Web Services User Manual

**Submitted by:** *Project Group 3*

[1 Create Admin User](#h.1n75aqdmacow)

[2 Student User Registration -POST /api/v1.0/students](#h.vcjdhsdbcu9k)

[3 Get Student by ID - GET /api/v1.0/students/{id}](#h.bebvv7hvnfoo)

[4 Get All Students - GET /api/v1.0/students](#h.x0cmqv8rhkym)

[5 Student User Updates - PUT /api/v1.0/students/{id}](#h.ny78jm9savii)

[6 Student User Gets List of Courses - GET /api/v1.0/courses](#h.ncw1zajzx3j6)

[7 Student User Queries For Courses with filter criteria - GET /api/v1.0/courses?filter=<queryString>](#h.g5urpqoav8w7)

[8 Student User Queries for Course by ID - GET /api/v1.0/courses/{id}](#h.wm9onhs1m8h1)

[9 Student User Account Deletion - DELETE /api/v1.0/students/{id}](#h.d56binbtczn5)

[10 Course Addition - POST /api/v1.0/courses](#h.10qq0pr56xxg)

[11 Course Update - PUT /api/v1.0/courses/{id}](#h.4dvd5ax58qy5)

[12 Course Deletion - DELETE /api/v1.0/courses/{id}](#h.bcvjkytebb5j)

This document describes all use cases for the different web services endpoints available in our implementation.

We are giving the screenshot for browser based REST client and the corresponding cURL command to be used to access the REST endpoints.

All of the REST endpoints have following root path: **/api/v1.0**

1. **Students resource endpoint is => /api/v1.0/students -- this is the only endpoint accessible to students.**
2. **Courses resource endpoint is => /api/v1.0/courses -- only Admin user can access.**

# 1 Create Admin User

Admin user is created for the system using a special CLI and is not exposed to users of the system.

**$ java -jar target/webservice-1.0-SNAPSHOT.jar create-user config.yml**

Enter first name:

Admin

Enter last name:

Admin

Enter email:

admin@sjsu.edu

Enter password:

password

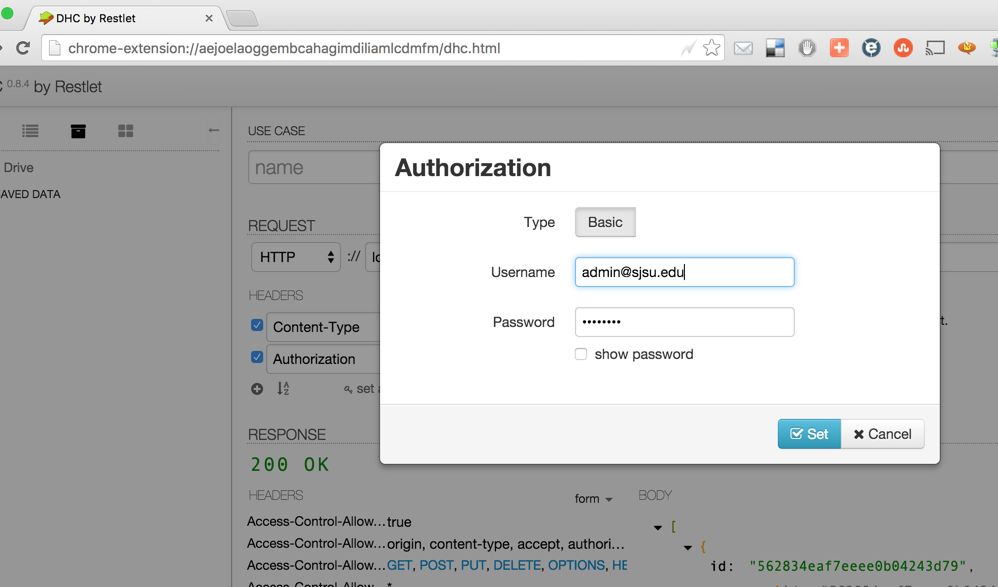
Re-enter password:

password

User successfully created:

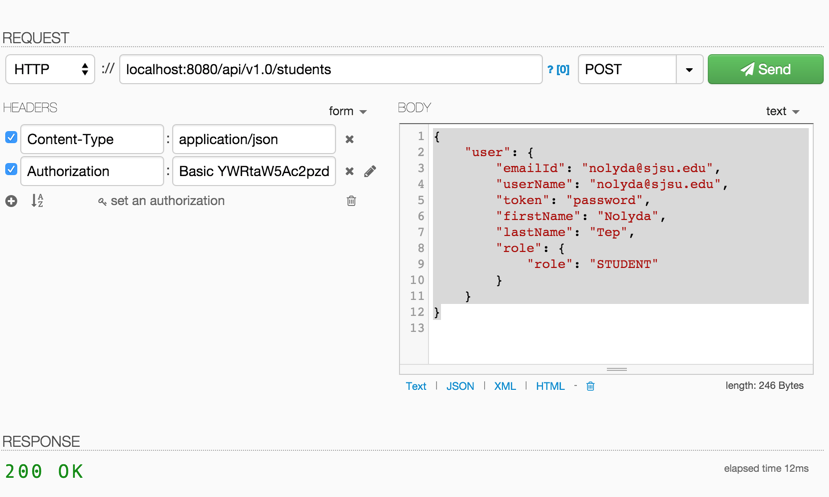
{"id":"56297a43f7eeee195caf8c78","emailId":"admin@sjsu.edu","userName":"admin@sjsu.edu","token":"password","firstName":"Admin","lastName":"Admin","role":{"role":"ADMIN"},"lastUpdated":1445558870808}

Once the Admin user is created, this user can add courses to the system using the REST API.



The credentials for a user will be used to create Authorization header in the request.

# 2 Student User Registration -POST /api/v1.0/students



|  |  |
| --- | --- |
| URI | <http://localhost:8080/api/v1.0/students> |
| Method | POST |
| Content Type | application/json |
| Payload | {  “user”: {  "emailId": "foo@sjsu.edu",  "userName": "foo@sjsu.edu",  "token": "password",  "firstName": "Foo",  "lastName": "Bar",  "role": {  "role": "STUDENT"  }  }  } |
| Response | Gets the newly created student JSON. |
| **$ curl -X POST -H "content-type: application/json" -d @./create\_student.json** [**http://localhost:8080/api/v1.0/students**](http://localhost:8080/api/v1.0/students)  {"entity":{"id":"562c8fb3f7eeee3cf4ca3b4e","user":{"id":"562c8fb3f7eeee3cf4ca3b4f","emailId":"java.guy@sjsu.edu","userName":"java.guy@sjsu.edu","token":"password","firstName":"Java","lastName":"Guy","role":{"role":"STUDENT"},"lastUpdated":1445760947437},"courseRefs":[],"lastUpdated":1445760947448},"variant":{"language":null,"mediaType":{"type":"application","subtype":"json","parameters":{},"wildcardType":false,"wildcardSubtype":false},"encoding":null,"languageString":null},"annotations":[],"language":null,"encoding":null,"mediaType":{"type":"application","subtype":"json","parameters":{},"wildcardType":false,"wildcardSubtype":false}} | |

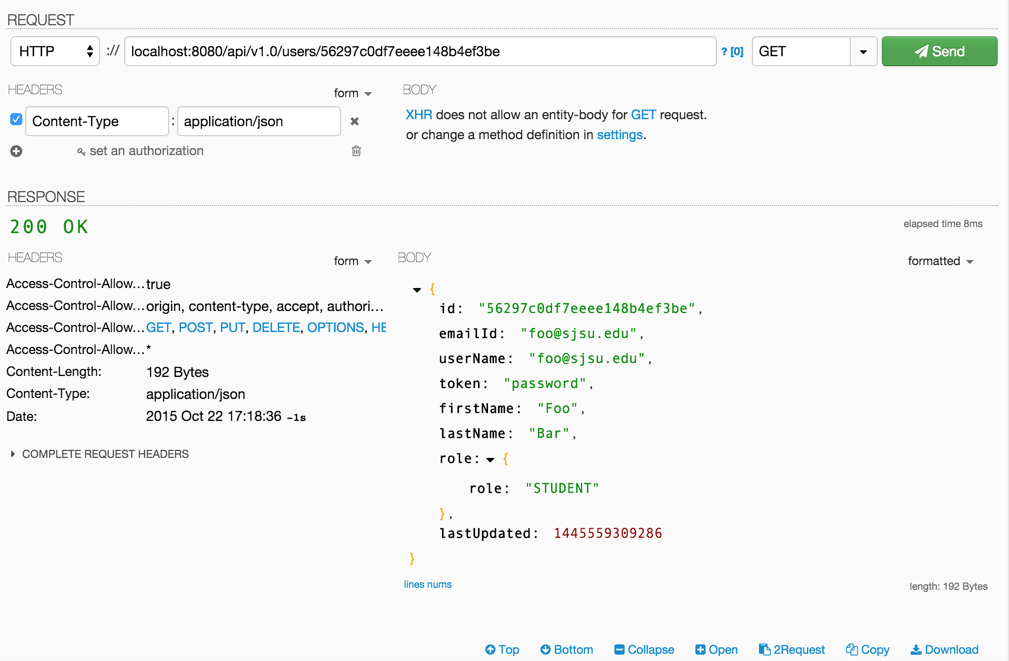
The above method creates a new user in the system with role STUDENT.

The created student has ID [56297c0df7eeee148b4ef3be](http://localhost:8080/api/v1.0/users/56297c0df7eeee148b4ef3be) as seen in the response.

This endpoint does not require the Authorization header.

Anyone can sign up as a student in the system. Upon sign up, student will require to provide the basic auth credentials for other requests.

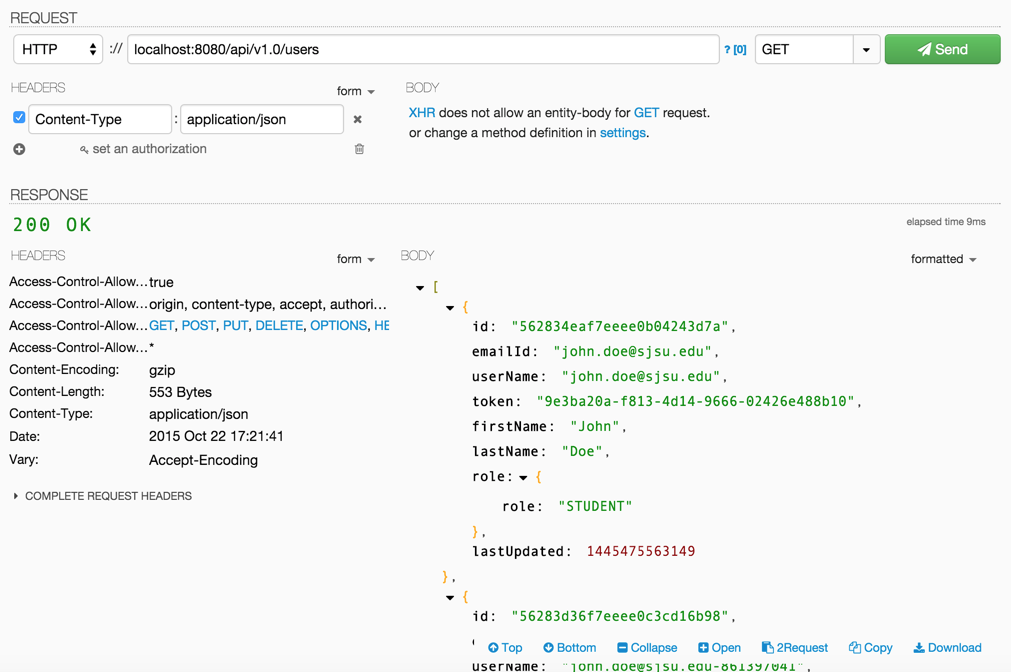
# 3 Get Student by ID - GET /api/v1.0/students/{id}



|  |  |
| --- | --- |
| URI | [http://localhost:8080/api/v1.0/students/56297c0df7eeee148b4ef3be](http://localhost:8080/api/v1.0/users/56297c0df7eeee148b4ef3be) |
| Method | GET |
| Content Type | application/json |
| Authorization | Basic YWRtaW5Ac2pzdS5lZHU6cGFzc3dvcmQ= |
| Response | Gets the specified student by ID |
| **$ curl -X GET -H "content-type: application/json" -u "java.guy@sjsu.edu:password" http://localhost:8080/api/v1.0/students/562c8fb3f7eeee3cf4ca3b4e**  {"id":"562c8fb3f7eeee3cf4ca3b4e","user":{"id":"562c8fb3f7eeee3cf4ca3b4f","emailId":"java.guy@sjsu.edu","userName":"java.guy@sjsu.edu","token":"password","firstName":"Java","lastName":"Guy","role":{"role":"STUDENT"},"lastUpdated":1445760947437},"courseRefs":[],"lastUpdated":1445760947448} | |

Note: The Authorization header is required or else pass the user credentials with -u option of cURL. The credentials could be of student or admin.

# 4 Get All Students - GET /api/v1.0/students



URI: <http://localhost:8080/api/v1.0/users>

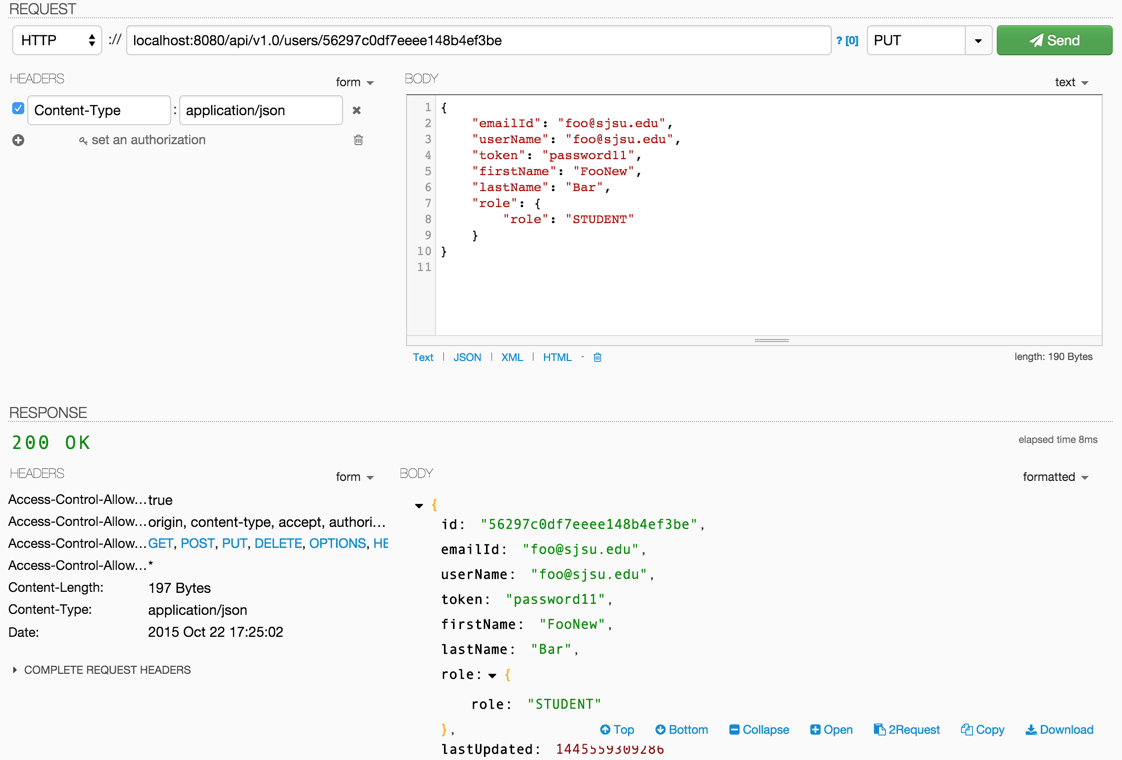
Method: GET

Content-Type: application/json

|  |  |
| --- | --- |
| URI | [http://localhost:8080/api/v1.0/](http://localhost:8080/api/v1.0/users/56297c0df7eeee148b4ef3be)students |
| Method | GET |
| Content Type | application/json |
| Authorization | Basic YWRtaW5Ac2pzdS5lZHU6cGFzc3dvcmQ= |
| Response | Gets all users in the system. |
| **$ curl -X GET -H "content-type: application/json" -u "java.guy@sjsu.edu:password" http://localhost:8080/api/v1.0/students**  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>  <title>Error 403 Forbidden</title>  </head>  <body><h2>HTTP ERROR 403</h2>  <p>Problem accessing /api/v1.0/students. Reason:  <pre> Forbidden</pre></p><hr><i><small>Powered by Jetty://</small></i><hr/>  </body>  </html> | |
| **$ curl -X GET -H "content-type: application/json" -u "admin@sjsu.edu:password"** [**http://localhost:8080/api/v1.0/students**](http://localhost:8080/api/v1.0/students)  ***Returns a list of students...*** | |

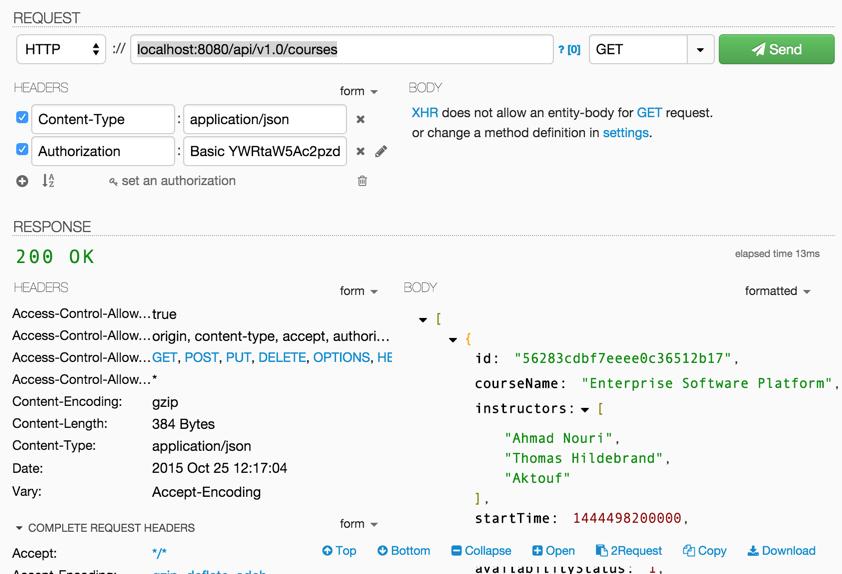
\* Note: This command can only be run by Admin user. If a student user executes this API then a 403 (Forbidden) error will be thrown (as shown above in cURL command o/p).

# 5 Student User Updates - PUT /api/v1.0/students/{id}



|  |  |
| --- | --- |
| URI | <http://localhost:8080/api/v1.0/students/562c8650f7eeee2204f05acd> |
| Method | PUT |
| Content Type | application/json |
| Authorization | Basic YWRtaW5Ac2pzdS5lZHU6cGFzc3dvcmQ= |
| Payload | {  "user": {  "emailId": "nolyda@sjsu.edu",  "userName": "nolyda@sjsu.edu",  "token": "password2",  "firstName": "Nolyda",  "lastName": "Tep",  "role": {  "role": "STUDENT"  }  },  "courseRef" : {  "enrollCourseName": "Cloud Technologies",  "unEnrollId": "56283cdbf7eeee0c36512b17"  }  } |
| Response | Gets the updated student. |
| **$ curl -X PUT -H "content-type: application/json" -u "java.guy@sjsu.edu:password" -d @./update\_student.json** [**http://localhost:8080/api/v1.0/students/562c8fb3f7eeee3cf4ca3b4e**](http://localhost:8080/api/v1.0/students/562c8fb3f7eeee3cf4ca3b4e)  {"id":"562c8fb3f7eeee3cf4ca3b4e","user":{"id":"562c8fb3f7eeee3cf4ca3b4f","emailId":"java.guy@sjsu.edu","userName":"java.guy@sjsu.edu","token":"password2","firstName":"Watsh","lastName":"Rajneesh","role":{"role":"STUDENT"},"lastUpdated":1445760947437},"courseRefs":[{"id":"56284226f7eeee0ce82e4e57","courseName":"Cloud Technologies","instructors":["Ahmad Nouri","Thomas Hildebrand","Aktouf"],"startTime":1444498200000,"endTime":1447189200000,"availabilityStatus":1,"maxCapacity":20,"price":200.0,"location":"Santa Clara, CA","keywords":["Java","MongoDB","REST"],"lastUpdated":1445478950536}],"lastUpdated":1445760947448}  where, **update\_student.json**:  {  "user": {  "emailId": "java.guy@sjsu.edu",  "userName": "java.guy@sjsu.edu",  "token": "password2",  "firstName": "Watsh",  "lastName": "Rajneesh",  "role": {  "role": "STUDENT"  }  },  "courseRef" : {  "enrollCourseName": "Cloud Technologies",  "unEnrollId": "56283cdbf7eeee0c36512b17"  }  }  **Note**: Enrolling to course can be performed with following attributes in “**courseRef”**:   1. enrollCourseName or 2. enrollId   Similarly, un enroll from course can be performed with:   1. unEnrollCourseName or 2. unEnrollId   The “user” JSON can be used to update one or more student user attributes. Not all user attributes need to be specified but only the ones being updated.  Only admin user can update any student’s profile.  Student user can only update his/her profile or course enrollment/unenrollment. If attempting to update another student data, a Forbidden (403) error is returned. | |

# 6 Student User Gets List of Courses - GET /api/v1.0/courses



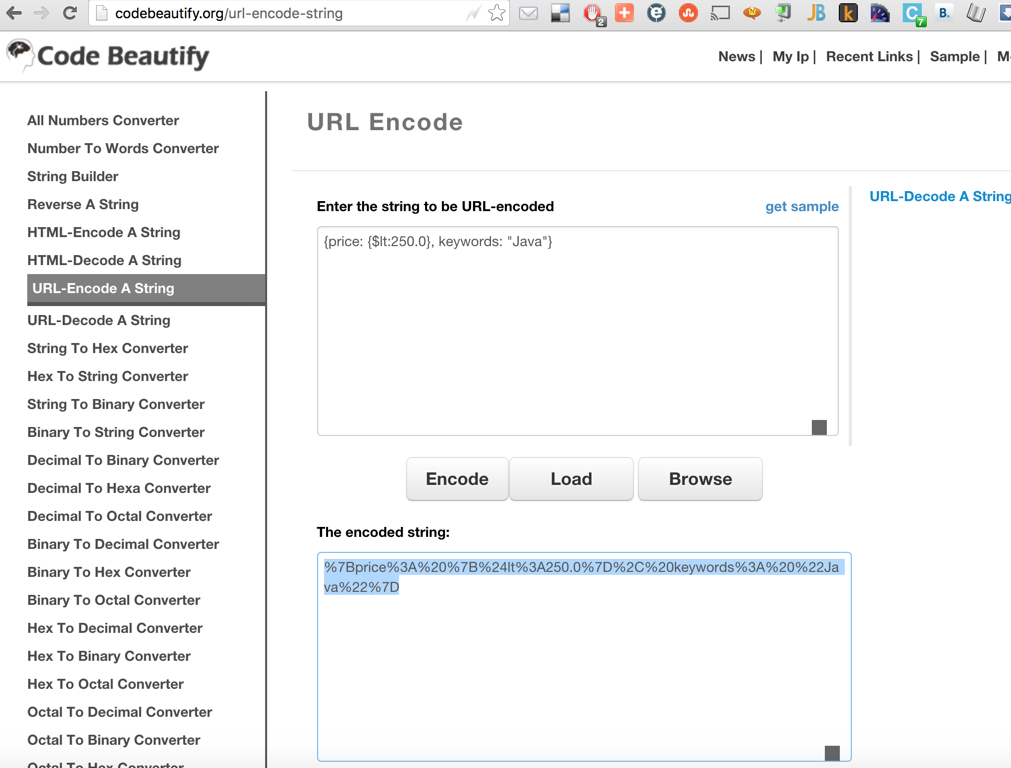
|  |  |
| --- | --- |
| URI | http://localhost:8080/api/v1.0/courses |
| Method | GET |
| Response | List of courses in the system. |
| **curl -i -X GET \**  **-H "Content-Type:application/json" \**  **-H "Authorization:Basic YWRtaW5Ac2pzdS5lZHU6cGFzc3dvcmQ=" \**  **'http://localhost:8080/api/v1.0/courses'** | |

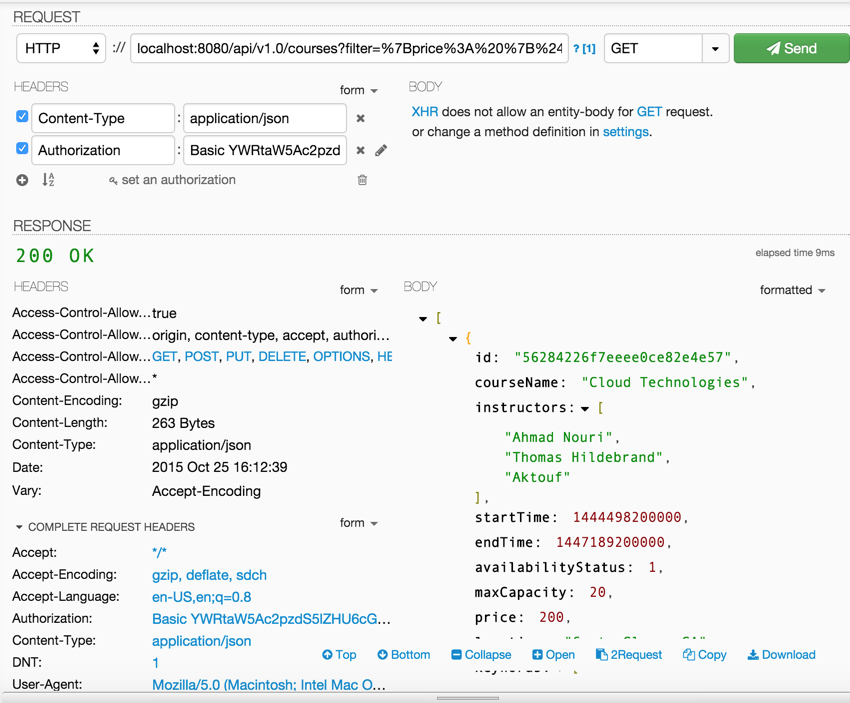
# 7 Student User Queries For Courses with filter criteria - GET /api/v1.0/courses?filter=<queryString>

The filter query string param will be an URL encoded JSON document. This JSON document is passed to mongodb as query string and the results are returned.

To try it with a browser client, we need to first URL encode the query string. There are several online sites to generate the URL encoded string.

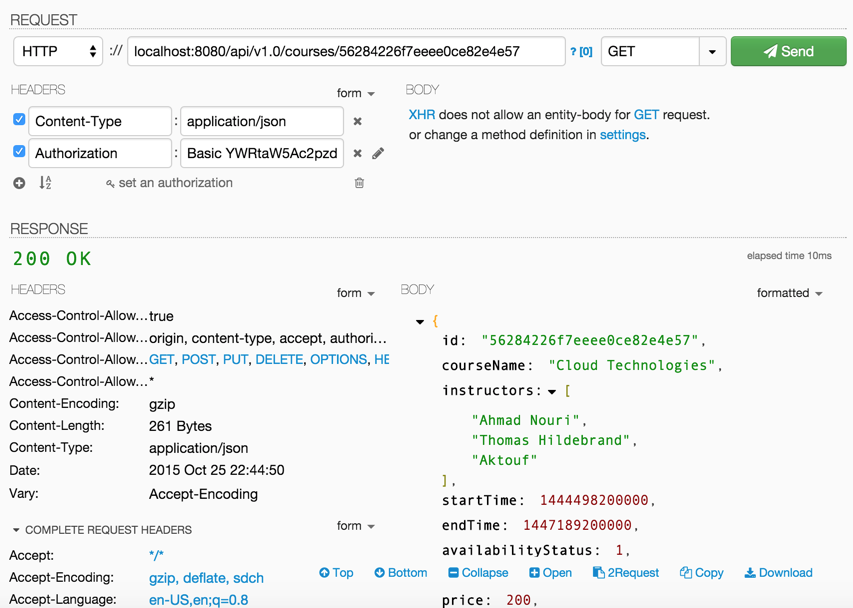
In code this encoding can be achieved in JavaScript as encodeURIComponent(url) method.





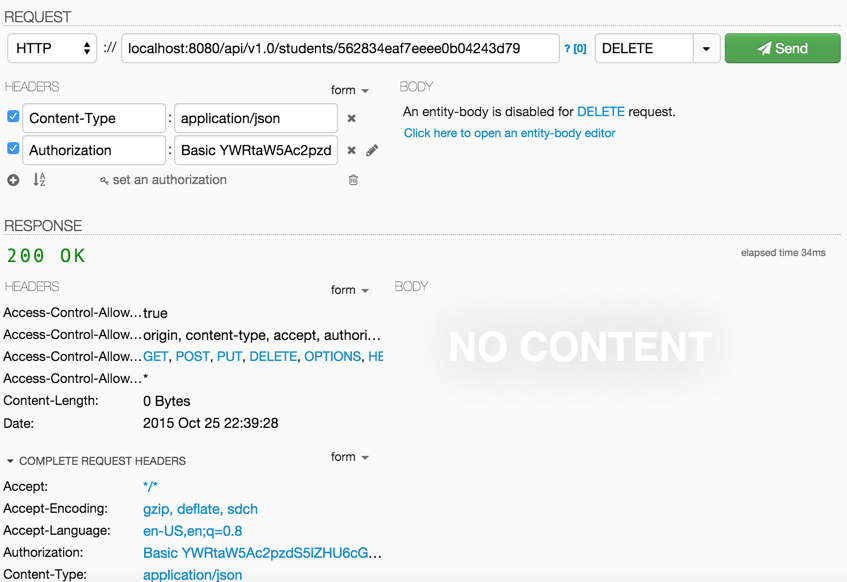
|  |  |
| --- | --- |
| URI | http://localhost:8080/api/v1.0/courses?filter=%7Bprice%3A%20%7B%24lt%3A250.0%7D%2C%20keywords%3A%20%22Java%22%7D  The above “filter” param value represents query string and is URL encoded form of the following query:  ***{price: {$lt:250.0}, keywords: "Java"}***  Give all courses which cost less than $250.0 and are Java courses. |
| Method | GET |
| Response | Shows only the filtered set of courses that satisfy the filter criteria. |
| **curl -i -X GET \**  **-H "Content-Type:application/json" \**  **-H "Authorization:Basic YWRtaW5Ac2pzdS5lZHU6cGFzc3dvcmQ=" \**  **'http://localhost:8080/api/v1.0/courses?filter=%7Bprice%3A+%7B%24lt%3A250.0%7D%2C+keywords%3A+%22Java%22%7D'** | |

# 8 Student User Queries for Course by ID - GET /api/v1.0/courses/{id}



|  |  |
| --- | --- |
| URI | <http://localhost:8080/api/v1.0/courses/56284226f7eeee0ce82e4e57> |
| Method | GET |
| Result | Gets the specified course by ID. |
| **curl -i -X GET \**  **-H "Content-Type:application/json" \**  **-H "Authorization:Basic YWRtaW5Ac2pzdS5lZHU6cGFzc3dvcmQ=" \**  **'http://localhost:8080/api/v1.0/courses/56284226f7eeee0ce82e4e57'** | |

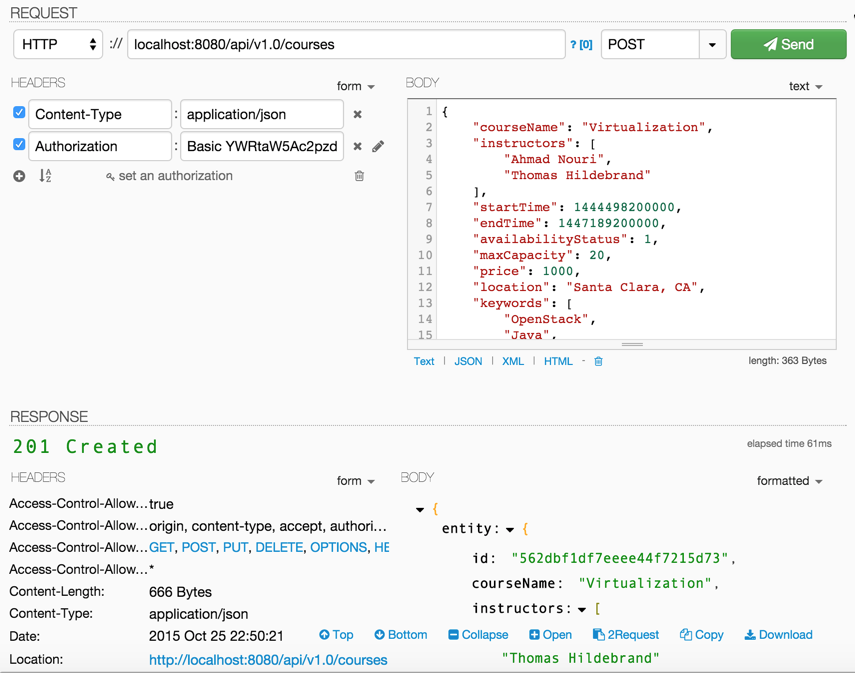
# 9 Student User Account Deletion - DELETE /api/v1.0/students/{id}



|  |  |
| --- | --- |
| URI | <http://localhost:8080/api/v1.0/students/562834eaf7eeee0b04243d79> |
| Method | DELETE |
| Result | The specified student by ID is deleted. |
| **curl -i -X DELETE \**  **-H "Content-Type:application/json" \**  **-H "Authorization:Basic YWRtaW5Ac2pzdS5lZHU6cGFzc3dvcmQ=" \**  **'http://localhost:8080/api/v1.0/students/562834eaf7eeee0b04243d79'** | |

\* Note: This command can only be run by Admin user. If a student user executes this API then a 403 (Forbidden) error will be thrown.

# 10 Course Addition - POST /api/v1.0/courses



|  |  |
| --- | --- |
| URI | <http://localhost:8080/api/v1.0/courses> |
| Method | POST |
| Payload | {  "courseName": "Virtualization",  "instructors": [  "Ahmad Nouri",  "Thomas Hildebrand"  ],  "startTime": 1444498200000,  "endTime": 1447189200000,  "availabilityStatus": 1,  "maxCapacity": 20,  "price": 1000,  "location": "Santa Clara, CA",  "keywords": [  "OpenStack",  "Java",  "REST"  ]  } |
| Result | Specified course is added. |
| **curl -i -X POST \**  **-H "Content-Type:application/json" \**  **-H "Authorization:Basic YWRtaW5Ac2pzdS5lZHU6cGFzc3dvcmQ=" \**  **-d \**  **'{**  **"courseName": "Virtualization",**  **"instructors": [**  **"Ahmad Nouri",**  **"Thomas Hildebrand"**  **],**  **"startTime": 1444498200000,**  **"endTime": 1447189200000,**  **"availabilityStatus": 1,**  **"maxCapacity": 20,**  **"price": 1000,**  **"location": "Santa Clara, CA",**  **"keywords": [**  **"OpenStack",**  **"Java",**  **"REST"**  **]**  **}**  **' \**  **'http://localhost:8080/api/v1.0/courses'** | |

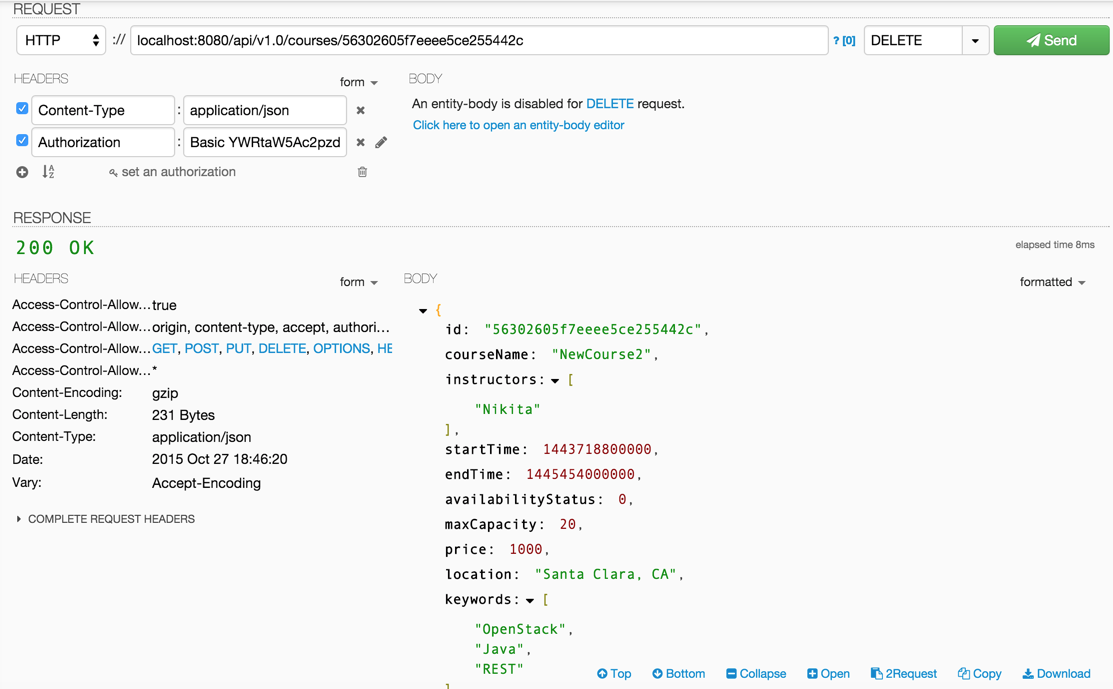
\* Note: This command can only be run by Admin user. If a student user executes this API then a 403 (Forbidden) error will be thrown.

# 11 Course Update - PUT /api/v1.0/courses/{id}

|  |  |
| --- | --- |
| URI | <http://localhost:8080/api/v1.0/courses/562dbf1df7eeee44f7215d73> |
| Method | PUT |
| Payload | {  "courseName": "NewCourse2",  "instructors": [  "Nikita"  ],  "startTime": "10-01-2015 10:00",  "endTime": "10-21-2015 12:00",  "availabilityStatus": 0,  "maxCapacity": 20,  "price": 1000,  "location": "Santa Clara, CA",  "keywords": [  "OpenStack",  "Java",  "REST"  ]  } |
| **curl -i -X PUT \**  **-H "Content-Type:application/json" \**  **-H "Authorization:Basic YWRtaW5Ac2pzdS5lZHU6cGFzc3dvcmQ=" \**  **-d \**  **'{**  **"courseName": "NewCourse2",**  **"instructors": [**  **"Nikita"**  **],**  **"startTime": "10-01-2015 10:00",**  **"endTime": "10-21-2015 12:00",**  **"availabilityStatus": 0,**  **"maxCapacity": 20,**  **"price": 1000,**  **"location": "Santa Clara, CA",**  **"keywords": [**  **"OpenStack",**  **"Java",**  **"REST"**  **]**  **}**  **' \**  **'http://localhost:8080/api/v1.0/courses/562dbf1df7eeee44f7215d73'** | |

\* Note: This command can only be run by Admin user. If a student user executes this API then a 403 (Forbidden) error will be thrown.

# 12 Course Deletion - DELETE /api/v1.0/courses/{id}



|  |  |
| --- | --- |
| URI | <http://localhost:8080/api/v1.0/courses/56284858f7eeee0d7d63a507> |
| Method | DELETE |
| Result | Specified course by ID is deleted. |
| **curl -i -X DELETE \**  **-H "Content-Type:application/json" \**  **-H "Authorization:Basic YWRtaW5Ac2pzdS5lZHU6cGFzc3dvcmQ=" \**  **'http://localhost:8080/api/v1.0/courses/56284858f7eeee0d7d63a507'** | |

When the course is deleted then the courseRefs for the students that were enrolled in that course is cascade deleted. This is handled by our application code as MongoDB does not support this behavior.

\* Note: This command can only be run by Admin user. If a student user executes this API then a 403 (Forbidden) error will be thrown.