used for registration and future authentication against the Institute’s User Management System
password	somesecret	**required** Password of the user
email	anthony.gonsalves@bollywood.com	**optional** If the user provides her email for future communication. This might or might not be the same as username.
firstName	Anthony	**required** First name of this user. Required for profile display.
lastName	Gonsalves	**optional** Last name of this user. Required for profile display.
gender	MALE	**optional** *Enum: [MALE, FEMALE, UNKNOWN]* Gender of this user. Required for generic profile-pic display.
role	STUDENT	**required** *Enum: [STUDENT, TEACHER, ADMIN]* Role of the member in the Institute.
additionalInfo	{ “CAT Roll Number” : “CAT-2013-67” , “Phone Number” : “9876543210”, “City” : “Vasco Da Gama”, “some other field 1” : “some value 1”, ... }	**optional** This field contains the extra information needed for user registration in the Institute’s User Management System. The Institute can define the field it needs in the Vedantu Admin Tool. These fields should be other than whatever fields are already provided in this API otherwise. For example, this set should not have a requirement for “username” or “firstName” etc.

Response fields

Field	Example	Description

userId	654321abc	**required** *This can be set to null if the user registration fails.* UserId of the user in the User Management System.
memberId	MBA2013999	**optional** **default: same as userId** ***Refer: Glossary - Member Id*** If not specified, this field will be assumed to have the value same as userId.
missingParameters	[“firstName”, “CAT Roll Number”]	**optional** Set of missing parameters

Errors

Error Code	Description
USER_ALREADY_EXISTS	The username provided by the user is already registered in the Institute’s User Management System
MISSING_PARAMETERS	If some information required for the user registration is missing

Sample Success Response

```
{
        “errorCode” : “”,
        “errorMessage” : “”,
        “result” : {
                “userId” : “654321abc”,
                “memberId” : “MBA2013999”
        }
}
```

Sample Error Response

```
{
        “errorCode” : “MISSING_PARAMETERS”,
        “errorMessage” : “”,
        “result” : {
```

```
                    “missingParameters” : [ “firstName”, “CAT Roll Number” ]
          }
}
```

How to Integrate?

Use vedantu-ei-api and implement:

Interface	IUserManager
API Method	RegResponse register(RegRequest request)
Manager Conf Key	user.manager.class
Sample Class	com.vedantu.ei.managers.sample.SampleUserManager
Sample Config	user.manager.class=com.vedantu.ei.managers.sample.SampleUserManager

Institute System based User Authentication Process

Refer the following figure to understand the call sequence for the User Authentication process.

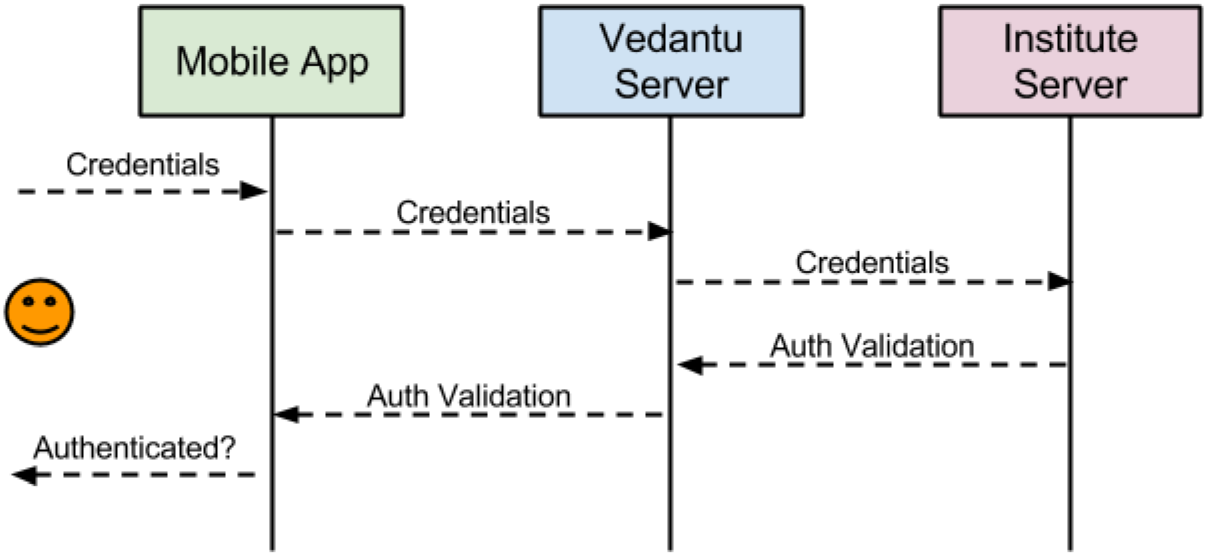

Figure 2. Sequence Diagram for User Authentication Process

Note: Vedantu Server simply acts as a proxy for the User Authentication Process. It does not log or store the user credentials in its database.

User Authentication API

https://<institute-endpoint-url>**/authenticate**

Authenticates the User.

HTTP Method	POST

Parameters

Parameter	Example	Description
username	MBA2013999	**required** Username to be used for authentication against the Institute's User Management System

password	somesecret	**required** Password of the user

Response fields

Field	Example	Description
userId	654321abc	**required** UserId of the user in the User Management System.
memberId	MBA2013999	**optional** **default: same as userId** ***Refer: Glossary - Member Id*** If not specified, this field will be assumed to have the value same as userId.
firstName	Anthony	**required** First name of this user. Required for profile display.
lastName	Gonsalves	**optional** Last name of this user. Required for profile display.
gender	MALE	**optional** *Enum: [MALE, FEMALE, UNKNOWN]* Gender of this user. Required for generic profile-pic display.
role	STUDENT	**required** *Enum: [STUDENT, TEACHER, ADMIN]* Role of the member in the Institute.
classes	[{ “classCode” : “classCode1”, “expiry” : 1387196796000 }, { “classCode” : “classCode2”, “expiry” : 1387196796000 }, ...]	**required** ***Refer: Glossary - Class Code*** Class-Codes of the classes of which the the user is a part. If the user earlier had access to a class whose code is not present here, she would be removed from that class. If there is a class-code here for which the user did not have access earlier she will become a part of that class.

Errors

Error Code	Description
AUTHENTICATION_FAILED	User credentials could not be authenticated successfully.

Sample Success Response

```
{
        “errorCode” : “”,
        “errorMessage” : “”,
        “result” : {
                “userId” : “654321abc”,
                “memberId” : “MBA2013999”,
                “firstName” : “Anthony”,
                “lastName” : “Gonsalves”,
                “role” : “STUDENT”,
                “classes” : [
                        { “classCode” : “classCode1”, “expiry” : 1387196796000 },
                        { “classCode” : “classCode2”, “expiry” : 1387196796000 },
                ...]
        }
}
```

Sample Error Response

```
{
        “errorCode” : “AUTHENTICATION_FAILED”,
        “errorMessage” : “”,
        “result” : {
                “success” : false
        }
}
```

How to Integrate?

Use vedantu-ei-api and implement:

Interface	IUserManager
API Method	AuthResponse authenticate(AuthRequest request)
Manager Conf Key	user.manager.class
Sample Class	com.vedantu.ei.managers.sample.SampleUserManager
Sample Config	user.manager.class=com.vedantu.ei.managers.sample.SampleUserManager

Institue System based User Enrollment in Class Process

Refer the following figure to understand the call sequence for the User Enrollment in Class process.

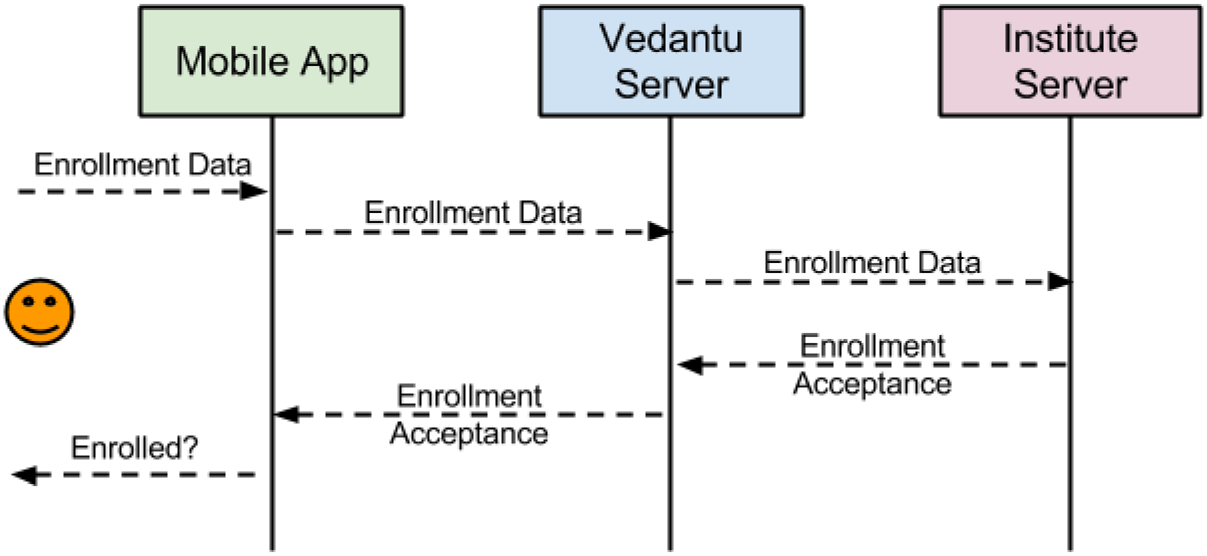

Figure 3. Sequence Diagram for User Enrollment in Class Process

Note: If the user is already enrolled in the class the existing class-enrollment details should be returned (i.e. it is not an exception case).

Class Enrollment API

https://<institute-endpoint-url>**/enroll**

Enroll an existing User to a Class.

HTTP Method	POST

Parameters

Parameter	Example	Description
userId	654321abc	**required** UserId of the user in the User Management System.

classCode	classCode1	**required** ***Refer: Glossary - Class Code*** Class-Code in which the user needs to be enrolled.

Response fields

Field	Example	Description
classCode	classCode1	**required** ***Refer: Glossary - Class Code*** Class-Code for which the enrollment request was submitted.
expiry	1387196796000	**required** Exact time in millis (since Epoch) till which the user can remain a part of this class.

Errors

Error Code	Description
INVALID_USER_ID	If the userId is invalid as per the the Institute's system.
INVALID_CLASS_CODE	If the classCode is invalid as per the the Institute's system.

Sample Success Response

```
{
        “errorCode” : “”,
        “errorMessage” : “”,
        “result” : {
                “classCode” : “classCode1”,
                “expiry” : 1387196796000
        }
}
```

Sample Error Response

```
{
        “errorCode” : “INVALID_USER_ID”,
        “errorMessage” : “”,
        “result” : {
                “success” : false
        }
}
```

How to Integrate?

Use vedantu-ei-api and implement:

Interface	IUserManager
API Method	EnrollResponse register(EnrollRequest request)
Manager Conf Key	user.manager.class
Sample Class	com.vedantu.ei.managers.sample.SampleUserManager
Sample Config	user.manager.class=com.vedantu.ei.managers.sample.SampleUserManager

Institute Usage Data Upload Process

Refer the following figure to understand the call sequence for the User Authentication process.

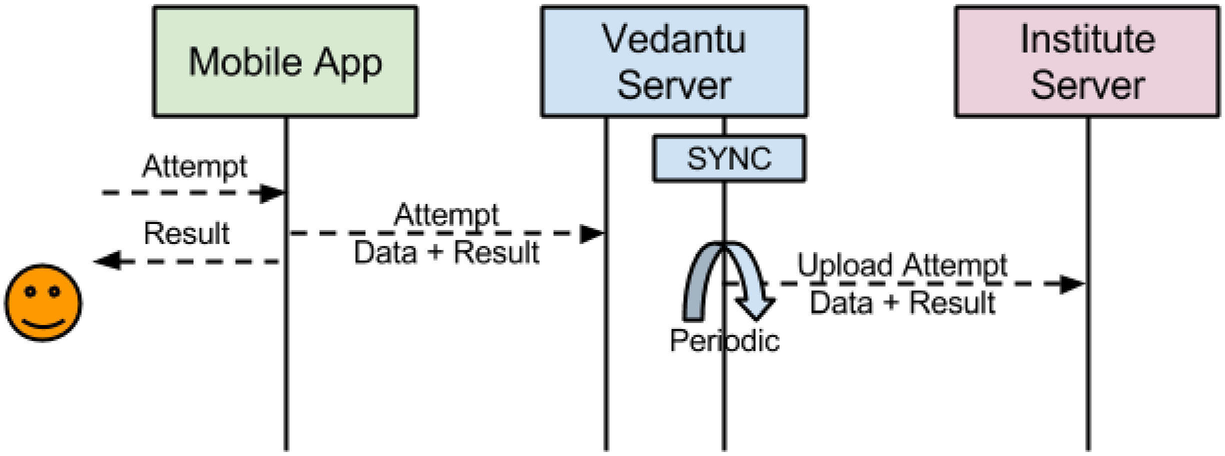

Figure 4. Sequence Diagram for Usage Data Upload Process

Test Attempt Data Upload Process

https://<institute-endpoint-url>**/uploadTestAttemptData**

Authenticates the User.

HTTP Method	POST

Parameters

Parameter	Example	Description
upload	{ “uploadId” : “6b5c4d3e”, “attempts” : [] }	**required** uploadId - A unique identifier for this upload data-set in Vedantu. attempts - Explained further below.
attempt	{“code” : “TEST-001”, “userId” : “654321abc”, “attemptId” :	**required** code - Unique code of the test. userId - UserId of the user as per the Institute’s User Management System. attemptId - A unique identifier for the attempt in Vedantu

	“6a5b4c3d2e1f” “maxScore” : “100”, “userScore” : “50”, “attemptStartTime” : 1387196796000, “attemptEndTime” : 1387196856000, answers : []}	maxScore - The maximum score the user could have attained in the test. userScore - The score attained by the user in the test. attemptStartTime - The time (in milliseconds since Epoch) at which the user started attempting the test. attemptEndTime - The time (in milliseconds since Epoch) at which the user ended attempting the test. answers - Explained further below.
answers	[{ “questionNumber” : 1, “isAttempted” : true, “userAnswer” : “some answer”, “isCorrect” : true, “maxScore” : 4, “userScore” : 4, “timeTaken” : 10000 }, { “questionNumber” : 2, “userAnswer” : “”, “isCorrect” : false, “maxScore” : 4, “userScore” : -1, “timeTaken” : 10000 }, ...]	**required** An array of answers and data thereof given with the following fields for each question: questionNumber - The number of the question in the tes for which the data is being furnished. This will be indexed from 0. isAttempted - Whether the user attempted this question. userAnswer - Answer provided by the user. isCorrect - Whether the user’s answer is correct. maxScore - The maximum score that the user could have attained for this question. userScore - The score attained by the user for this question. timeTaken - The time (in milliseconds) spent by the user on this question.

Response fields

Field	Example	Description
failedAttempts		**required** Only attemptId of attempts that were *not* successfully

		uploaded. This should be empty array in case all the attempts were uploaded successfully.

Errors

Error Code	Description
UPLOAD_FAILED	If there was any error in the upload of data to the Institute’s system.
INVALID_TEST_CODE	*specific to an attempt* If the Test Code is invalid as per the Institute’s system.
INVALID_USER_ID	*specific to an attempt* If the userId is invalid as per the the Institute’s system.

Sample Success Response

```
{
        “errorCode” : “”,
        “errorMessage” : “”,
        “result” : {
                “failedAttempts” : [ ]
        }
}
```

Sample Error Response

```
{
        “errorCode” : “UPLOAD_FAILED”,
        “errorMessage” : “”,
        “result” : {
                “failedAttempts” : [
                        {“attemptId” : “6a5b4c3d2e1f3”, “errorCode” : “INVALID_USER_ID”},
                        {“attemptId” : “6a5b4c3d2e1f4”, “errorCode” : “INVALID_TEST_CODE”},
                        {“attemptId” : “6a5b4c3d2e1f5”, “errorCode” : “INVALID_TEST_CODE”}
                ],

        }
}
```

How to Integrate?

Use vedantu-ei-api and implement:

Interface	IUploadManager
API Method	UploadTestAttemptsResponse uploadTestAttempts(UploadTestAttemptsRequest request)
Manager Conf Key	upload.manager.class
Sample Class	com.vedantu.ei.managers.sample.SampleUploadManager
Sample Config	upload.manager.class=com.vedantu.ei.managers.sample.SampleUploadManager

vedantu-ei-api

Vedantu External-Integration API server-side code for Institutes for integration of Vedantu with Institute's Systems. This enables Vedantu to communicate with the Institute’s Systems.

Download the latest release of vedantu-ei-api from GitHub.

Requirements

Platform	Java 6 ***tested with:*** *java version "1.6.0_30"* *Java(TM) SE Runtime Environment (build 1.6.0_30-b12)* *Java HotSpot(TM) 64-Bit Server VM (build 20.5-b03, mixed mode)*
Build system	Ant (1.8.2 or above) ***tested with:*** *Apache Ant(TM) version 1.8.2*
Application Server	Apache Tomcat ***tested with:*** *Apache Tomcat 7.0.42* *Using some other Application Server would require appropriately adapting the build.xml file, as presently it refers ${env.CATALINA_HOME} or ${env.TOMCAT_HOME} for compilation.*

How to build?

The code can be built using Ant on a shell prompt.

To get the available build options:

```
ant
OR
ant help
```

Note: Appropriately set the environment variable CATALINA_HOME or TOMCAT_HOME to refer to the base directory of your Apache Tomcat installation.

Once you build you will find the following files in the dist folder:

1. vedantu-ei-api.jar
2. vedantu-ei-api.war

Configuring Managers

Manager configuration needs to be specified in *managers.properties* file. This file needs to be on the path at runtime.

managers.properties file can be found in the following locations:

code-base	*conf* folder
war file	*WEB-INF/classes* folder
jar file	Absent.

Supported Managers

Manager	Key	Sample
IUserManager	user.manager.class	com.vedantu.ei.managers.sample.SampleUserManager
IUploadManager	upload.manager.class	com.vedantu.ei.managers.sample.SampleUploadManager

Glossary

Term	Definition
Member Id	Member Id of the user as per the User Management System. This field represents Enrollment Number, Employee Id etc. of the user in the User Management System. It is recommended that this field be provided by the User Management System to allow Admin to be able to effectively do Management operations through Vedant Admin Tool, if desired. If not specified, this field will be assumed to have the value same as userId.
Class Code	Unique codes of sections created in Vedantu of the classes of which the the user is a part. Only Libraries and other resources of the authorized classes will be available to the user.