

API Documentation

Project Title: Online Training Management System : A Case Study of BJIT

API 01: Sign-in / Authenticate

Description: This is the API that will authenticate the user. Users have to pass their username (email) and password to this API. After a successful request, the API will send a response with one access token (JWT), one refresh token and some public details about the user.

URL: <http://localhost:8080/api/v1/auth/authenticate>

Method: POST

Roles Needed: N/A

Request Body:

```
1 {  
2   "username": "rahman.rezaur@bjitgroup.com",  
3   "password": "iamuser"  
4 }
```

Response: 200 Ok

```
1 {  
2   "type": "Bearer",  
3   "token":  
4     "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJyYWhtYW4ucmV6YXVyQGJqaXRncm91c5jb20iLCJST0xFUyI6WyJST0xFX1VTRVIiXSwiZXhwIjoxNjQwNzg3NTY4LCJpYXQiOiJlNDA3MDEh9.RpwSfkoakIl6dea-o9u0-SbbjNovun3dqBD9iGG7_K0",  
5   "refreshToken": "98bea30f-af34-439d-ba76-cf88d4dca464",  
6   "userId": "11364",  
7   "username": "rahman.rezaur@bjitgroup.com",  
8   "fullName": "Rezaur Rahman",  
9   "roles": [  
10    "ROLE_USER"  
11  ]  
12 }
```

API 02: Token refresh API

Description: Our application uses JWT token for role based authentication and authorization. This JWT access token has a relatively short lifespan for security reasons. So we need to refresh this token after some time. For this purpose, we have to use this token refresh API. All we have to do is pass our refresh token with the request. And if that refresh token is valid and is not expired yet, then we will get a new access token with the refresh token as response.

URL: <http://localhost:8080/api/v1/auth/refreshToken>

Method: POST

Roles Needed: N/A

Request Body:

```
1 {  
2   "refreshToken": "b894e93f-8f81-499a-a373-b922909a955c"  
3 }
```

Response: 201 Created

```
1 {  
2   "tokenType": "Bearer",  
3   "accessToken":  
4     "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJzYWxlZWVpbkBiaml0Z3JvdXAuY29tIiwiaUkiOiJMRV9BRE1JTjJdLCJleHAiOiJlE2NDA2NzI4NjcsImh0CI6MTY0MDU4NjQ2N30.VXKh0wZC892ZbZkTpOXccsvngDJw5-ag6B3K0UuMyPY",  
5   "refreshToken": "b894e93f-8f81-499a-a373-b922909a955c"  
}
```

API 03: Invalidate token of a particular user

Description: This API is used to invalidate all tokens and sessions of a particular user. This is used to logout users from the back-end too. Though it is not required. Because we can simply logout the users by removing their access token from the front-end. But still sometimes we need to invalidate all the sessions of a user from the back-end too. For that purpose, this API is implemented.

URL: <http://localhost:8080/api/v1/auth/invalidate>

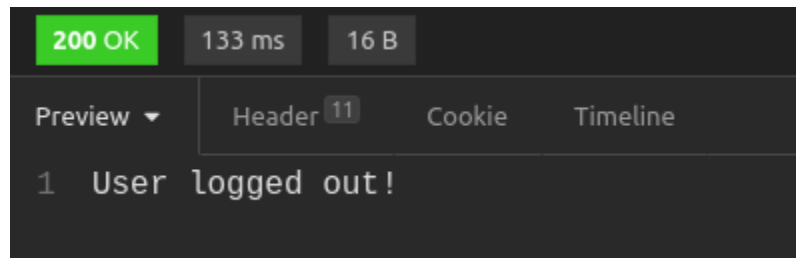
Method: POST

Roles Required: N/A

Request Body:

```
1 {  
2   "userid": 11364  
3 }
```

Response: 200 Ok



The screenshot shows a REST client interface with a dark theme. At the top, there are three status indicators: a green box with '200 OK', a grey box with '133 ms', and another grey box with '16 B'. Below these, there are four tabs: 'Preview' (selected), 'Header', 'Cookie', and 'Timeline'. The 'Preview' tab displays the response body as '1 User logged out!'.

API 04: Get personal profile data

Description: Users (both admin & user role) can use this API to retrieve their personal profile data. This API will only give the profile information of the user that actually made the request. There is no chance that one can access other people's profile information using this API. Thus making our application more secure.

URL: <http://localhost:8080/api/v1/user/profile>

Method: GET

Roles Needed: ROLE_ADMIN, ROLE_USER

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body: N/A

Response: 200 Ok

```
1 {  
2   "userId": "11364",  
3   "username": "rahman.rezaur@bjitgroup.com",  
4   "password": "iamuser",  
5   "role": "ROLE_USER",  
6   "fullName": "Rezaur Rahman",  
7   "imageUrl": "nothing -1",  
8   "gender": "Male",  
9   "contact": null,  
10  "presentAddress": null,  
11  "permanentAddress": null,  
12  "department": null  
13 }
```

API 05: Update personal profile information

Description: Users (both admin & user role) can use this API to update their personal profile data.

URL: <http://localhost:8080/api/v1/user/profile/update>

Method: POST

Roles Needed: ROLE_ADMIN, ROLE_USER

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body:

```
1 {  
2   "userId": "11364",  
3   "username": "rahman.rezaur@bjitgroup.com",  
4   "password": "iamuser",  
5   "role": "ROLE_USER",  
6   "fullName": "Rezaur Rahman",  
7   "imageUrl": null,  
8   "gender": "Male",  
9   "contact": null,  
10  "presentAddress": null,  
11  "permanentAddress": null,  
12  "department": null  
13 }
```

Response: 200 Ok

```
1 {  
2   "userId": "11364",  
3   "username": "rahman.rezaur@bjitgroup.com",  
4   "password": "iamuser",  
5   "role": "ROLE_USER",  
6   "fullName": "Rezaur Rahman",  
7   "imageUrl": null,  
8   "gender": "Male",  
9   "contact": null,  
10  "presentAddress": null,  
11  "permanentAddress": null,  
12  "department": null  
13 }
```

API 06: Get personal image URL

Description: Users (both admin & user role) can use this API to get their profile image URL. This image URL is the location where our image is stored. Using this image URL, we can display the user's image in our front-end application.

URL: <http://localhost:8080/api/v1/user/profile/imageUrl>

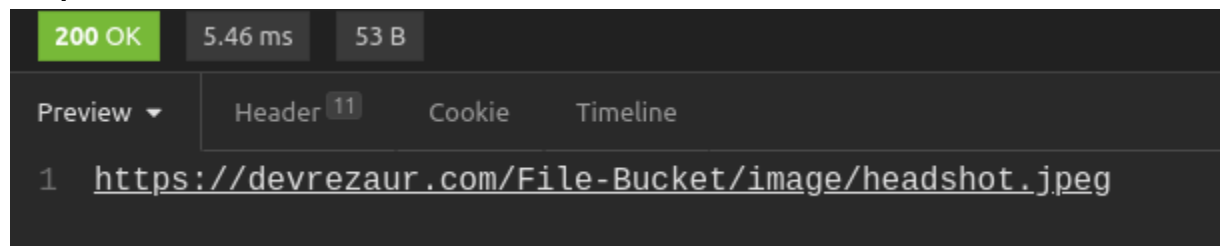
Method: GET

Roles Needed: ROLE_ADMIN, ROLE_USER

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body: N/A

Response: 200 Ok



API 07: Change personal profile image

Description: Users (both admin & user role) can use this API to change / update their profile image URL. This image URL is the location where our image is stored. Using this image URL, we can display the user's image in our front-end application.

URL: <http://localhost:8080/api/v1/user/profile/imageUrl/update>

Method: POST

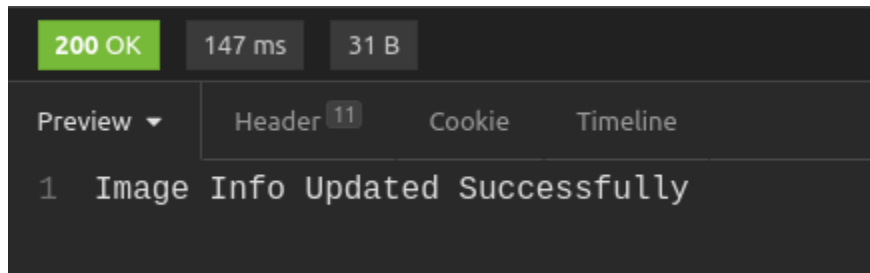
Roles Needed: ROLE_ADMIN, ROLE_USER

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body:

```
1 {  
2   "userId": "11364",  
3   "username": "rahman.rezaur@bjitgroup.com",  
4   "imageUrl": "https://devrezaur.com/File-Bucket/image/headshot.jpeg"  
5 }
```

Response: 200 Ok



API 08: Add new user to the system

Description: Using this API only system admins can add new system users (both ROLE_ADMIN & ROLE_USER) to the application. The request body will contain all the information of a newly created user. Some information fields are mandatory and others are non mandatory fields.

URL: <http://localhost:8080/api/v1/user/add>

Method: POST

Roles Needed: ROLE_ADMIN

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body:

```
JSON ▾  Bearer ▾  Query  Header 1  Docs
1 ▾ {
2   "userId": "10595",
3   "username": "mahin@bjitgroup.com",
4   "password": "mahin",
5   "role": "ROLE_USER",
6   "fullName": "Mr Mahin",
7   "imageUrl": null,
8   "gender": "Male",
9   "contact": null,
10  "presentAddress": null,
11  "permanentAddress": null,
12  "department": "SQA"
13 }
```

Response: 201 Created

```
1 ▾ {
2   "userId": "10595",
3   "username": "mahin@bjitgroup.com",
4   "password": "mahin",
5   "role": "ROLE_USER",
6   "fullName": "Mr Mahin",
7   "imageUrl": null,
8   "gender": "Male",
9   "contact": null,
10  "presentAddress": null,
11  "permanentAddress": null,
12  "department": "SQA"
13 }
```


API 09: Get user data by user id

Description: Using this API only system admins can fetch user data by their user id. There is no request body. Just hit the url with the proper access token with admin credentials.

URL: <http://localhost:8080/api/v1/user/profile/{userId}>

Method: GET

Roles Needed: ROLE_ADMIN

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body: N/A

Response: 200 Ok

```
1▼ {
2   "userId": "10595",
3   "username": "mahin@bjitgroup.com",
4   "password": "mahin",
5   "role": "ROLE_USER",
6   "fullName": "Mr Mahin",
7   "imageUrl": null,
8   "gender": "Male",
9   "contact": null,
10  "presentAddress": null,
11  "permanentAddress": null,
12  "department": "SQA"
13 }
```

API 10: Update existing user data

Description: Using this API only system admins can update user (Both admins & users) data by their user id. The request body will contain updated user data. And the request header will have the proper access token with admin credentials.

URL: <http://localhost:8080/api/v1/user/update>

Method: POST

Roles Needed: ROLE_ADMIN

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body:

```
1 {  
2   "userId": "10595",  
3   "username": "mahin@bjitgroup.com",  
4   "password": "mahin",  
5   "role": "ROLE_USER",  
6   "fullName": "Mr Mahin",  
7   "imageUrl": null,  
8   "gender": "Male",  
9   "contact": "01839100546",  
10  "presentAddress": "Some Place",  
11  "permanentAddress": null,  
12  "department": "Java"  
13 }
```

Response: 200 Ok

```
1 {  
2   "userId": "10595",  
3   "username": "mahin@bjitgroup.com",  
4   "password": "mahin",  
5   "role": "ROLE_USER",  
6   "fullName": "Mr Mahin",  
7   "imageUrl": null,  
8   "gender": "Male",  
9   "contact": "01839100546",  
10  "presentAddress": "Some Place",  
11  "permanentAddress": null,  
12  "department": "Java"  
13 }
```

API 11: List all the users

Description: Using this API both system users and system admins can list all the active users of the system. Only public information will be listed here.

URL: <http://localhost:8080/api/v1/user/all>

Method: GET

Roles Needed: ROLE_ADMIN, ROLE_USER

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body: N/A

Response: 200 Ok

```
1▼ [
2▼  {
3      "userId": "11111",
4      "username": "saleehin@bjitgroup.com",
5      "role": "ROLE_ADMIN",
6      "fullName": "Mohammad Shamsus Saleehin",
7      "imageUrl": "https://devrezaur.com/File-Bucket/image/unnamed.png",
8      "department": "Management"
9  },
10▼ {
11      "userId": "11364",
12      "username": "rahman.rezaur@bjitgroup.com",
13      "role": "ROLE_USER",
14      "fullName": "Rezaur Rahman",
15      "imageUrl": "https://devrezaur.com/File-Bucket/image/headshot.jpeg",
16      "department": "Java"
17  },
18▼ {
19      "userId": "10595",
20      "username": "mahin@bjitgroup.com",
21      "role": "ROLE_USER",
22      "fullName": "Mr Mahin",
23      "imageUrl": null,
24      "department": "Java"
25  }
26  ]
```

API 12: Add new training batch

Description: Using this API only admins can add new training batches to the system. The request body contains the data for the newly created batch.

URL: <http://localhost:8080/api/v1/batch/add>

Method: POST

Roles Needed: ROLE_ADMIN

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body:

```
1 {  
2   "batchName": "Java Fresh Batch 02",  
3   "description": "This is a demo description of Java Batch 01. This  
   batch started at 1 October 2021, and expects to finish it's  
   training activity at 31 December 2021.",  
4   "imageUrl": "https://devrezaur.com/File-Bucket/image/spring.jpg"  
5 }
```

Response: 200 Ok

```
1 {  
2   "batchId": 2,  
3   "batchName": "Java Fresh Batch 02",  
4   "description": "This is a demo description of Java Batch 01. This batch started at 1  
   October 2021, and expects to finish it's training activity at 31 December 2021.",  
5   "imageUrl": "https://devrezaur.com/File-Bucket/image/spring.jpg"  
6 }
```

API 13: List all the training batches

Description: Using this API both users & admins can fetch all the training batches. The response will only contain batch details, not it's internal contents.

URL: <http://localhost:8080/api/v1/batch/all>

Method: GET

Roles Needed: ROLE_ADMIN, ROLE_USER

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body: N/A

Response: 200 Ok

```
1▼ [
2▼  {
3    "batchId": 1,
4    "batchName": "Java Batch 01",
5    "description": "This is a demo description of Java Batch 01. This batch started at 1
6    October 2021, and expects to finish it's training activity at 31 December 2021.",
7    "imageUrl": "https://devrezaur.com/File-Bucket/image/spring.jpg"
8▼  },
9  {
10   "batchId": 2,
11   "batchName": "Java Fresh Batch 02",
12   "description": "This is a demo description of Java Batch 01. This batch started at 1
13   October 2021, and expects to finish it's training activity at 31 December 2021.",
14   "imageUrl": "https://devrezaur.com/File-Bucket/image/spring.jpg"
15  }
16 ]
```

API 14: Update existing training batch's details

Description: Using this API only admins can update an existing batch's details. Like, batch name, description and banner image.

URL: <http://localhost:8080/api/v1/batch/update>

Method: POST

Roles Needed: ROLE_ADMIN

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body:

```
1 {  
2   "batchId": 2,  
3   "batchName": "Java Fresh Batch 02",  
4   "description": "This is upadated demo description of Java Batch 01.  
   This batch started at 1 October 2021, and expects to finish it's  
   training activity at 31 December 2021.",  
5   "imageUrl": "https://devrezaur.com/File-Bucket/image/spring.jpg"  
6 }
```

Response: 200 Ok

```
1 {  
2   "batchId": 2,  
3   "batchName": "Java Fresh Batch 02",  
4   "description": "This is upadated demo description of Java Batch 01. This batch  
   started at 1 October 2021, and expects to finish it's training activity at 31  
   December 2021.",  
5   "imageUrl": "https://devrezaur.com/File-Bucket/image/spring.jpg"  
6 }
```

API 15: Get a particular training batch details by batch id

Description: Using this API both users and admins can fetch public details of an existing training batch.

URL: <http://localhost:8080/api/v1/batch/{batchId}>

Method: GET

Roles Needed: ROLE_ADMIN, ROLE_USER

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body: N/A

Response: 200 Ok

```
1 {  
2   "batchId": 2,  
3   "batchName": "Java Fresh Batch 02",  
4   "description": "This is updated demo description of Java Batch 01. This batch  
   started at 1 October 2021, and expects to finish it's training activity at 31  
   December 2021.",  
5   "imageUrl": "https://devrezaur.com/File-Bucket/image/spring.jpg"  
6 }
```

API 16: Get list of enrolled user's public details of a particular batch

Description: Using this API both users and admins can fetch the public details of all the enrolled users of an existing training batch.

URL: <http://localhost:8080/api/v1/batch/{batchId}/enrolledUsers>

Method: GET

Roles Needed: ROLE_ADMIN, ROLE_USER

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body: N/A

Response: 200 Ok

```
1▼ [
2▼  {
3    "userId": "11364",
4    "username": "rahman.rezaur@bjitgroup.com",
5    "role": "ROLE_USER",
6    "fullName": "Rezaur Rahman",
7    "imageUrl": "https://devrezaur.com/File-Bucket/image/headshot.jpeg",
8    "department": "Java"
9  },
10▼ {
11    "userId": "10595",
12    "username": "mahin@bjitgroup.com",
13    "role": "ROLE_USER",
14    "fullName": "Mr Mahin",
15    "imageUrl": null,
16    "department": "Java"
17  }
18 ]
```


API 17: Enroll user to a batch

Description: Using this API only admins can enroll users to a particular batch, only if the user exists and not yet enrolled. Otherwise it will throw exceptions. The request body will contain the batch id in which batch we want to enroll. And a user id of the user that we want to enroll.

URL: <http://localhost:8080/api/v1/batch/enroll>

Method: POST

Roles Needed: ROLE_ADMIN

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body:

```
1 {  
2   "batchId": 13,  
3   "userId": 10595  
4 }
```

Response: 201 Created

201 Created	10.7 ms	26 B
Preview ▼	Header 11	Cookie
1 Successfully Enrolled User		

API 18: Un-enroll a user from a particular batch

Description: Using this API only admins can un-enroll users to a particular batch. If the user is already un-enrolled, then it will throw exceptions. The request body will contain the batch id from which batch we want to un-enroll. And a user id of the user that we want to un-enroll.

URL: <http://localhost:8080/api/v1/batch/un-enroll>

Method: POST

Roles Needed: ROLE_ADMIN

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body:

```
1 {  
2   "batchId": 1,  
3   "userId": 10595  
4 }
```

Response: 201 Created

The screenshot shows an API client interface with a dark theme. At the top, a green status bar displays '201 Created', '13.9 ms', and '29 B'. Below this, a tabbed interface has 'Preview' selected, showing a single line of text: '1 Successfully Un-Enrolled User'. Other tabs visible are 'Header' (with a count of 11), 'Cookie', and 'Timeline'.

API 19: Get the list of all the enrolled batches by the user id

Description: Using this API both users & admins can fetch all the batches that they are enrolled into. The request body will contain the user id of the user.

URL: <http://localhost:8080/api/v1/batch/batches>

Method: GET

Roles Needed: ROLE_ADMIN, ROLE_USER

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body:

```
1 {  
2   "userid": 11364  
3 }
```

Response: 200 Ok

```
1 [  
2   {  
3     "batchId": 1,  
4     "batchName": "Java Batch 01",  
5     "description": "This is a demo description of Java Batch 01. This batch started  
   at 1 October 2021, and expects to finish it's training activity at 31 December  
   2021.",  
6     "imageUrl": "https://devrezaur.com/File-Bucket/image/spring.jpg"  
7   }  
8 ]
```

API 20: Make a post to the batch dashboard

Description: Using this API both users & admins can make posts to the batch dashboard. The request body will contain the batch's id, the user's id of who is making the post, post description and additional resource name & link (if any resource is attached to the post).

URL: <http://localhost:8080/api/v1/post>

Method: POST

Roles Needed: ROLE_ADMIN, ROLE_USER

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body:

```
JSON ▾  Bearer ▾  Query  Header 1  Docs
1 {
2   "batchId": 13,
3   "userId": 11111,
4   "description": "This is a demo post. This was used for testing
5   purpose.",
6   "resourceName": "Research Methodology",
7   "resourcesLink": "https://devrezaur.com/File-Bucket/file/Research Methodology Lecture - 8.pptx"
8 }
```

Response: 201 Created

```
201 Created  9.28 ms  25 B
Preview ▾  Header 11  Cookie  Timeline
1 Successfully Post Created
```

API 21: Fetch all posts of a particular batch

Description: Using this API both users & admins can fetch all the posts and contents of a particular batch. There is no request body.

URL: <http://localhost:8080/api/v1/post/{batchId}>

Method: GET

Roles Needed: ROLE_ADMIN, ROLE_USER

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body: N/A

Response: 200 Ok

```
20▼ {
21   "postId": 3,
22   "fullName": "Mohammad Shamsus Saleehin",
23   "imageUrl": "https://devrezaur.com/File-Bucket/image/unnamed.png",
24   "desc": "This is first demo post by the admin. Please go through the attached training
material and submit the assignment via the link https://www.google.com Thank you.",
25   "resourceName": "Research Methodology Lecture - 8.pptx",
26   "resourcesLink": "https://devrezaur.com/File-Bucket/file/Research Methodology Lecture -
8.pptx",
27   "createDate": "2021-12-28T16:46:15.059+00:00"
28 },
29▼ {
30   "postId": 2,
31   "fullName": "Mohammad Shamsus Saleehin",
32   "imageUrl": "https://devrezaur.com/File-Bucket/image/unnamed.png",
33   "desc": "This is first demo post by the admin. Please go through the attached training
material and submit the assignment via the link https://www.google.com Thank you.",
34   "resourceName": "[BASIC] Software-design-v3.pptx",
35   "resourcesLink": "https://devrezaur.com/File-Bucket/file/[BASIC] Software-design-
v3.pptx",
36   "createDate": "2021-12-28T16:44:19.346+00:00"
37 },
38▼ {
39   "postId": 1,
40   "fullName": "Mohammad Shamsus Saleehin",
41   "imageUrl": "https://devrezaur.com/File-Bucket/image/unnamed.png",
42   "desc": "This is first demo post by the admin. Please go through the attached training
material and submit the assignment via the link www.google.com. Thank you.",
43   "resourceName": "",
44   "resourcesLink": "https://devrezaur.com/File-Bucket/file/",
45   "createDate": "2021-12-28T16:38:16.232+00:00"
46 }
47 ]
```

API 22: Post new message in the discussion section

Description: Using this API both users & admins can post messages to the batch's discussion forum. The request body will contain the batch's id, the user's id of who is sending the message and the message text.

URL: <http://localhost:8080/api/v1/message>

Method: POST

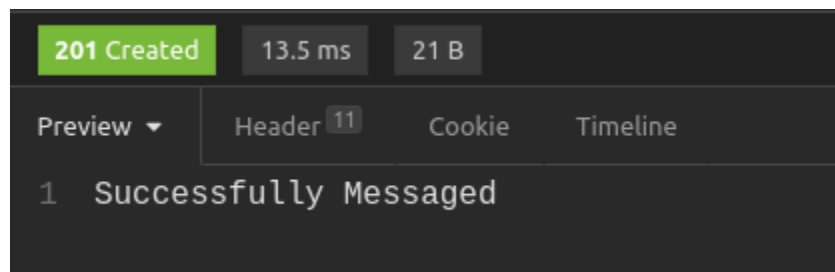
Roles Needed: ROLE_ADMIN, ROLE_USER

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body:

```
1 {  
2   "batchId": 1,  
3   "userId": 11111,  
4   "fullName": "Mohammad Shamsus Saleehin",  
5   "message": "This is a demo message. This was used for testing  
6   purpose."  
7 }
```

Response: 201 Created



The screenshot shows a REST client interface with a dark theme. At the top, a green status bar displays '201 Created', '13.5 ms', and '21 B'. Below this is a tabbed interface with 'Preview' selected, showing 'Header' (11), 'Cookie', and 'Timeline' tabs. The main content area displays '1 Successfully Messaged'.

API 23: Fetch messages of a particular batch's discussion forum

Description: Using this API both users & admins can fetch messages posted to a batch's discussion forum. There will be no request body, and the response will contain a list of messages.

URL: <http://localhost:8080/api/v1/message/{batchId}>

Method: GET

Roles Needed: ROLE_ADMIN, ROLE_USER

Access Token Required: Bearer token (JWT access token) should be added to the request header

Request Body: N/A

Response: 200 Ok

```
1▼ [
2▼  {
3    "messageId": 2,
4    "batchId": 13,
5    "fullName": "Mohammad Shamsus Saleehin",
6    "message": "This is a demo message. This was used for testing purpose.",
7    "createDate": "2021-12-28T09:49:34.599+00:00"
8  },
9▼  {
10   "messageId": 1,
11   "batchId": 13,
12   "fullName": "Rezaur Rahman",
13   "message": "This is a demo message. This was used for testing purpose.",
14   "createDate": "2021-12-28T09:48:13.958+00:00"
15 }
16 ]
```

FINISH