# End to End Testing using Postman

# SEM 34c

**User story: User authenticates through Authentication microservice**

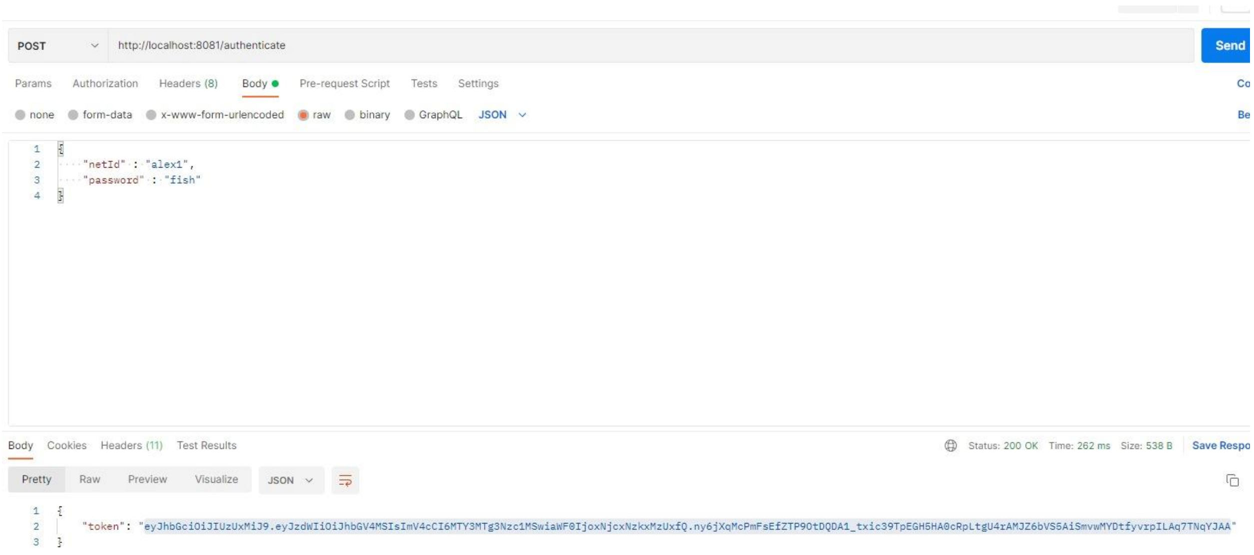
Path: http://localhost:8081/authenticate

Input:

```
{
   "netId" : "alex1",
   "password" : "fish"
}
```

Expected output: User is authenticated and receives a token

Actual output:



```
POST http://localhost:8081/authenticate
Body: raw JSON
{
    "netId" : "alex1",
    "password" : "fish"
}
```

```
Status: 200 OK  Time: 262 ms  Size: 538 B
{
    "token": "eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJhbGV4MSIsImV4cCI6MTY3MTg3Nzc1MSwiaWF0IjoxNjcxNzkxMzUxfQ.ny6jXqMcPmFsEfZTP9OtDQDA1_txic39TpEGH5HA0cRpLtgU4rAMJZ6bVS5AiSmvwMYDtfyvrpILAq7TNqYJAA"
}
```

**User story: Set details for a new account**

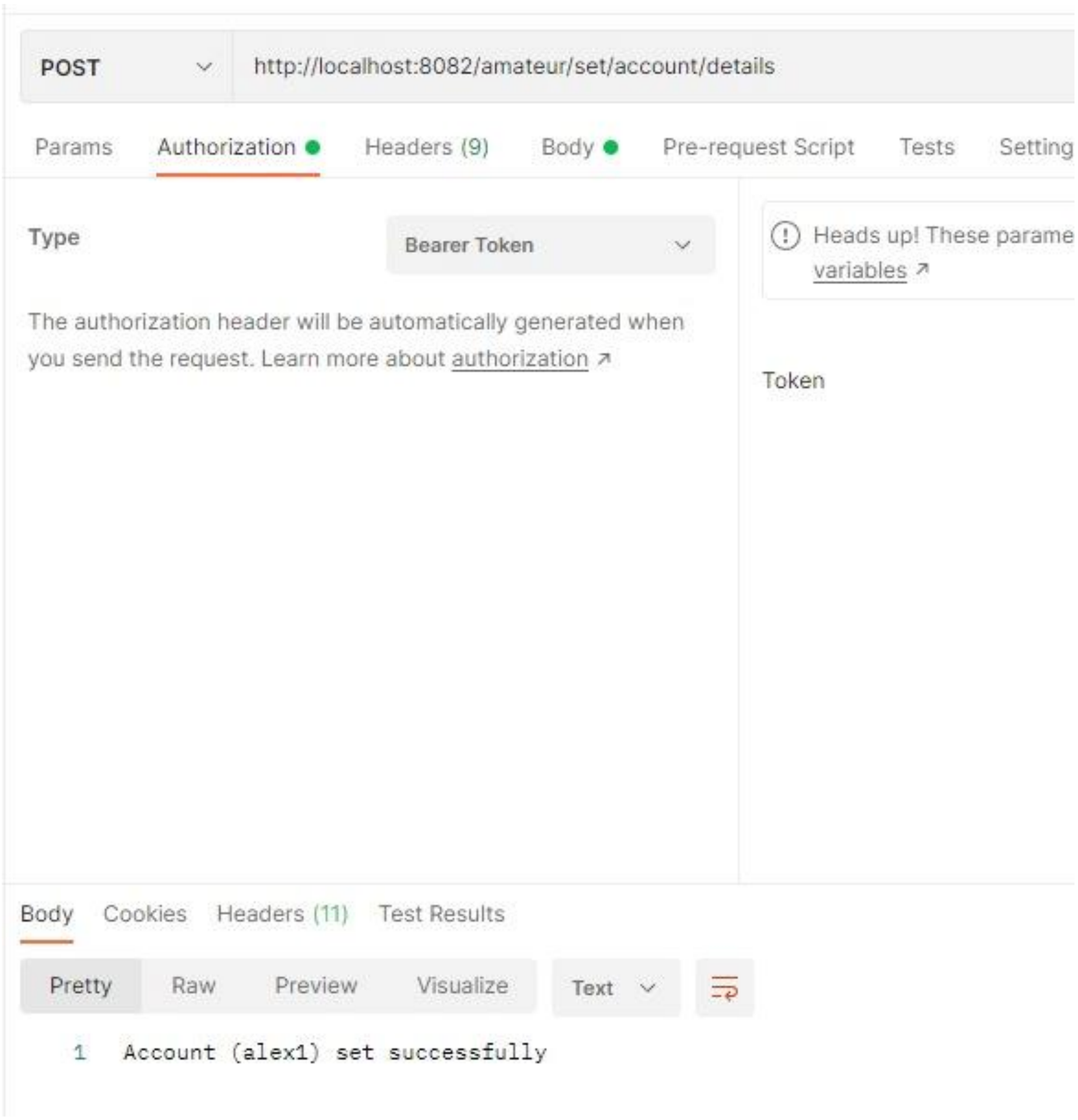
Path: http://localhost:8082/amateur/set/account/details

Input:

```
{
   "netId" : "alex1",
   "password" : "fish",
   "gender" : "MALE",
   "positions": ["COX", "COACH"],
   "availabilities" :
   [
      {
            "first" : "2022-12-12T12:00",
            "second" : "2022-12-12T13:00"
        },
        {
            "first" : "2022-12-12T13:00",
            "second" : "2022-12-12T14:00"
        },
        {
            "first" : "2022-12-12T16:00",
            "second" : "2022-12-12T17:00"
        }
    ],
    "certificates" : ["C4", "8+"],
    "organization" : "teamAlpha"
}
```

Expected output: 200 ok response and the details are added successfully to the account

Actual output:

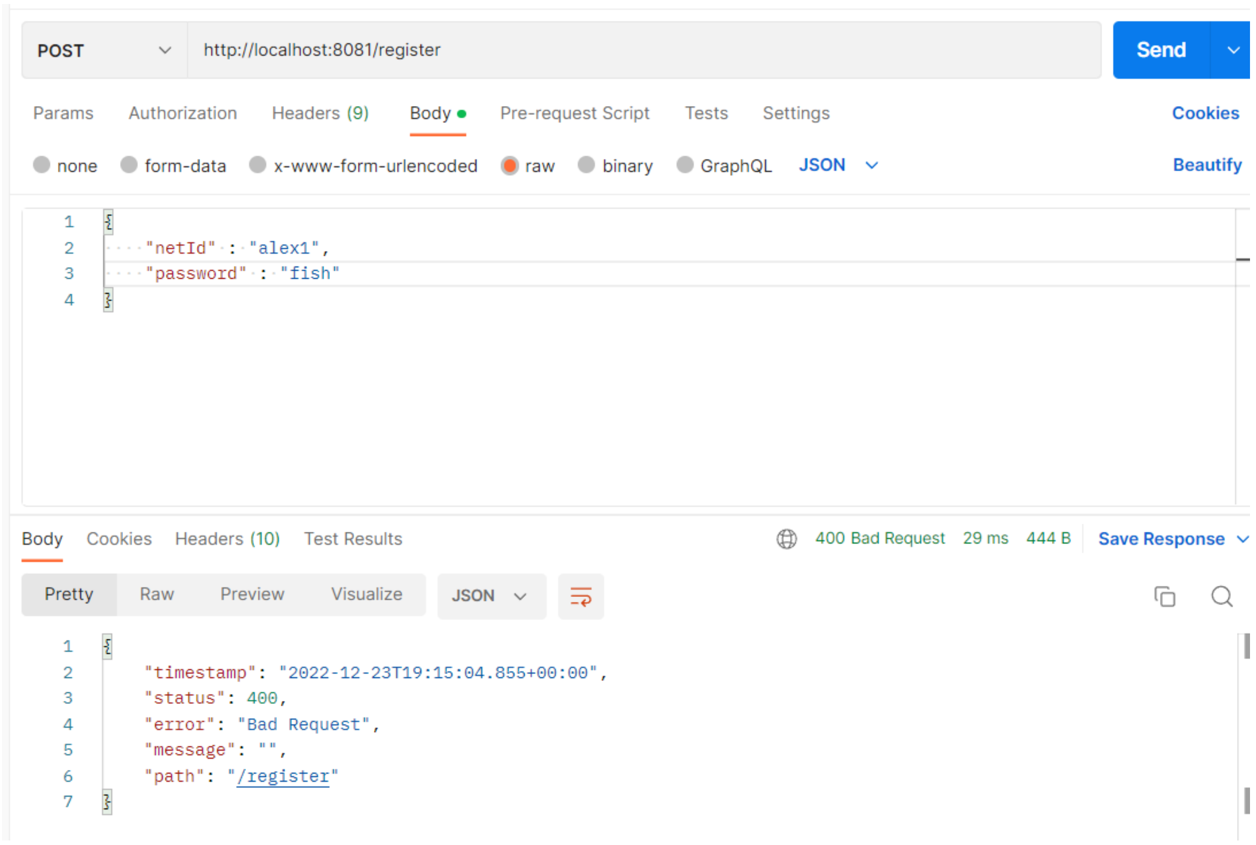


http://localhost:8082/amateur/set/account/details

The authorization header will be automatically generated when you send the request. Learn more about authorization ↗

```
Account (alex1) set successfully
```

**User story: Already registered user cannot register again**
Path: http://localhost:8081/register
Input:
```
{
   "netId" : "alex1",
   "password" : "fish"
}
```
Expected output: Bad request response or a similar error
Actual output:


POST http://localhost:8081/register

```
{
    "netId" : "alex1",
    "password" : "fish"
}
```

400 Bad Request 29 ms 444 B

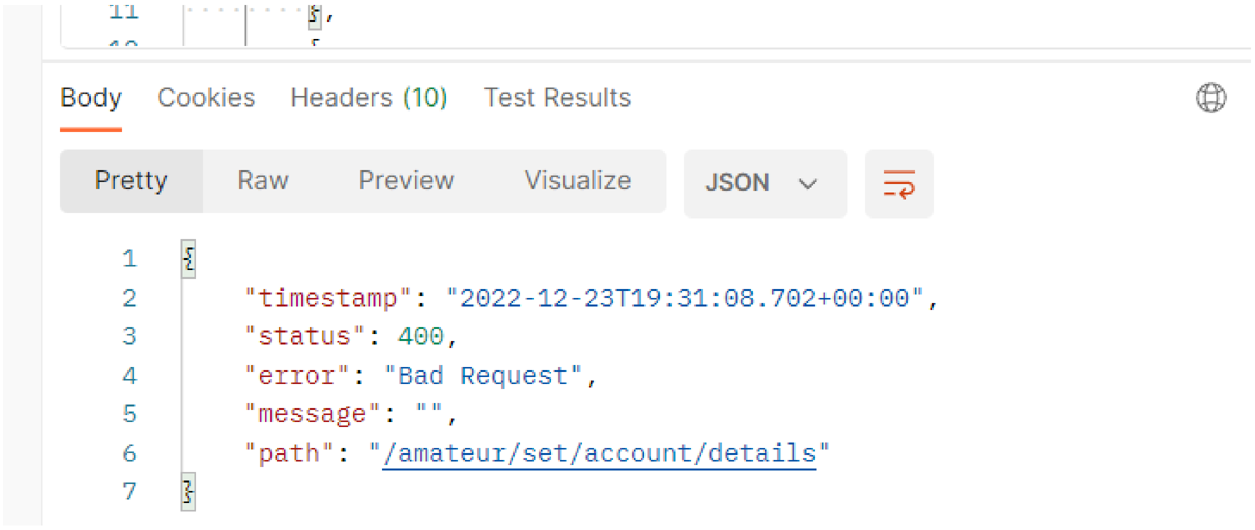
```
{
    "timestamp": "2022-12-23T19:15:04.855+00:00",
    "status": 400,
    "error": "Bad Request",
    "message": "",
    "path": "/register"
}
```

**User story: Try to set incorrect details to a user, such as position (for example, CO instead of COX)**

Path: http://localhost:8082/amateur/set/account/details

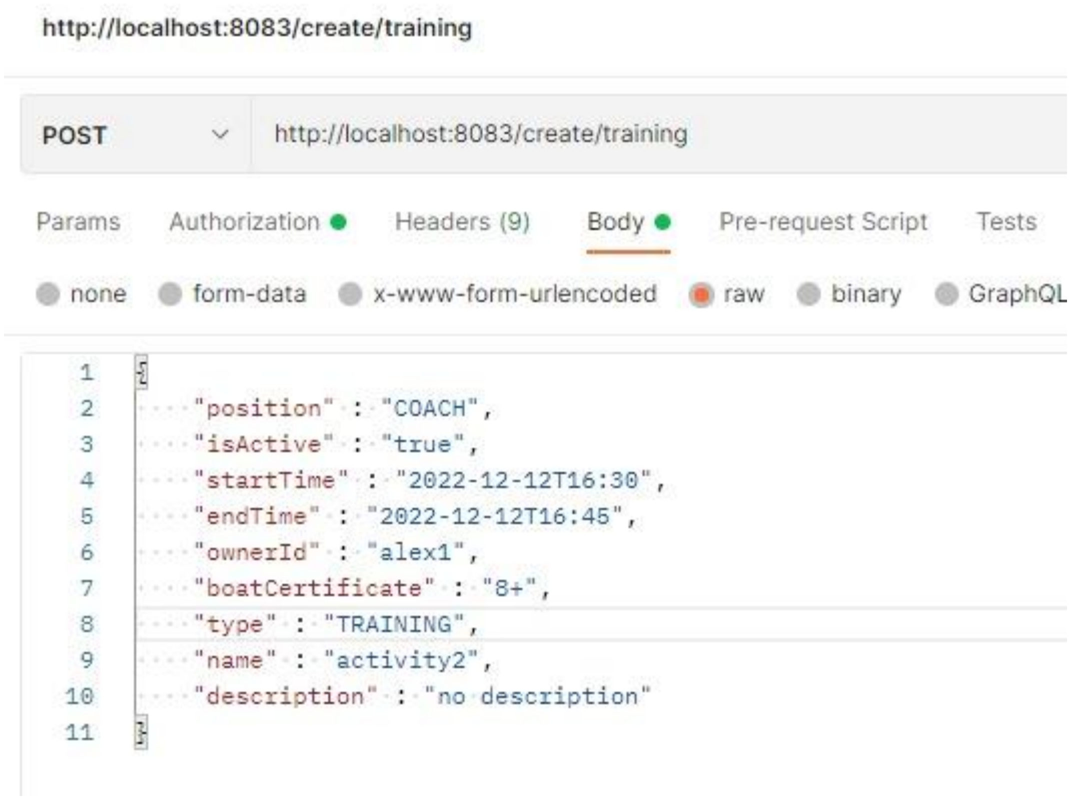
Input:

```
{
   "netId" : "ale",
   "password" : "fish",
   "gender" : "MALE",
   "positions": ["CO", "COACH"],
   "availabilities" :
   [
      {
         "first" : "2022-12-12T12:00",
         "second" : "2022-12-12T13:00"
      },
      {
         "first" : "2022-12-12T13:00",
         "second" : "2022-12-12T14:00"
      },
      {
            "first" : "2022-12-12T16:00",
            "second" : "2022-12-12T17:00"
        }
    ],
    "certificates" : ["C4", "8+"],
    "organization" : "teamAlpha"
}
```
Expected output: Bad request response or a similar error
Actual output:



```
{
    "timestamp": "2022-12-23T19:31:08.702+00:00",
    "status": 400,
    "error": "Bad Request",
    "message": "",
    "path": "/amateur/set/account/details"
}
```

**User story: Creation of training**
Path: http://localhost:8083/create/training
Input:



http://localhost:8083/create/training

POST http://localhost:8083/create/training

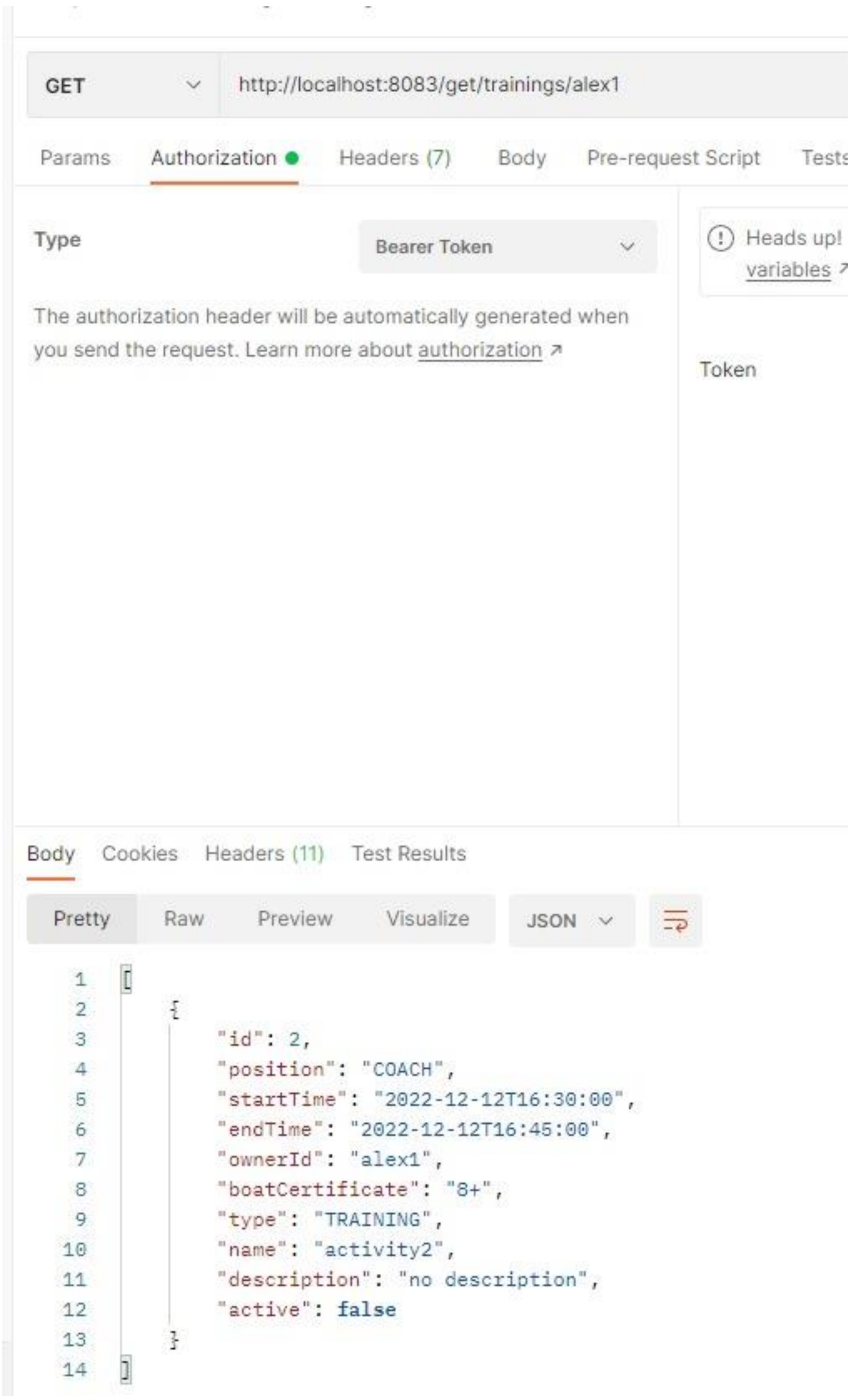
Params | Authorization ● | Headers (9) | Body ● | Pre-request Script | Tests

none | form-data | x-www-form-urlencoded | [✓ raw] | binary | GraphQL

```
{
    "position" : "COACH",
    "isActive" : "true",
    "startTime" : "2022-12-12T16:30",
    "endTime" : "2022-12-12T16:45",
    "ownerId" : "alex1",
    "boatCertificate" : "8+",
    "type" : "TRAINING",
    "name" : "activity2",
    "description" : "no description"
}
```

Expected output: Training has been created
Actual output: 200 ok response, training is now in database and can be looked for

**User Story: Get a training created by a specific user**
Path: http://localhost:8083/get/trainings/alex1
Input: -
Expected output: Trainings created by that user are shown
Actual output:


GET http://localhost:8083/get/trainings/alex1

Type: Bearer Token

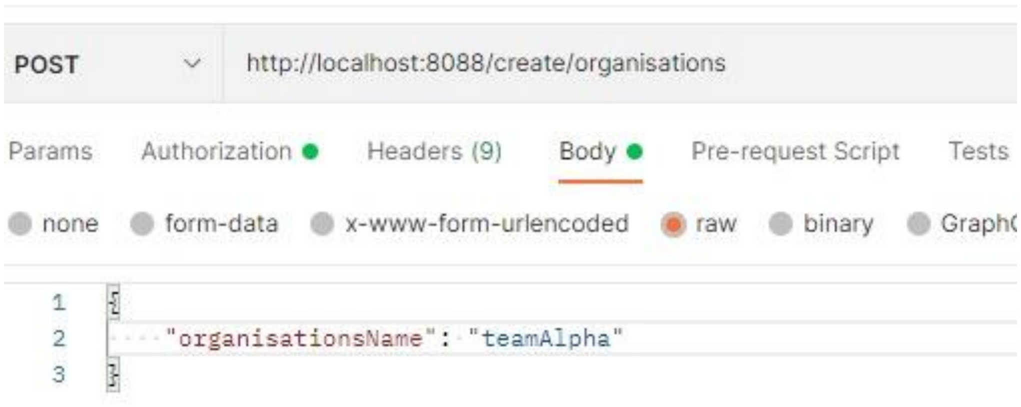
The authorization header will be automatically generated when you send the request. Learn more about authorization ↗

Heads up! variables

Token

```
[
    {
        "id": 2,
        "position": "COACH",
        "startTime": "2022-12-12T16:30:00",
        "endTime": "2022-12-12T16:45:00",
        "ownerId": "alex1",
        "boatCertificate": "8+",
        "type": "TRAINING",
        "name": "activity2",
        "description": "no description",
        "active": false
    }
]
```

**User story: Create an organization**
Path: http://localhost:8088/create/organisations
Input:

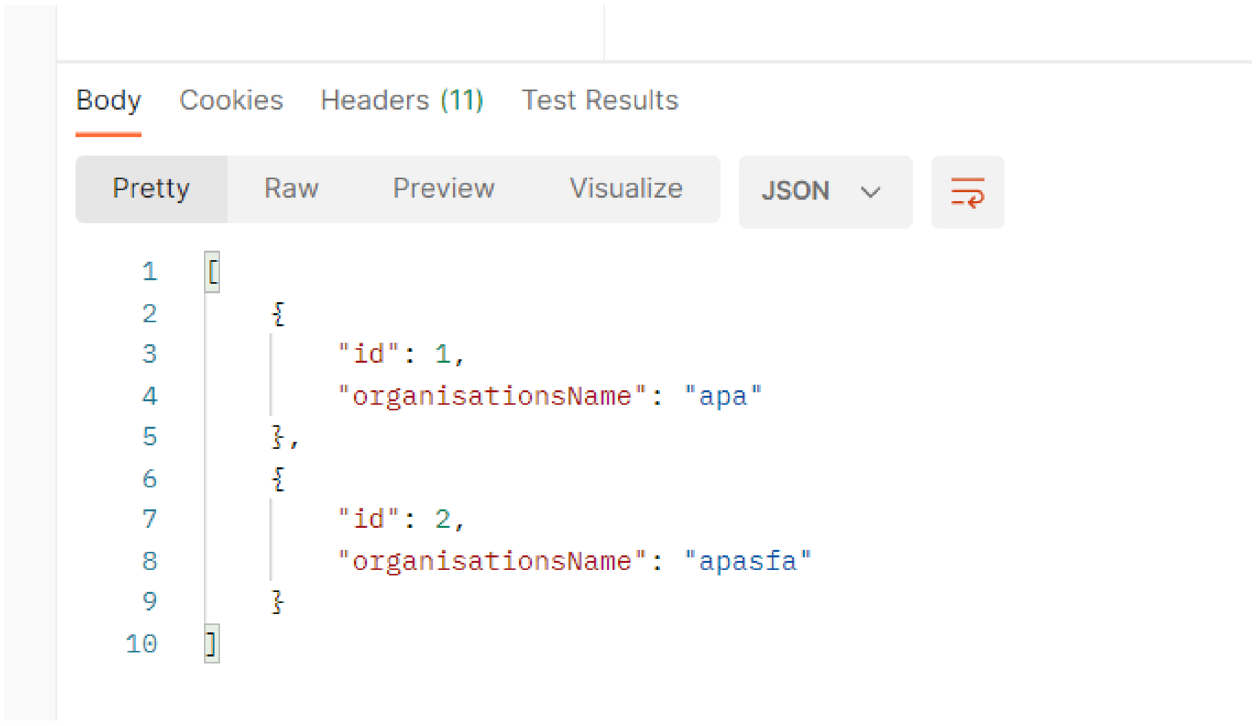


POST http://localhost:8088/create/organisations

```
{
    "organisationsName": "teamAlpha"
}
```

Expected output: Organisation is created
Actual output: Organisation is created successfully with response 200 ok

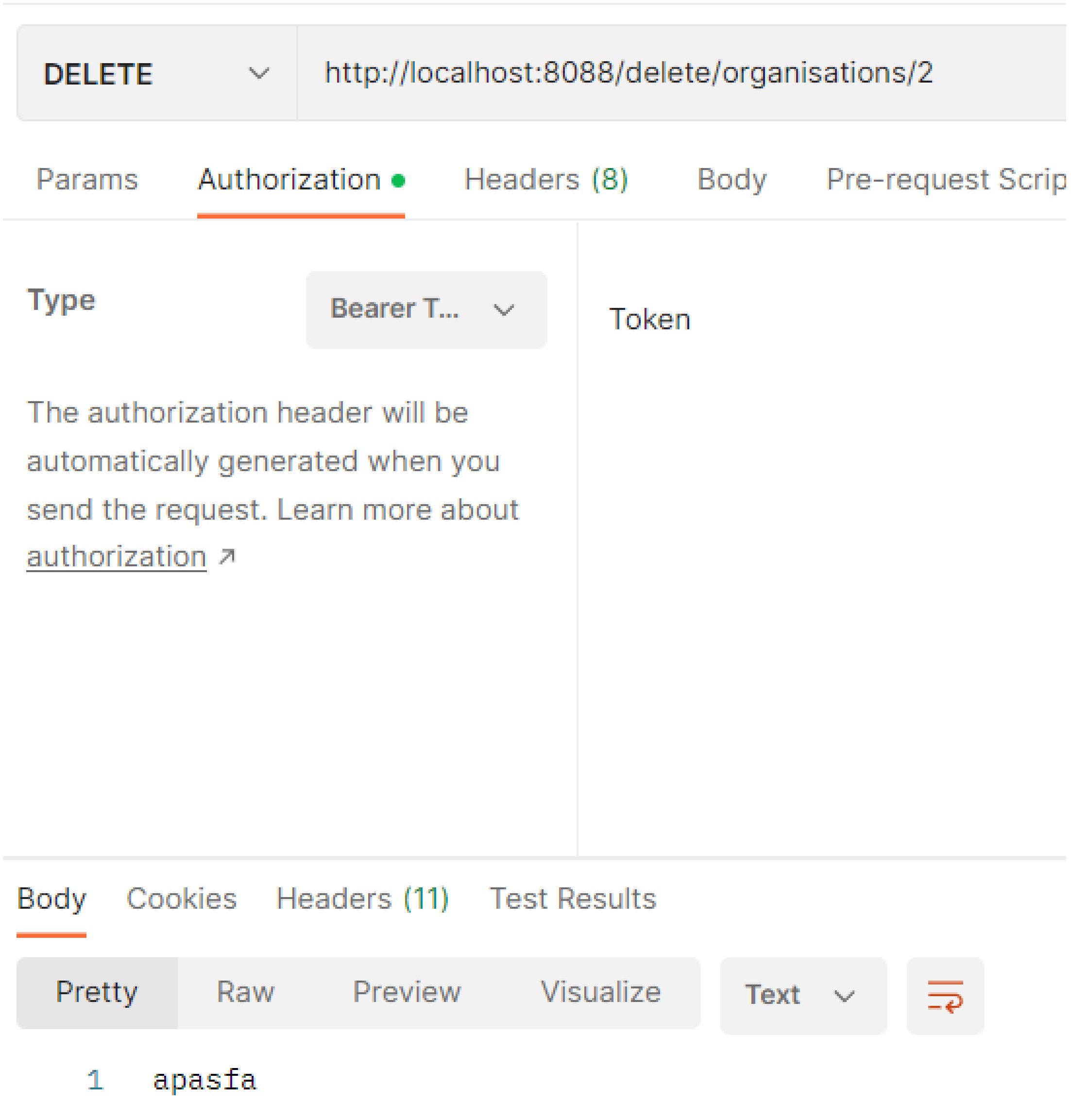
**User story: All organisations can be received**
Path: http://localhost:8088/get/organisations
Input: -
Expected output: All organisations are displayed
Actual output:



```
[
    {
        "id": 1,
        "organisationsName": "apa"
    },
    {
        "id": 2,
        "organisationsName": "apasfa"
    }
]
```

**User story: Delete a specific organization with admin privilege (user must have admin netId)**

Path: http://localhost:8088/delete/organisations/2
Input: -
Expected output: Organisation is not in the database anymore
Actual result: Name of deleted organisation is displayed, organisation is successfully deleted



DELETE http://localhost:8088/delete/organisations/2

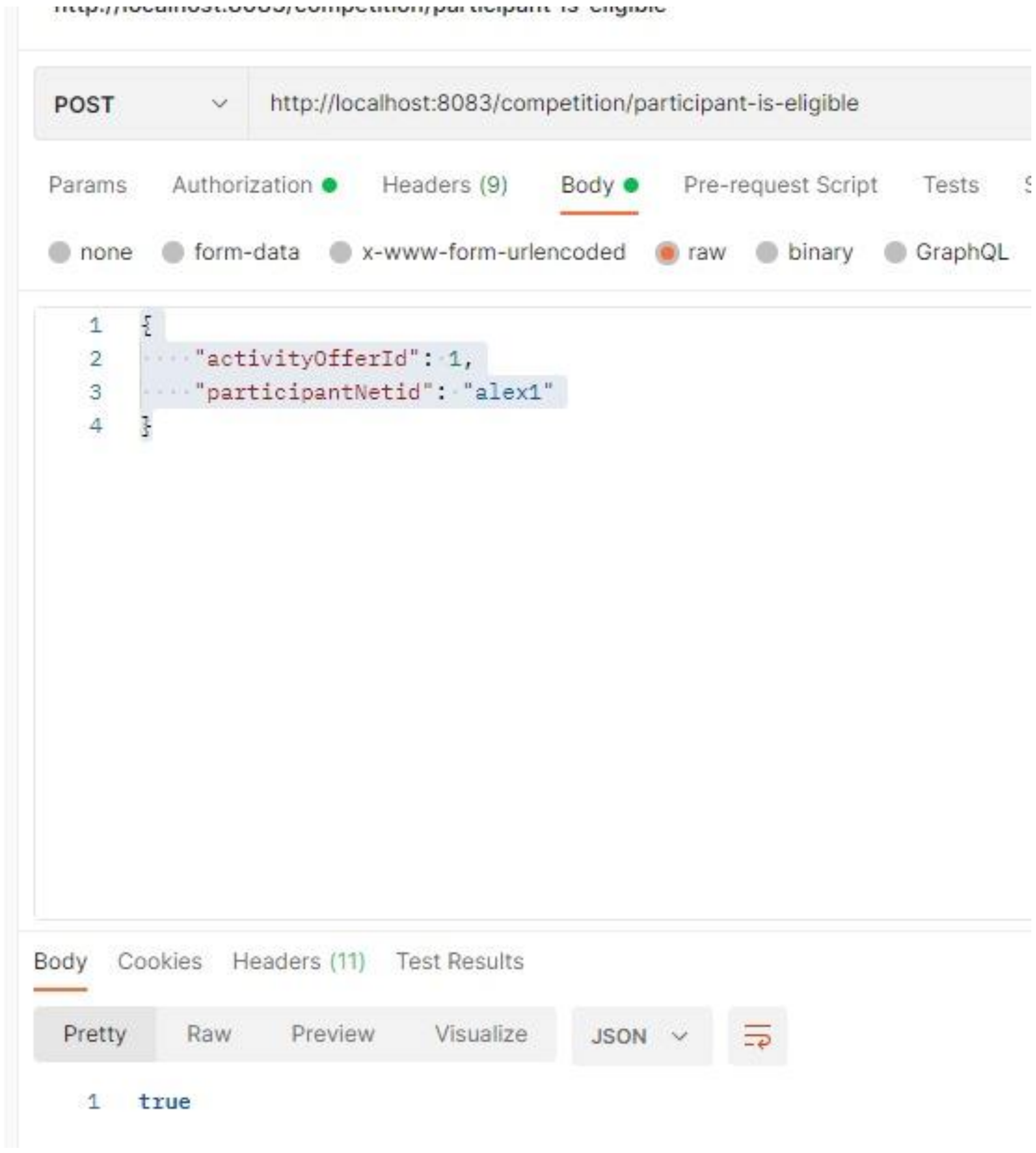
Type: Bearer T...

Token

The authorization header will be automatically generated when you send the request. Learn more about authorization ↗

```
apasfa
```

**User story: Check whether a participant can join a specific activity (competition in this case)**
Path: http://localhost:8083/competition/participant-is-eligible
Expected output: Should be true by using the aforementioned examples of user and training activity
Input and Actual output:



POST http://localhost:8083/competition/participant-is-eligible

Params | Authorization ● | Headers (9) | Body ● | Pre-request Script | Tests

none | form-data | x-www-form-urlencoded | [✓ raw] | binary | GraphQL

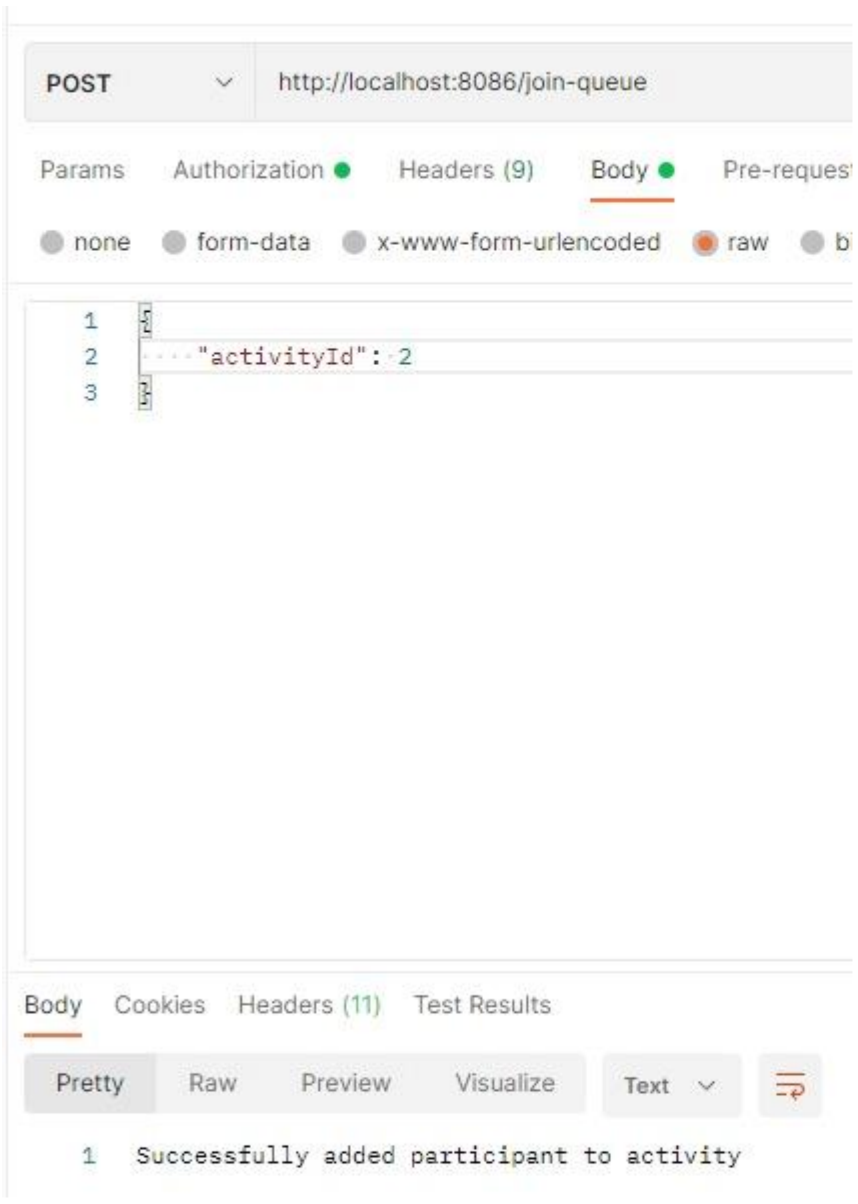
```
{
    "activityOfferId": 1,
    "participantNetid": "alex1"
}
```

Body | Cookies | Headers (11) | Test Results

Pretty | Raw | Preview | Visualize | JSON

```
true
```

**User story: User can join a specific activity and be enlisted in its queue**
Path: http://localhost:8086/join-queue
Expected output: Participant is added to that specific activity queue
Input and Actual output:


POST http://localhost:8086/join-queue

Params | Authorization | Headers (9) | Body | Pre-reques

none | form-data | x-www-form-urlencoded | raw | b

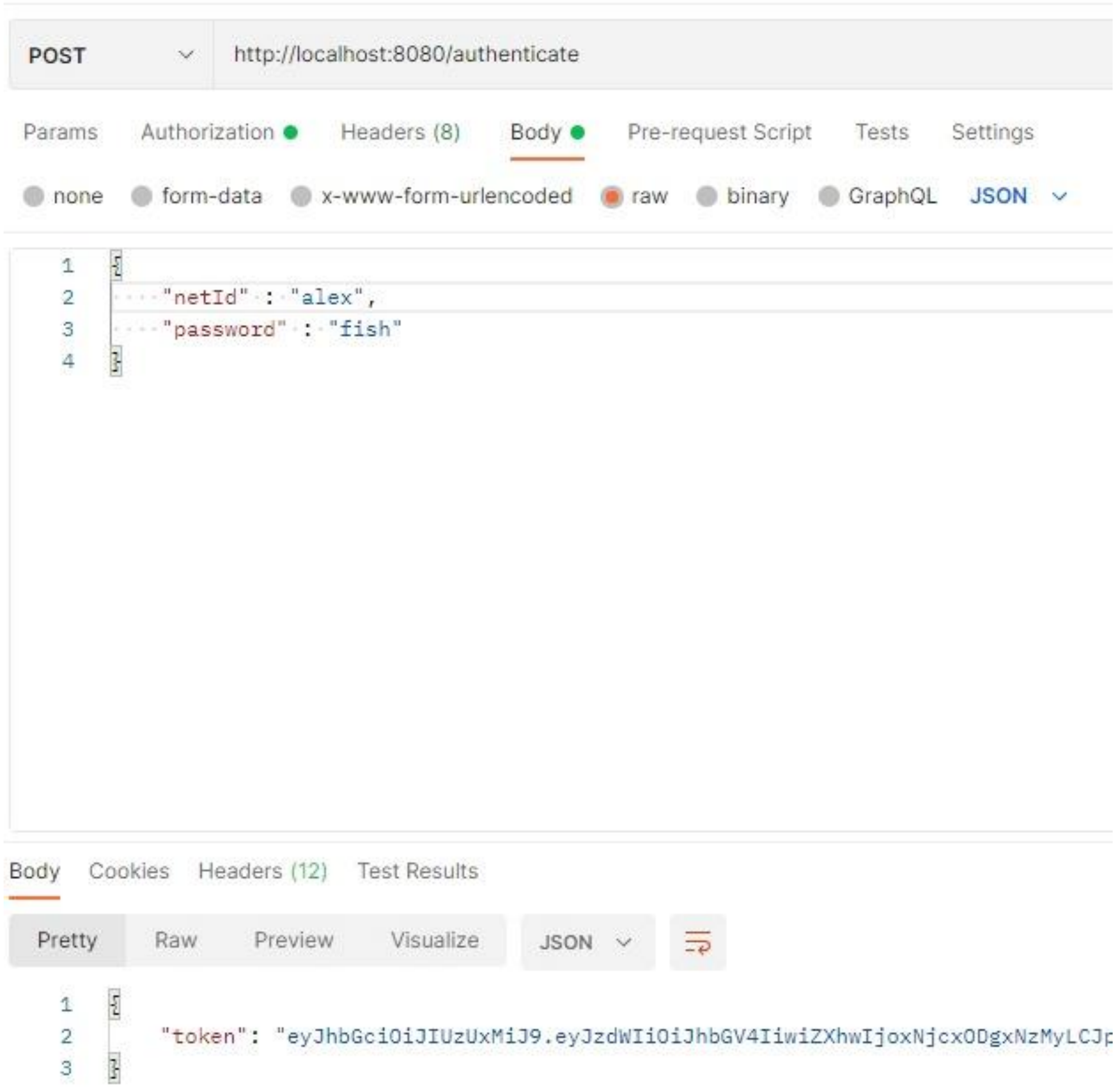
```
{
    "activityId": 2
}
```

Body | Cookies | Headers (11) | Test Results

Pretty | Raw | Preview | Visualize | Text

```
Successfully added participant to activity
```

**User story: Authenticate user through the Gateway microservice**
Path: http://localhost:8080/authenticate
Input:
```
{
   "netId" : "alex1",
   "password" : "fish"
}
```
Expected output: User is authenticated successfully and receives a token
Acutal output:



POST http://localhost:8080/authenticate

none | form-data | x-www-form-urlencoded | [✓ raw] | binary | GraphQL | JSON

```
{
    "netId" : "alex",
    "password" : "fish"
}
```

Body — Pretty — JSON

```
{
    "token": "eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJhbGV4IiwiZXhwIjoxNjcxODgxNzMyLCJp
}
```