A-VIEW API Document (REST APIs)

Contents

Use Cases

You may use the below APIs to communicate with A-VIEW database for the given scenarios using HTTP POST requests.

1. User

1.1. Create a new user

This API provides you the ability to create a new user to A-VIEW.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/createuser.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (*adminId*)

This is the user ID of your A-VIEW administrator. This will be provided to you separately.

Parameter 2: User Details *(userDetails)*

This parameter (*userDetails*) should contain the following details of the user in JSON format. Field name for each of the below items should be exactly as given in the example given below.

a. User name(userName)

This is the user preferred login name; this has to be unique.
Data type: Varchar (100)
Accepted Characters: A-z0-9-&,.';:_()/+|

b. First name(fname)

This is the first name of the user.
Data type: Varchar (100)
Accepted Characters: 0-9A-z_.– and space

c. Last name(lname)

This is the last name of the user.
Data type: Varchar (100)
*Accepted Characters:*0-9A-z_.–and space

d. Email ID(email)

This is the email ID of the user and should be in email ID format.
Data type: Varchar (256)

e. User Address(address)

This is the address of the user.
Data type: Varchar (500)
Accepted Characters: A-z 0-9-,./ and space

f. City(city)

City is a part of the user's address.
Data type: Varchar (100)
*Accepted Characters:*A-z 0-9_- and space

g. District ID(districtId)

This will be district ID of the user. As per your request, ID of Chennai will be provided to you separately.

h. Pin code(zipId)

Pin code/Zip code is a part of the address. This should be a number of length 6.

i. Password(password)

This is the password of the user encrypted using SHA-1.
Data type: Varchar (50)

j. Mobile Number(mobileNumber)

This is the mobile number of the user; this should be a 10 digit number.

k. Institute ID(instituteId)

This is the ID of your institute. This will be provided to you separately.

Optional fields for Parameter2:

a. User Role (role)

User role can be either "*STUDENT*" or "*TEACHER*". If the user is going to be made as Moderator for some class(es), the user should be of role "Teacher". Default value is "*STUDENT*".

Parameter 3: Send Email (*sendEmail*)

This is for sending email notification to the user after the successful creation. If you don't want to send the notification, you can leave this parameter value blank, otherwise value should be mentioned as Y or y.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can be either the User ID or the error message, based on the HTTP status.

a. User ID

This should be used to map A-VIEW users with QEEE user table.

b. Error message (if status code is 4xx or 5xx)

Example 1:

Input

adminId: 18756
userDetails: {"userName": "qeeeTest23", "fname": "QEEE", "lname": "Test23", "email": "tst@gmail.com","address": "IITMadras,Adayar", "city": "Chennai", "districtId": 489,"zipId": "320525","password": "f95016b63cd4e3f74b3029639e7c7d9fde71d7db", "mobileNumber": "9945128902","instituteId": 1243,"role": "TEACHER"}
sendEmail:

Output


```
{
   "statusType": "OK",
   "entity": 36445,
   "entityType": "java.lang.Long",
   "metadata": {},
   "status": 200
}
```

Example 2:

Input

adminId: 18756
userDetails: {"userName": "qeeeTest24", "fname": "QEEE", "lname": "Test23", "email": "tst1@gmail.com","address": "IITMadras,Adayar", "city": "Chennai", "districtId": 489,"zipId": "320525","password": "f95016b63cd4e3f74b3029639e7c7d9fde71d7db", "mobileNumber": "9945128902","instituteId": 1243,"role": "TEACHER"}
sendEmail: y

Output


```
{
   "statusType": "OK",
   "entity": 36446,
   "entityType": "java.lang.Long",
   "metadata": {},
   "status": 200
}
```


1.2. Search an existing user

This API provides you the ability to search an existing user from A-VIEW, using username, role, firstname, last name, city or instituteId. This will list out all the users with respective matches.

URL

The post request can be passed to the below URL:

http://<*IP Address*> :< *Port Number*>/aview/searchuser.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (admin Id)

This is the user ID of your A-VIEW admin user. This will be provided to you.

Parameter 2: username (userName)

This is the user nameof the user(s) who are to be searched; it can also be a part of it.

Parameter 3: Role (role)

This parameter should be *student, teacher or administrator.*

Parameter 4: First name (fname)

This parameter contains part of first name or first name.

Parameter5: Last name (lname)

This is the last name of the user(s) who are to be searched, it can also be a part of it.

Parameter6: City (city)

This is the city of the user(s) who are to be searched, it can also be a part of it.

Parameter7: Institute Id (instituteId)

This should be a valid institute Id.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can be either the list of users or the error message, based on the HTTP status.

a. User List

This will list all the corresponding user details (user name, user id, institute, parent institute, mobile number, first name, last name, email id and role)

b. Error message (if status code is 4xx or 5xx)

Example 1:

Input

adminId: 18456
userName: test
fname: qeee
lname: test
city: chennai
instituteId: 13
role: teacher

Output

The following is the sample of output


```
{
   "statusType": "OK",
   "entity": [
      [
         "userName:qeeeTest2303-05-2014",
         "userId:42889",
         "firstName:QEEE",
         "lastName:Test23",
         "institute:Amrita E-Learning Research Lab",
         "parentInstitute:null",
         "role:TEACHER",
         "email:tst@gmail.com",
         "mobileNumber:9945128902"
        ]
        ]
   "entityType": "java.util.ArrayList",
   "metadata": {},
   "status": 200
}
```


Example 2:

Input

```
adminId: 18456
userName: test
fname:
lname:
city:
instituteId:
role:
```

Output


```
{
   "statusType": "OK",
   "entity": [
      [
         "userName:Test 123",
        "userId:11425",
        "firstName:",
        "lastName:",
        "institute:__DUMMY__INSTITUTE__",
        "parentInstitute:null",
        "role:GUEST",
        "email:",
```

```
                "mobileNumber:null"
                ],
                [
               "userName:testpu1",
                "userId:42539",
                "firstName:Test",
                "lastName:User1",
                "institute:Parent Institute1",
                "parentInstitute:null",
                "role:TEACHER",
                "email:aview@amrita.edu",
                "mobileNumber:1234567890"
                ]
    ]
       "entityType": "java.util.ArrayList",
       "metadata": {},
       "status": 200
    }
```


Example 3:

Input

adminId: 18456
userName: test
fname: test
lname:
city: kollam
instituteId:
role: teacher

Output

The following is the sample of output


```
{
   "statusType": "OK",
   "entity": [
      [
         "userName:testpu1",
         "userId:42539",
         "firstName:Test",
         "lastName:User1",
         "institute:Parent Institute1",
         "parentInstitute:null",
         "role:TEACHER",
         "email:aview@amrita.edu",
         "mobileNumber:1234567890"
         ],
```

```
        ]
    "entityType": "java.util.ArrayList",
    "metadata": {},
    "status": 200
}
```


1.3. Update an existing user

This API provides you the ability to update an existing userin A-VIEW.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:< *Port Number*>/aview/updateuser.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW administrator. This will be provided to you separately.

Parameter 2: User Details (userDetails)

This parameter (userDetails) should contain the following details of the user in JSON format. Field name for each of the below items should be exactly as given in the example given below. Only the values which are to be updated are needed to be specified.

a. UserId(userId)

This is the user ID of the user whose details are to be updated.
Data type: int(10)

b. First name (fname)

This is the new value for first name of the user.
Data type: Varchar (100)
Accepted Characters: 0-9A-z_.- and space

c. Last name (lname)

This is the new value for last name of the user.
Data type: Varchar (100)
Accepted Characters: 0-9A-z_.- and space

d. Email ID (email)

This is the new email ID of the user and should be in email ID format.
Data type: Varchar (256)

e. User Address (address)

This is the new address of the user.
Data type: Varchar (500)
Accepted Characters: A-z 0-9-,./ and space

f. District ID (districtId)

This is the new district ID of the user. As per your request, ID of Chennai will be provided to you separately.

g. City (city)

This is the newcity of the user, it is a part of the user's address.
Data type: Varchar (100)
Accepted Characters: A-z 0-9_- and space

h. Pin code (zipId)

This is the new Pin code/Zip code of the user's address. This should be a number of length 6.

i. Mobile Number (mobileNumber)

This is the new mobile number of the user; this should be a 10 digit number.

j. Institute ID

This is the ID of your institute. This will be provided to you separately.

k. User Role (role)

User role can be either "STUDENT" or "TEACHER". If the user is going to be made as Moderator for some class(es), the user should be of role "Teacher".

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can be either the success message or the error message, based on the HTTP status.

a. success message with User ID

This should be used to map A-VIEW users with QEEE user table.

b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 18756
userDetails: {"userId":36440, "fname":"QEEE", "lname":"Test 23", "city":"Chennai"}

Note: Here, only the values which are to be updated are specified; those values which are not mentioned will have the previous values.

Output


```
{
   "statusType": "OK",
   "entity": "Updated user(ID:36440)successfully",
   "entityType": "java.lang.String",
   "metadata": {},
   "status": 200
}
```


1.4. Delete an existing user

This API provides you the ability to delete an existing user in A-VIEW.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/deleteuser.html

Input parameters

The input parameters are:

*Parameter 1:*Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you.

Parameter 2: userID (userId)

This is the user Id of the user who is to be deleted.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can be either the success message or the error message, based on the HTTP status.

a. Success message with User ID

b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 10010
userId: 42896

Output


```
{
  "statusType": "OK",
  "entity": "Deleted test06-05-2014(ID: 42896) successfully",
  "entityType": "java.lang.String",
  "metadata": {},
  "status": 200
}
```


2. Institute

2.1. Search an existing institute

This API is used to search institute from A-VIEW (i.e. using institute name, this will list out all the institute details with corresponding matches).

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/searchinstitute.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of you're A-VIEW admin user/MASTER_ADMIN. This will be provided to you.

Parameter 2: Institute Name (instituteName)

This parameter contains part of institute name.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can be either the class list or the error message, based on the HTTP status.

a. institute List

This will list all the corresponding institute details (institute, institute id, instituteType, city, category, address, city,district and parentInstitute)

b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 10010

instituteName: adi

Output


```
{    "statusType": "OK",
   "entity": [
     [
          "institute: MAR BASELOUS COLLEGE ADIMALI ",
          "instituteId: 180",
          "instituteType: College",
          "category: Arts and Sciences",
          "address: THE PRINCIPAL MAR BASELOUS COLLEGE X/781,,ADIMALI",
          "city: Adimaly",
          "district: Idukki",
          "parentInstitute: null"
   ],
   [
          "institute: First Grade Ladies College",
          "instituteId: 262",
          "instituteType: College",
          "category: Arts and Sciences",
          "address: First Grade Ladies College,Kamblikoppa,sagar ",
          "city: Sagar",
          "district: Uttar Kannada",
          "parentInstitute: null"
      ],
      [
          "institute: AdikaviNannaya University",
          "instituteId: 745",
          "instituteType: University",
          "category: Agriculture",
          "address: AdikaviNannaya University, 25-07-9/1,  Jayakrishnapuram Rajahmundry-
          533105. Andhra Pradesh",
          "city: Rajahmundry",
          "district: East Godavari",
          "parentInstitute: null"
      ]
   ],
“entityType": "java.util.ArrayList",
   "metadata": {},
   "status": 200
}
```

3. Course

3.1. Search an existing course

This APIprovides you the ability to search course from A-VIEW (i.e. either using course name or institute id); this will list out all the details about the course with corresponding matches.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/searchcourse.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you.

Parameter 2: course name (courseName)

This can be a part of course name (or the full course name).

Parameter 3: institute ID (instituteId)

This is the institute under which the courses are to be fetched.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can be either the class list or the error message, based on the HTTP status.

a. course List

This will list all the corresponding course details (course, course id, course code and institute)

b. Error message (if status code is 4xx or 5xx)

Example 1:

Input

adminId: 18456
courseName:
instituteId:1

Output


```
{
   "statusType": "OK",
   "entity": [
      [
         "course: Web Technology",
         "courseId: 1",
         "courseCode: Web Technology",
         "institute: Amrita E-Learning Research Lab"
      ]
   ],
   "entityType": "java.util.ArrayList",
   "metadata": {},
   "status": 200
}
```


Example: 2

Input

adminId: 18456
courseName: web
instituteId:

Output


```
{
   "statusType": "OK",
   "entity": [
      [
         "course: Web Technology",
         "courseId: 1",
         "courseCode: Web Technology",
         "institute: Amrita E-Learning Research Lab"
      ]
   ],
   "entityType": "java.util.ArrayList",
   "metadata": {},
   "status": 200
}
```

4. Class

4.1. Create a new class

This API provides you the ability to create a new class on A-VIEW.

URL

The post request can be passed to the below URL:

http://<*IP* Address>:<*Port Number*>/aview/createclass.html

Input parameters

The input parameters are:

Parameter 1: AdminID (*adminId*)

This is the user ID of your A-VIEW admin user. This will be provided to you separately.

Parameter 2: Class Details (classDetails)

This parameter (*classDetails*) should contain the following details of the class in JSON format.

a. Class Name(className)

This is the desired name for the class. This should be unique for the given course.
Data type: Varchar (100)
*Accepted Characters:*A-z 0-9-&,.';:_()/+|

b. Course ID(courseId)

This is the ID of the course under which the class should be created. This will be provided to you separately.

c. Start Date(startDate)

This is the start date for the class. It should be in the format YYYY-MM-DD as string.
e.g. 2013-09-24

d. End Date(endDate)

This is the end date for the class. It should be in the format YYYY-MM-DD as string.
e.g. 2014-01-24

Optional fields for Parameter2:

a. Class Description(classDescription):

This can be brief description about the class. Default value is null.
Data type: Varchar (4000)
*Accepted Characters:*A-z0-9<>'/@?&*()#$%{}!~=.,;?: and space

b. Maximum Number of Students in the class(maxStudents)

This is the maximum number of students who can log into the class at a time. Default value is 500.
Data type: Integer (10)

c. Minimum bandwidth to publish for students(minPublishingBandwidthKbps)

This will be the minimum bandwidth any student can publish their video at, possible values are 28, 56, 128, 256, 512, 768, 1024, 2520, 5670 and 8505. Minimum value should be always less than or equal to max value. Default value is 28.

d. Maximum bandwidth to publish for students(maxPublishingBandwidthKbps)

This will be the maximum bandwidth any student can publish their video at,Possiblevalues are 28, 56, 128, 256, 512, 768, 1024, 2520, 5670 and 8505. Maximum value should be always greater than or equal to min value. Default value is 1024.

e. Maximum number of viewers for interaction(maxViewerInteraction)

This is the way to enable or disable multiple user interaction. Possible values are 1 to 8. If the given value is 1, Presenter can interact with only one Viewer at a time. If the value is between 2-8, Presenter can interact with that many viewers at a time. If the class is for presentation/teaching, value 1 is recommended; if the class is for some discussion/ meeting, values from 2 to 8 are recommended; this value should be decided considering the bandwidth at the Viewer nodes. Default value is 1.

f. DefaultTeacherBandwidth(presenterPublishingBwsKbps)

This will be the default bandwidth selected for the teachers; they can change the bandwidth later from the A-VIEW settings screen. Possible values are 28, 56, 128, 256, 512, 768, 1024, 2520, 5670 and 8505. Default value is 256.

g. Video Codec(videoCodec)
   This is the video codec that will be used during the live class. This can be either "H.264" or "Sorenson" for A-VIEW web version. Default value is H.264 and it is the recommended value, it provides better quality at low latency.
h. Enable People Count (enablePeopleCount)
   This is to enable/disable People Count feature during the live class. Possible values are Yes/No. Default value is No.

Parameter 3: Class Server Details(classServerDetails)

This parameter (*classServerDetails*) value will be a constant for a given server. Please use the exact data in the below example(in JSON format) for this parameter.

Output parameters

The following are the main output parameters:

a. HTTP status code

   This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

   This will be the HTTP status message of the process.

c. Entity
   This can be either the Class ID or the error message, based on the HTTP status.
   a. Class ID
      This should be used to map A-VIEW classes with QEEE course table.
   b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 18756
classDetails:

```
{ "className":"Test Class 2", "courseId":1,
    "startDate":"2013-10-19", "endDate":"2013-12-31",
    "classDescription":"Test Class 2 is for testing purposes",
    "maxStudents":500,
    "minPublishingBandwidthKbps":28,
    "maxPublishingBandwidthKbps":1024,
    "maxViewerInteraction":1,
    "presenterPublishingBwsKbps":"512",
    "videoCodec":"H.264"
    }
```

classServerDetails:

{"serverDetail": [{"serverTypeId": 3 ,"serverPort": 1935,
"presenterPublishingBandwidthKbps":-1,
"server":{"serverId":7}},{"serverTypeId": 4, "serverPort": 80,
"presenterPublishingBandwidthKbps":-1, "server":{"serverId":7}},
{"serverTypeId": 5 ,"serverPort": 1935,
"presenterPublishingBandwidthKbps":256, "server":{"serverId":7}},
{"serverTypeId": 2 ,"serverPort": 1935,
"presenterPublishingBandwidthKbps":-1, "server":{"serverId":7}},
{"serverTypeId": 1, "serverPort": 1935,
"presenterPublishingBandwidthKbps":-1 ,"server":{"serverId":7}}] }

Output


```
{
    "statusType": "OK",
    "entity": 1001,
    "entityType": "java.lang.Long",
    "metadata": {},
    "status": 200
}
```


4.2. Search an existing class

This API provides you the ability to search an existing class in A-VIEW (i.e. either using classname or course Id or instituteId or all); this will list out all the class details with corresponding matches.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/searchclass.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you.

Parameter 2: class name (className)

This is the class nameof which details are to be fetched, it can be given in part also.

Parameter 3: course Id (courseId)

This is the course under which the classes are to be fetched.

Parameter 4: institute Id (instituteId)

This must be a valid institute id.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can be either the class list or the error message, based on the HTTP status.

a. Class List
This will list all the corresponding class details(class, class id, institute, course, start date and end date)

b. Error message (if status code is 4xx or 5xx)

Example1

Input

adminId: 18456
className:
courseId: 1
instituteId:

Output

{
"statusType": "OK",
"entity":
[
[
"class: Abcd Course test1",
"classId: 636",
"institute: Amrita E-Learning Research Lab",
"course: Web Technology",
"startDate: 2013-10-19",
"endDate: 2013-12-31"
],
[
"class: Abcd Course test2",

"classId: 638",
"institute: Amrita E-Learning Research Lab",
"course: Web Technology",
"startDate: 2013-10-19",
"endDate: 2013-12-31"
],
],
"entityType": "java.util.ArrayList",
"metadata": {},
"status": 200
}

Example 2

Input

adminId: 18456
className: test
courseId:
instituteId: 13

Output

{
"statusType": "OK",
"entity": [
[
"class: Abcd Course test1",
"classId: 636",
"institute: Amrita E-Learning Research Lab",
"course: Web Technology",
"startDate: 2013-10-19",
"endDate: 2013-12-31"
],
[
"class: Abcd Course test2",
"classId: 638",
"institute: Amrita E-Learning Research Lab",
"course: Web Technology",
"startDate: 2013-10-19",
"endDate: 2013-12-31"
],
],
"entityType": "java.util.ArrayList",
"metadata": {},
"status": 200
}

4.3. Update an existing class

This API provides you the ability to update an existing class in A-VIEW.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/updateclass.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW administrator. This will be provided to you separately.

Parameter 2: Class Details (classDetails *)*

This parameter (classDetails) should contain the following details of the class in JSON format. Only the values which are to be updatedareneeded to bespecified.

a. ClassId(classId)

   This is the class ID of the class which is to be updated.
   Data type: int (10)

b. Class Name (className)

   This is the new name for the class, but should be unique for the given course.
   Data type: Varchar (100)
   Accepted Characters: A-z 0-9-&,.';:_()/+|

c. Start Date (startDate)

   This is the new start date for the class. It should be in the format YYYY-MM-DD as string.
   e.g. 2013-09-24

d. End Date (endDate)

   This is the new end date for the class. It should be in the format YYYY-MM-DD as string.
   e.g. 2014-01-24

e. Class Description(classDescription):

This can be new brief description about the class.
Data type: Varchar (4000)
Accepted Characters: A-z0-9<>'/@?&*()#$%{}!~=.,;?: and space

f. Maximum Number of Students in the class (maxStudents)

This is the new value for maximum number of students who can log into the class at a time.
Data type: Integer (10)

g. Minimum bandwidth to publish for students (minPublishingBandwidthKbps)

This will be the new value for minimum bandwidth any student can publish their video at; possible values are 28, 56, 128, 256, 512, 768, 1024, 2520, 5670 and 8505. Minimum value should be always less than or equal to max value.

h. Maximum bandwidth to publish for students

This will be the new value for maximum bandwidth any student can publish their video at; possible values are 28, 56, 128, 256, 512, 768, 1024, 2520, 5670 and 8505. Maximum value should be always greater than or equal to min value.

i. Maximum number of viewers for interaction (maxViewerInteraction)

This is the way to enable or disable multiple user interaction. Possible values are 1 to 8. If the given value is 1, Presenter can interact with only one Viewer at a time. If the value is between 2 -8, Presenter can interact with that many Viewers at a time. If the class is for presentation/teaching, value 1 is recommended; if the class is for some discussion/ meeting, values from 2 to 8 are recommended; this value should be decided considering the bandwidth at the Viewer nodes.

j. Default Teacher Bandwidth (presenterPublishingBwsKbps)
This will be the new value for default bandwidth selected for the teachers; they can change the bandwidth later from the A-VIEW settings screen. Possible values are 28, 56,128, 256, 512, 768, 1024, 2520, 5670 and 8505.

k. Video Codec (videoCodec)

This is the new value for video codec that will be used during the live class. This can be either “H.264” or “Sorenson” for A-VIEW web version. H.264 provides better quality at low latency.

*Parameter 3:Class Server Details(*classServerDetails)

This parameter (classServerDetails) value will be a constant for a given server. Please use the exact data in the below example (in JSON format) for this parameter.

Note: If we are updating some server in the given list, the entire server block has to be specified in the Parameter 3. If we don't want to update any of the servers, we can leave this blank.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can be either the success message with Class ID or the error message, based on the HTTP status.

a. Success message with Class ID
b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 18756
classDetails:

```
{ "classId": 769,
"className":"Test Class 2",
"startDate":"2013-10-19", "endDate":"2013-12-31",
"classDescription":"Test Class 2 is for testing purposes",
"maxStudents":500,
"minPublishingBandwidthKbps":28,
"maxPublishingBandwidthKbps":1024,
"maxViewerInteraction":1,
"presenterPublishingBwsKbps":"512",
"videoCodec":"H.264"
}
```

classServerDetails:

```
{"serverDetail": [{"serverTypeId": 3 ,"serverPort": 1935,
```

"presenterPublishingBandwidthKbps":-1,
"server":{"serverId":7}},{"serverTypeId": 8,
"serverPort": 80, "presenterPublishingBandwidthKbps":-1,
"server":{"serverId":18}},
{"serverTypeId": 11 ,"serverPort": 1935,
"presenterPublishingBandwidthKbps":256,
"server":{"serverId":7}}, {"serverTypeId": 12 ,"serverPort": 1935,
"presenterPublishingBandwidthKbps":-1, "server":{"serverId":7}},
{"serverTypeId": 13,
"serverPort": 1935, "presenterPublishingBandwidthKbps":-1,
"server":{"serverId":7}}] }

Output

{
"statusType": "OK",
"entity": "Updated class 769(ID:769) successfully",
"entityType": "java.lang.String",
"metadata": {},
"status": 200
}

4.4. Delete an existing class

This API provides you the ability to delete an existing class in A-VIEW.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/deleteclass.html

Input parameters

The input parameters are:

*Parameter 1:*Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you.

Parameter 2: classID(classId)

This is the class ID of the class which is to be deleted.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can be either the success message or the error message, based on the HTTP status.

a. Success message with class ID

b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 10010
classId: 979

Output


```
{
   "statusType": "OK",
   "entity": "Deleted Test Class 05-05-2014(ID: 979) successfully”,
   "entityType": "java.lang.String",
   "metadata": {},
   "status": 200
}
```


5. Lecture

5.1. Create a new lecture

This API provides you the ability to create a new lecture in A-VIEW.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<Port *Number*>/aview/createlecture.html

Input parameters

The input parameters are:

*Parameter 1:*Admin ID *(adminId)*

This is the user ID of your A-VIEW admin user. This will be provided to you separately.

Parameter 2: LectureDetails (lectureDetails)

This parameter *(lectureDetails)*should contain the following details of the lecture in JSON format.

a. Lecture Name (Should be unique for the class)(lectureName)

   This is the desired name for the lecture. This should be unique for the given class.
   Datatype*:* Varchar (256)
   Accepted Characters*:*A-z 0-9-&,.';:_()/+|

b. Class ID(classId)

   This is the ID of the class under which the lecture should be created.
   Data type: Integer

c. Date(startDate)
   This is the date on which the lecture is to be created.
   Data type: Date
   Format: YYYY-MM-DD; e.g. "2013-09-24"

   Note: Only highlighted value should be changed, time is just a dummy value, but it is needed.

d. Start Time(startTime)

   This is the start time of the lecture.
   Data type: Time
   Format: YYYY-MM-DDThh:mm:ss.sTZDe.g. "1899-12-31T08:30:00.000+0530"

   Note: Only highlighted value should be changed, date is just a dummy date, but it is needed.

e. End Time(endTime)

   This is the end time of the lecture.
   Data type: Time
   Format: YYYY-MM-DDThh:mm:ss.sTZD e.g. "1899-12-31T14:03:00.000+0530"

   Note: Only highlighted value should be changed, date is just a dummy date, but it is needed.

Optional fields for Parameter 2:

a. Keywords(keywords)

   This field can contain some keywords about the lecture which can be used for searching purposes. Keywords can be separated with space.
   Data type: Varchar (1000)
   *Accepted Characters:*A-z 0-9-&,.';:_()/+|

Parameter 3: Send Email (*sendEmail*)

This is for sending email notification to the users after creating the lecture. If you don't want to send the notification, you can leave this parameter value blank, otherwise value should be mentioned as Y or y.

Output parameters

The following are the main output parameters:

a. HTTP status code
   This will be a 3 digit number denoting the status of the API call.
b. HTTP status Message
   This will be the HTTP status message of the process.
c. Entity
   This can be either the Lecture ID or the error message, based on the HTTP status.
   a. Lecture ID
      This should be used to map A-VIEW lectures with QEEE lecture table.
   b. Error message (if status code is 4xx or 5xx)

Example1

Input

adminId: 18756
lectureDetails: {"lectureName":"Test Lecture 5 2013-12-22","startDate":"2013-10-19","classId":638,"startTime":"1899-12-31T08:00:00.000+0530","endTime":"1899-12-31T17:59:00.000+0530","keywords":"test"}
sendEmail : y

Output


```
{
   "statusType": "OK",
   "entity": 160371,
   "entityType": "java.lang.Long",
   "metadata": {},
   "status": 200
}
```


Example2

Input

adminId: 18756
lectureDetails: {"lectureName":"Test Lecture 01 2013-12-22","startDate":"2013-10-19","classId":638,"startTime":"1899-12-31T08:00:00.000+0530","endTime":"1899-12-31T17:59:00.000+0530","keywords":"test"}
sendEmail :

Output


```
{
   "statusType": "OK",
   "entity": 160372,
   "entityType": "java.lang.Long",
   "metadata": {},
   "status": 200
}
```


5.2. Search an existing lecture

This API provides you the ability to search an existing lecture in A-VIEW (using classId);this will list out all the lecturescreated under the given class with the lecture details.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/searchlecture.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you.

Parameter 2: class ID classId)

This is the class ID under which lectures are to be fetched.

Output parameters

The following are the main output parameters:

a. HTTP status code

   This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

   This will be the HTTP status message of the process.

c. Entity

   This can be either the lecture list or the error message, based on the HTTP status.

   a. Lecturelist

This will list all the corresponding lectures'details(lecture, lecture id, class, date, start time and end time)

b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 18456
classId: 675

Output


```
{
   "statusType": "OK",
   "entity": [
      [
         "lecture: Try 2013-4-28",
         "lectureId: 152495",
         "class: class03",
         "date: 2013-04-28",
         "startTime: 00:00:00",
         "endTime: 23:55:00"
      ],
      [
         "lecture: Try 2013-4-29",
         "lectureId: 152496",
         "class: class03",
         "date: 2013-04-29",
         "startTime: 00:00:00",
         "endTime: 23:55:00"
      ],
      [
         "lecture: Try 2014-4-30",
         "lectureId: 152862",
         "class: class03",
         "date: 2014-04-30",
         "startTime: 00:00:00",
         "endTime: 23:55:00"
      ]
   ],
      "entityType": "java.util.ArrayList",
      "metadata": {},
      "status": 200
 }
```

5.3. Update an existing lecture

This API provides you the ability to update an existing lecture in A-VIEW.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:< *Port Number*>/aview/updatelecture.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW administrator. This will be provided to you separately.

*Parameter 2: Lecture Details(*lectureDetails*)*

This parameter (lectureDetails) should contain the following details of the lecture in JSON format. Only the values which are to be updatedareneeded to bespecified.

a. LectureId(lectureId)

This is the lecture id of the lecture to be updated.
Data type:Integer

b. Lecture Name (lectureName)

This is the new name of the lecture, but should be unique for the given class.
Data type: Varchar (256)
Accepted Characters: A-z 0-9-&,.';:_()/+|

c. Date (startDate)

This is the new date of the lecture.
Data type: Date
Format: YYYY-MM-DD; e.g. “2013-09-24”

d. Start Time (startTime)

This is the new start time of the lecture.
Data type: Time
Format: YYYY-MM-DDThh:mm:ss.sTZD e.g. "1899-12-11T08:30:00.000+0530"

Note: Only highlighted value should be changed, date is just a dummy date, but it is needed.

e.End Time(endTime)

This is the new end time of the lecture.
Data type: Time
Format: YYYY-MM-DDThh:mm:ss.sTZD
e.g. "1899-12-31T14:03:00.000+0530"

Note: Only highlighted value should be changed, date is just a dummy date, but it is needed.

f. Keywords (keywords)

This is the newset of keywords about the lecture which can be used for searching purposes, older values will get completely replaced with this new set. Keywords can be separated with space.
Data type: Varchar (1000)
Accepted Characters: A-z 0-9-&,.';:_()/+|

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can be either the success message or the error message, based on the HTTP status.

a. Success message with lecture ID
b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 18756
lectureDetails:

```
{
"lectureId":638,
"lectureName":"Test Lecture 5 2013-12-22",
"startDate":"2013-10-19",
"endTime":"1899-12-31T17:59:00.000+0530",
}
```

Note: Here, only the fields which are to be updated are specified.

Output


```
{
   "statusType": "OK",
   "entity": "Updated lecture638 (ID: 638)successfully",
   "entityType": "java.lang.String",
   "metadata": {},
   "status": 200
}
```


5.4. Delete an existing lecture

This API provides you the ability to delete an existing lecture in A-VIEW.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/deletelecture.html

Input parameters

The input parameters are:

*Parameter 1:*Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you.

Parameter 2: lectureID (lectureId)

This is the lecture ID of the lecture to be deleted.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can be either the success message or the error message, based on the HTTP status.

a. Success message with lecture ID
b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 10010
lectureId: 36444

Output


```
{
   "statusType": "OK",
   "entity": "Deleted test lecture api 12-03-2014(ID:36444) successfully",
   "entityType": "java.lang.String",
   "metadata": {},
   "status": 200
}
```


6. Class Enrollment

6.1. Create a Class Enrollment

This API provides you the ability to enroll user to A-VIEW class. Right now, we can do enrollment of only one student at a time, if you want to enroll multiple students at a time, you may have to call the API repeatedly.

URL

The post request can be passed to the below URL:

http://<*IP* Address>:<*Port Number*>/aview/createclassregister.html

Input parameters

The input parameters are:

*Parameter 1:*Admin ID *(adminId)*

This is the user ID of your A-VIEW admin user. This will be provided to you separately.

*Parameter 2:*Class ID *(classId)*

This is the ID of the class to which you want to enroll the user to.
Data type: Integer

Parameter 3: User ID *(userId)*

This is the ID of the user whom you want to enroll to the given class. The combination of class ID and user ID should be unique.
Data type: Integer

Parameter 4: Is Moderator*(isModerator)*

This is the field which determines who is the moderator for the given class; for a class, there can be only one Moderator and the user should be of role "Teacher". *Possible Values:* 'Y'-if the user is going to be the moderator;otherwise 'N'.

Parameter 5: Send Email (*sendEmail*)

This is for sending email notification to the user regarding the class enrollment. The email will contain the schedule of upcoming sessions (lectures) of this class. If you don't want to send the notification, you can leave this parameter value blank, otherwise value should be mentioned as Y or y.

Output parameters

The following are the main output parameters:

a. HTTP status code

   This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

   This will be the HTTP status message of the process.

c. Entity

   This will contain details about the message.

Example1

Input

adminId: 18756
classId: 123
userId: 1222
isModerator: N
sendEmail : y

Output


```
{
   "statusType": "OK",
   "entity": 82838,
   "entityType": "java.lang.Long",
   "metadata": {},
   "status": 200
}
```

Example2

Input

adminId: 18756
classId: 123
userId: 1222
isModerator: N
sendEmail :

Output


```
{
   "statusType": "OK",
   "entity": 82839,
   "entityType": "java.lang.Long",
   "metadata": {},
   "status": 200
}
```


6.2. Search a Class Enrollment

This API provides you the ability to search class enrollment from A-VIEW (either using userId or classId); this will list out all the class enrollments for the given user or for the given class.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/searchclassregister.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you.

Parameter 2: userID(userId)

This is the user ID of the user whose class enrollments are to be found.

Parameter 3: Class ID (classId)

This is the class ID of the class of which class enrollments are to be found.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can be either the class enrolled list or the error message, based on the HTTP status.

a. Class Enrollment List

This will list all the corresponding class enrollments (class registration id, user name, user id, class id, class, and whether the is moderator or not)

b. Error message (if status code is 4xx or 5xx)

Example1

Input

adminId: 18456
classId:
userId: 36440

Output

{
"statusType": "OK",
"entity": [
[
"classRegistrationId: 82836",
"userName: user1-04-03-2014",
"userId: 36440",
"classId: 994",
"class: Test Class 02 22-03-2014",
"isModerator: N"
]
],
"entityType": "java.util.ArrayList",
"metadata": {},
"status": 200
}

Example 2

Input

adminId: 18456
classId: 994
userId:

Output

{
"statusType": "OK",
"entity": [
[
"classRegisterId: 82836",
"userName: user1-04-03-2014",
"userId: 36440",
"classId: 994",
"class: Test Class 02 22-03-2014",
"isModerator: N"]
],
"entityType": "java.util.ArrayList",
"metadata": {},
"status": 200
}

6.3. Update a Class Enrollment

This API provides you the ability to update an existing class enrollment in A-VIEW. This can be mainly used to make a user (teacher) moderator or to make him/her a Viewer.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:< *Port Number*>/aview/updateclassregister.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW administrator. This will be provided to you separately.

Parameter 2: Class Enrollment ID (*classRegisterId)*

This is class enrollment id which is to be updated.
Data type: Integer

Parameter 3: Is Moderator (isModerator)

Possible Values: 'Y'- if the user is going to be the moderator; otherwise 'N'.

Output parameters

The following are the main output parameters:

a. HTTP status code
This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message
This will be the HTTP status message of the process.

b. Entity
This can be either the success message or the error message, based on the HTTP status.
a. Success message with class enrollment ID

b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 18756
classRegisterId: 82786
isModerator: N

Output


```
{
   "statusType": "OK",
   "entity": "Updated class enrollment (ID:82830) successfully",
   "entityType": "java.lang.String",
   "metadata": {},
   "status": 200
}
```


6.4. Delete a Class Enrollment

This API provides you the ability to delete an existing class enrollment inA-VIEW.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/ deleteclassregister.html

Input parameters

The input parameters are:

*Parameter 1:*Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you.

*Parameter 2*Classenrollment ID (classregisterId)

This is class enrollment id which is to be deleted.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can be either the success message or the error message, based on the HTTP status.

a. Success message with class enrollment ID

b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 10010
classRegisterId: 82830

Output


```
{
   "statusType": "OK",
   "entity": "Deleted class enrollment (ID:82830) successfully",
   "entityType": "java.lang.String",
   "metadata": {},
   "status": 200
}
```


7. Password

7.1. Change Password

This API provides you the ability to change password of users in A-VIEW.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/changepassword.html

Input parameters

The input parameters are:

Parameter 1: User ID (userId)

This is the user ID of the user whose password needs to be changed

Parameter 2: New Password (newPassword)

This is the New Password of your user for the above user ID and the password of the user encrypted using SHA-1.
Data type: Varchar (20).

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can be either the success or the error message, based on the HTTP status.

Example

Input

userId: 2
newPassword: f95016b63cd4e3f74b3029639e7c7d9fde71d7db

Output


```
{
   "statusType": "OK",
   "entity": "User Password has updated successfully",
   "entityType": "java.lang.String",
   "metadata": {},
   "status": 200
}
```


8. File upload

This API can be used to upload documents to A-VIEW server.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/fileupload.html

Input parameters

The input parameters are:

Parameter 1: User ID (userId)

Data type: Integer

This is the user IDwhichshould be enrolled to the class id given below, of theA-VIEW class. This ID will be provided to you separately.

Parameter 2: Document (file)

This is the path of the file to be uploaded.
Supported file formats are "pdf", "ppt", "jpg", "doc", "docx", "pptx", "xlsx", "bmp", "gif", "xls", "txt".
The allowed size is up to 100 MB.

Parameter 3: ClassId (classId)

Data type: Integer
Specifies the class id to which the document needs to be uploaded. This ID is provided to you separately.

Parameter 4: Is Animated (isAnimated)

Specifies if the uploading file has any animation content or not. Value for this field is optional, if the value is left blank, API will take the default value "N". The allowed values are only "Y", "y", "N" and "n".

Parameter 5: overwrite (overwrite)

Specifies whether we want to overwrite the existing uploaded file or not. The allowed values are "Y", "y", "N" and "n". Value for this field is optional, if the value is left blank, API will take the default value "N".

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can be either the success message or the error message, based on the HTTP status.

Scenario

a. If the file already exist, and overwrite is ‘n’ or ‘N’ or blank, then the messages is “File already exists”
b. If the file doesn’t exist, and overwrite is blank or ‘n’ or ‘N’, then the messages is “File uploaded successfully”.
c. If the file already exist, and overwrite is ‘y’ or ‘Y’, then the messages is “File replaced successfully”.

Example:

Input

userId: 18765

file: Reference.txt

classId: 1426

isAnimated: N

overwrite:

Output

{

"statusType": "OK",

"entity": "File uploaded successfully",

"entityType": "java.lang.String",

"metadata": {},

"status": 200

}

9. Analytics

9.1. Login time and Logout time of A-VIEW user

This API provides you the ability to get the login and logout time of user in A-VIEW and session (lecture) entry/exit time, using the lecture Id.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/logindetailsforlecture.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided separately.

*Parameter 2:*Lecture ID (lectureId)

This is the Lecture ID for which login/logout time of all the participated users to be fetched

*Parameter 3:*User ID (userId)

This is the user ID of the user, which is unique for all users. Value for this field is optional, if the value is left blank, API will output details about all the users.

Output parameters

The following are the main output parameters:

a. HTTP status code

   This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

   This will be the HTTP status message of the process.

c. Entity

   This can either list out the login details or the error message, based on the HTTP status.

   a. Login details List

      This will list all the corresponding login details (user id, user name, A-VIEW login time, session login time, last action time in session and A-VIEW logout time).

"Last action time in session" can be considered as the session exit time, for now; but it may not be accurate in some cases.

b. Error message (if status code is 4xx or 5xx)

Example1

Input

adminId: 10010
lectureId: 76595
userId :

Output


```
{
   "statusType": "OK",
   "entity": [
[
        "userId: 43",
        "userName: kamal",
        "aviewLoginTime: 2014-01-20 16:23:29",
        "sessionLoginTime: 2014-01-20 16:23:41",
        "lastActionTimeInSession: 2014-01-20 18:29:38",
        "aviewLogoutTime: 2014-01-20 18:30:58"
    ],
    [
        "userId: 44",
        "userName: administrator",
        "aviewLoginTime: 2014-01-20 16:19:57",
        "sessionLoginTime: 2014-01-20 16:23:01",
        "lastActionTimeInSession: 2014-01-20 16:23:02",
        "aviewLogoutTime: Unknown"
    ]  ],
   "entityType": "java.util.ArrayList",
   "metadata": {},
   "status": 200
}
```


Example 2

Input

adminId: 10010
lectureId: 76595
userId : 43

Output


```
{
   "statusType": "OK",
   "entity": [
```

```
[
        "userId: 43",
        "userName: kamal",
        "aviewLoginTime: 2014-01-20 16:23:29",
        "sessionLoginTime: 2014-01-20 16:23:41",
        "lastActionTimeInSession: 2014-01-20 18:29:38",
        "aviewLogoutTime: 2014-01-20 18:30:58"
    ]  ],
  "entityType": "java.util.ArrayList",
  "metadata": {},
  "status": 200
}
```


9.2. Audio/Video Interaction Details

This API provides you the ability to get the users who have interacted with the Presenter for a given lecture, such as user id, username, how many times did the user interact with, interaction start dateTime, interaction end dateTime, user institute.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/userinteractiondetailsforlecture.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you separately.

*Parameter 2:*Lecture ID (lectureId)

This is the Lecture ID for which the interaction details are to be fetched.

*Parameter 3:*User ID (userId)

This is the user ID of the user, which is unique for all users. Value for this field is optional, if the value is left blank, API will output details about all the users.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can either list out the users interacted with the presenter or the error message, based on the HTTP status.

a. List interaction details with the presenter

This will list all the interaction details (interactedUserId ,interactedUserName, InteractionCount, InteractionStartDateTime, interactionEndDateTime and userInstitute)

Note: Since the interaction date time can be more than one for a particular user, it is separated by a "~,~"(tilde)symbol. In A-VIEW 3.7 client version, it was noted that in some unknown scenarios, interactionEndDateTime was not recorded in the databases.

b. Error message (if status code is 4xx or 5xx)

Example1

Input

adminId: 20091
lectureId: 89240
userId :

Output


```
{
   "statusType": "OK",
   "entity": [
[
        "interactedUserId: 8787",
        "interactedUserName: direct_guest59",
        "interactionCount: 2",
        "interactionStartDateTime: 2014-01-20 11:40:31 ~ , ~ 2014-01-20 17:29:52",
        "interactionEndDateTime: 2014-01-20 11:47:53 ~ , ~ 2014-01-20 17:30:34",
        "interactedUserInstitute: Amrita Test Institute" ,
        "publishingStatus: 1"
    ],
    [
        "interactedUserId: 8755",
        "interactedUserName: direct_guest27",
        "interactionCount: 10",
        "interactionStartDateTime: 2014-01-20 09:49:26 ~ , ~ 2014-01-20 09:50:39 ~ , ~
        2014-01-20 09:54:04 ~ , ~ 2014-01-20 09:55:52 ~ , ~ 2014-01-20 10:00:40 ~ , ~
        2014-01-20 10:02:05 ~ , ~ 2014-01-20 18:13:46 ~ , ~ 2014-01-20 10:05:05 ~ , ~
        2014-01-20 15:18:30 ~ , ~ 2014-01-20 17:04:11",
```

```
            "interactionEndDateTime: 2014-01-20 09:50:36 ~ , ~ 2014-01-20 09:53:17 ~ , ~
            2014-01-20 09:55:19 ~ , ~ 2014-01-20 09:56:56 ~ , ~ 2014-01-20 10:01:33 ~ , ~
            2014-01-20 10:04:44 ~ , ~ 2014-01-20 10:06:42 ~ , ~ 2014-01-20 18:15:59 ~ , ~
            2014-01-20 15:19:56 ~ , ~ 2014-01-20 17:04:37",
            "interactedUserInstitute: Amrita Test Institute" ,
             "publishingStatus: 2"
        ]
      ],
        "entityType": "java.util.ArrayList",
        "metadata": {},
        "status": 200
    }
```


Example 2

Input

```
adminId: 20091
lectureId: 89240
userId     : 8787
```

Output


```
{
    "statusType": "OK",
    "entity": [
        [
            "interactedUserId: 8787",
            "interactedUserName: direct_guest59",
            "interactionCount: 2",
            "interactionStartDateTime: 2014-01-20 11:40:31 ~ , ~ 2014-01-20 17:29:52",
            "interactionEndDateTime: 2014-01-20 11:47:53 ~ , ~ 2014-01-20 17:30:34",
            "interactedUserInstitute: Amrita Test Institute,
            "publishingStatus: 2"
        ]
    ],
    "entityType": "java.util.ArrayList",
    "metadata": {},
    "status": 200
}
```


9.3. Question Interaction Information

This API is to get the list of questions asked during a particular lecture, if any.This also provides the details of the users who asked the questions, and the votes for the questions, if any, and whether the presenter had marked the questions as answered.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/questiondetailsforlecture.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you separately.

*Parameter 2:*Lecture ID (lectureId)

This is the Lecture ID for which the question details need to be fetched.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can either list out the details of questions or error message, based on the HTTP status.

a. List of question details

This will list all the corresponding users question details (question text, user who raised the question, its time, number of votes received and whether the question was answered by the presenter)

Note: If the same question (literally)was asked by more than one user, the username will be added to questionRaisedBy and separated by a "," symbol.

b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 18756
lectureId: 86042

Output

{
"statusType": "OK",

```
  "entity": [
    [
      "questionText: we want one more interaction in Jan",
      "questionRaisedBy: localt13",
      "questionRaisedDateTime: 2013-12-07 16:35:53",
      "questionVote: 0",
      "isQuestionAnswered: No",
      "questionAnswerDateTime: NA"
    ],
[
      "questionText: hi",
      "questionRaisedBy:  localt15",
      "questionRaisedDateTime: 2013-12-07 16:32:53",
      "questionVote: 2",
      "isQuestionAnswered: Yes",
      "questionAnswerDateTime: 2013-12-07 16:34:15"
    ]
  ],
  "entityType": "java.util.ArrayList",
  "metadata": {},
  "status": 200
}
```


9.4. Network reconnection

This API provides you the ability to get the network connection rejection and failure occurrences during a lecture.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/reconnectiondetailsforlecture.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you separately.

*Parameter 2:*Lecture ID (lectureId)

This is the Lecture ID of which the connection rejection and failure occurrences are to be fetched.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can either list out the network rejection/failure occurrences or error message, based on the HTTP status.

a. List of details

This will list all the corresponding networkreconnection details occurring during the sessions (user name, failure type, failure time and success time (if the network connection was reestablished)).

b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 10010
lectureId: 86015

Output


```
{
   "statusType": "OK",
   "entity": [
[
        "userName: direct_guest6",
        "failureType: ConnectionFail",
        "failureTime: 2014-01-20 12:10:23",
        "sucessTime: 2014-01-20 12:11:44"
    ],
    [
        "userName: direct_guest6",
        "failureType: ConnectionFail",
        "failureTime: 2014-01-20 13:22:57",
        "sucessTime: 2014-01-20 13:22:59"
 ],
   "entityType": "java.util.ArrayList",
   "metadata": {},
   "status": 200
```

}

9.5. User participation by Class

This API provides you the ability to get all the details of a user with respect to a given class. For example, which all sessions the user attended, login and logout time, audio/video interaction details, question interaction details etc.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/userparticipationbyclass.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you separately.

*Parameter 2:*User ID (userId)

This is the user ID of the user, of whom we need to fetch the details.

*Parameter 3:*Class ID (classId)

This is the ID of the class to which the user is enrolled to.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can either list out the all details or error message, based on the HTTP status.

a. Expected result

This will list all the corresponding lectures/sessions of the class to which the user logged in, with the log in time, audio/video interaction and question details.

b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 18756
userId: 18785
classId : 364

Output

{
"statusType": "OK",
"entity": [
[
"lecture : QEEE Workshop 1 2013-12-7",
"lectureId : 86042",
"loginDetails : [[userId: 18785, userName: localt8, aviewLoginTime: 2013-12-07 16:19:07, sessionLoginTime: 2013-12-07 16:19:13, lastActionTimeInSession: 2013-12-07 16:38:23, aviewLogoutTime: 2013-12-07 16:21:16]]",
"userInteractionDetails : [[interactedUserId: 18785, interactedUserName: localt8, interactionCount: 1, interactionStartDateTime: 2013-12-07 16:29:33, interactionEndDateTime: 2013-12-07 16:30:21, interactedUserInstitute: RTBI]]",
"questionDetails : [[[questionText: how are you, questionRaisedDateTime: 2013-12-07 16:35:20, questionVote: 1, isQuestionAnswered: No, questionAnswerDateTime: NA], [questionText: hello sir, questionRaisedDateTime: Unkown, questionVote: 0, isQuestionAnswered: No, questionAnswerDateTime: NA]]]"
],
[
"lecture : QEEE Workshop 1 2013-12-20",
"lectureId: 86105",
"loginDetails : No login details are available for this lecture id",
"userInteractionDetails : No interaction details happened in this session",
"questionDetails : No question details happened in this session"
]
]
"entityType": "java.util.ArrayList",
"metadata": {},
"status": 200
}

9.6. Live Sessions

This API provides the ability to get all the live sessions (lectures) scheduled for now.

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/livesessions.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you separately.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can either list out all the live lecture details or error message, based on the HTTP status.

a. List of lecture details

This will list all the live sessions with their details (lecture id, lecture name, start time and end time).

b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 18756

Output


```
{
   "statusType": "OK",
   "entity": [
         [
          "lecture: QEEE Workshop 1 2013-12-7",
         "lectureId: 86042",
         "startTime: 14:00:00",
         "endTime: 14:55:00"
         ]
    ],
   "entityType": "java.util.ArrayList",
```

```
    "metadata": {},
    "status": 200
}
```


9.7. Attending users

This API provides the ability to get all the participating users for a given session (lecture).

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/attendingusers.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you separately.

*Parameter 2:*Lecture ID (lectureId)

This is the session (lecture) ID of which the attending users are to be fetched.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can either list out all the attending users with their details or error message based on the HTTP status.

a. List of attending users

This will list all the corresponding users who have attended the session till the execution time (userID and username).

b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 18756
lectureId: 86015

Output


```
{
   "statusType": "OK",
   "entity": [
                [
                   "userName: Test_api_user1",
                   "userId: 22616" ]
          ],
   "entityType": "java.util.ArrayList",
   "metadata": {},
   "status": 200
}
```


9.8. Audio/Video publishing status of users

This API provides the ability to get the audio/video publishing status of all theusers for a given session (lecture).

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/audiovideopublishingstatus.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you separately.

*Parameter 2:*Lecture ID (lectureId)

This is the session (lecture) ID of which the status has to be fetched.

Output parameters

The following are the main output parameters:

a. HTTP status code

This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

This will be the HTTP status message of the process.

c. Entity

This can either list out the audio/video publishing status of the users or error message, based on the HTTP status.

a. Audio/video publishing status of users

This will list the audio/video publishing status of the users (userID, userName and publishingStatus). Publishing status can have values

0 - if not publishing audio/video

1- if publishing only audio

2 - if publishing both audio and video

The publishing value ,will be the latest publishing status when the function is fired.

b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 18756
lectureId: 86015

Output


```
{
   "statusType": "OK",
    "entity": [
                    [
                       "userName: Test_api_user1",
                       "userId: 22616",
                       "publishingStatus: 0"
                    ],
                    [
                       "userName: Test_api_user2",
                       "userId: 22617",
                      "publishingStatus: 2"
                       ],
```

```
                    [
                        "userName: Test_api_user4",
                        "userId: 22619",
                        "publishingStatus: 1"
                    ]
                ],
    "entityType": "java.util.ArrayList",
    "metadata": {},
    "status": 200
}
```


9.9. Users who had chat interaction

This API provides you the ability to get the list of users who had chat interaction for a given session (lecture).

URL

The post request can be passed to the below URL:

http://<*IP Address*>:<*Port Number*>/aview/chatinteractedusers.html

Input parameters

The input parameters are:

Parameter 1: Admin ID (adminId)

This is the user ID of your A-VIEW admin user. This will be provided to you separately.

Parameter 2: Lecture ID (lectureId)

This is the session (lecture) ID of which the list of users who did chat interaction are to be fetched.

Output parameters

The following are the main output parameters:

a. HTTP status code

   This will be a 3 digit number denoting the status of the API call.

b. HTTP status Message

   This will be the HTTP status message of the process.

c. Entity

This can either list out all the users who had chat interaction during the session or error message, based on the HTTP status.

a. List of users who had chat interaction

This will list all the users who had chat interaction (userID, userName, chatText, chatInteractionDateTime).

b. Error message (if status code is 4xx or 5xx)

Example

Input

adminId: 18756
lectureId: 86015

Output


```
{
   "statusType": "OK",
   "entity": [
         [
          "userName: Test_api_user1",
          "userId: 22616",
          "chatText: How h r u",
          "chatInteractionDateTime: 2014-01-20 11:40:32"
         ],
         [
          "userName: Test_api_user2",
          "userId: 22617",
          "chatText: hi",
         "chatInteractionDateTime: 2014-01-20 11:41:31"
          ],
         [
          "userName: Test_api_user4",
          "userId: 22619",
          "chatText: good",
         "chatInteractionDateTime: 2014-01-20 11:40:31"
          ]
          ],
   "entityType": "java.util.ArrayList",
   "metadata": {},
   "status": 200
}
```

HTTP Status Codes and Messages

If the code is 2xx, it indicates a success case and if the code is 4xx or 5xx some error has occurred. These are some of the main standard HTTP status codes related to the above APIs currently.

1. Success 2xx

These codes indicate success.

OK 200
The request was fulfilled.

CREATED 201
Following a POST command, this indicates success, but the textual part of the response line indicates the URI by which the newly created document should be known.

No Response 204
Server has received the request but there is no information to send back.

2. Error 4xx, 5xx

The 4xx codes are intended for cases in which the client seems to have erred, and the 5xx codes for the cases in which the server is aware that the server has erred. It is impossible to distinguish these cases in general, so the difference is only informational.

Bad request 400
The request had bad syntax or was inherently impossible to be satisfied.

Unauthorized 401
The parameter to this message gives a specification of authorization schemes which are acceptable. The client should retry the request with a suitable Authorization header.

Forbidden 403
The request is for something forbidden. Authorization will not help.

Not found 404
The server has not found anything matching the URI given

Internal Error 500
The server encountered an unexpected condition which prevented it from fulfilling the request.

Not implemented 501
The server does not support the facility required.