

Boundary Testing

User-Microservice

Endpoint 8081/Register

- Both the password and the netId need to have between 6 and 20 characters. So we try to register with 5 and 21 characters for each and expect a bad request answer. Empty netIds and passwords should have the same behavior. We also test for 6 and 20 characters on both and expect it to work.

The image displays two screenshots of a REST client interface, likely Postman, showing a POST request to the endpoint `http://localhost:8081/register`.

Left Screenshot: The request body is set to raw (JSON) and contains the following JSON:

```
{  "netId": "TheseAreTwentyOneChar",  "password": "123456"}
```

The response body is shown in JSON format:

```
{  "timestamp": "2022-12-23T18:17:05.074+00:00",  "status": 400,  "error": "Bad Request",  "message": "",  "path": "/register"}
```

Right Screenshot: The request body is set to raw (JSON) and contains the following JSON:

```
{  "netId": "TheseAreTwentyCharss",  "password": "TheseAreTwentyCharss"}
```

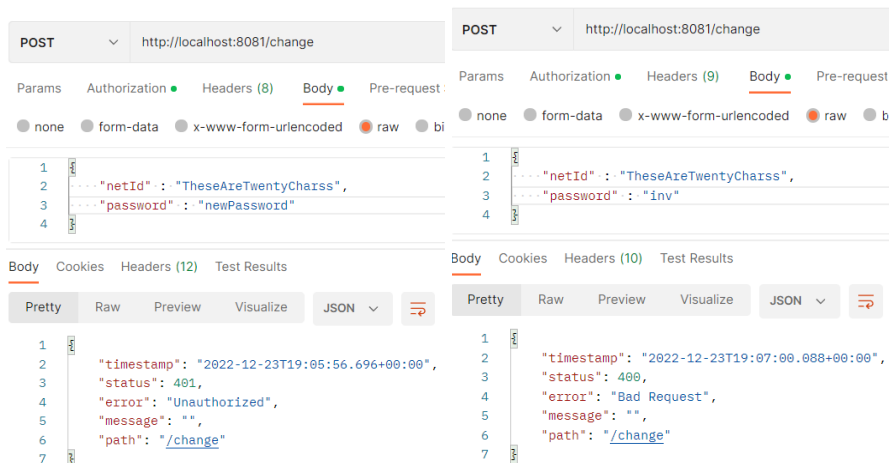
The response body is shown in JSON format:

```
{  "timestamp": "2022-12-23T18:17:05.074+00:00",  "status": 400,  "error": "Bad Request",  "message": "",  "path": "/register"}
```

- NOTE: the body being empty means the user was successfully registered.
- Presented correct behavior.

Endpoint 8081/authenticate

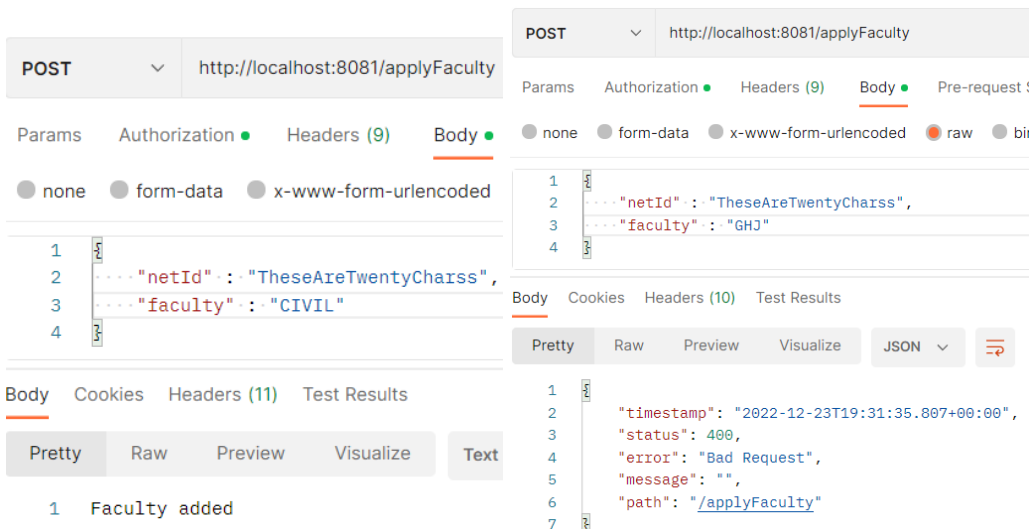
- The user needs to be registered to be authenticated. The password needs to be correct. We first check for an unregistered user, then for a registered user but with the wrong password. Furthermore, we check the behavior for an empty user and password. At last, we authenticate a registered user correctly.



- On the second screenshot, we tried changing the password from another registered account and it worked. This is a bug that should be fixed in the future, since a user should not change the password of another user.
- Other than that, the behavior is as expected.

Endpoint 8081/applyFaculty

- We tested with faculties that are in the database and faculties that are not. We also tried adding multiple faculties to a user and adding multiple times the same faculty.

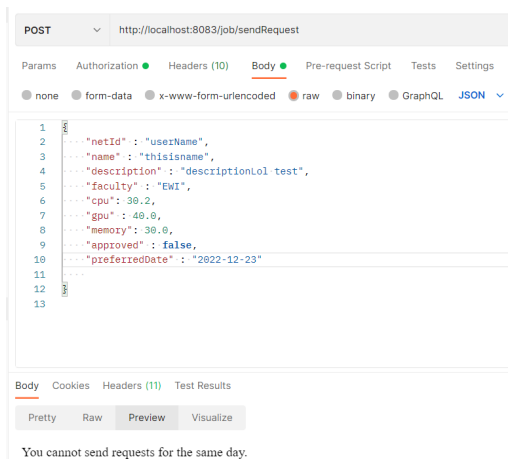


- When trying to assign a non-existing faculty, we get a bad request (correct behavior).
- When we try to add a faculty where the user is already a member, it says the faculty was added. It could say that the user is already a member in that faculty, instead.
- The other two cases work correctly as well.
- A possible improvement for the future is that only faculty managers and Sysadmins can do it.

Request-Microservice

Endpoint 8083/job/sendRequest

This endpoint is for users that want to make a request to use part of the cluster's computing power. You also need to specify a date when you want the job to be fulfilled. However when the preferred date is the same

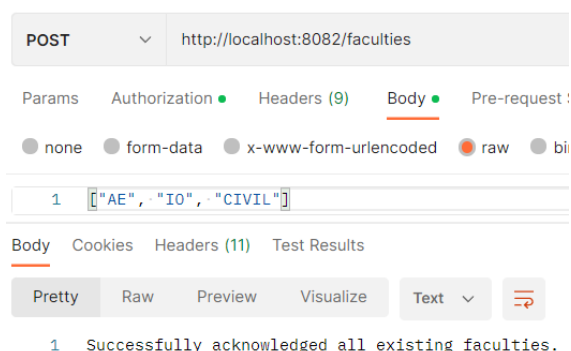


We tested that when a request's preferred completion date is the same day as today, the request will not be saved. In the image you can see that the response is as expected.

Cluster-Microservice

Endpoint 8083/faculties

- The cluster receives the list of faculties from the authentication microservice, and trusts it to send only existing faculties. So any list of faculties received will be added to the cluster as existing valid faculties.



- NetId's of faculties start with the word 'fac', so we should adjust faculties to conform to that in some way. One possibility is could be extracting the faculty name from the NetId.

- If the list is empty, it also says the existing faculties were acknowledged, although none are actually added to the cluster. That could be changed in the future, to say that no faculties were received.

Endpoint 8083/nodes/add(/facultyId)

- Can be called with or without the faultyId. Try adding a node with less cpu than both memory and gpu resources. Then less than just memory, less than just gpu, and finally with equal amount for the three. Only the last one should be added. After that, we try adding a node with an url that is equal to a node that is already in the cluster (should also fail to be added). Furthermore, we try adding a node with negative resources. Finally, we try to add a node with an empty url.

POST ▼ http://localhost:8082/nodes/add Send

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

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL JSON ▼ Beauti

```

1  {
2    "cpuResources": 14.9,
3    "gpuResources": 15.0,
4    "memoryResources": 15.0,
5    "name": "node",
6    "url": "/nodeUrl"
7  }

```

Body Cookies Headers (10) Test Results ⚙ Status: 400 Bad Request Time: 22 ms Size: 468 B Save Response

Pretty Raw Preview Visualize Text ▼ ↻ 📄 🔍

1 The amount of CPU resources should be at least as much as the amount of GPU resources and at least as much as the amount of memory resources.

POST ▼ http://localhost:8082/nodes/add

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

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL JSON ▼

```

1  {
2    "cpuResources": 14.9,
3    "gpuResources": 14.9,
4    "memoryResources": 15.0,
5    "name": "node",
6    "url": "/nodeUrl"
7  }

```

Body Cookies Headers (10) Test Results ⚙ Status: 400 Bad Request Time: 22 ms Size: 468 B Save Response

Pretty Raw Preview Visualize Text ▼ ↻ 📄 🔍

1 The amount of CPU resources should be at least as much as the amount of memory resources.

POST

http://localhost:8082/nodes/add

ParamsAuthorizationHeaders (9)BodyPre-request ScriptTestsSettings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

2

3

4

5

6

7

1

2

3

4

5

6

7

Body

Cookies

Headers (10)

Test Results

Pretty

Raw

Preview

Visualize

Text

1

2

3

4

5

6

7

1 The amount of CPU resources should be at least as much as the amount of GPU resources.

POST

http://localhost:8082/nodes/add

ParamsAuthorizationHeaders (9)BodyPre-request Script

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

2

3

4

5

6

7

1

2

3

4

5

6

7

Body

Cookies

Headers (11)

Test Results

Pretty

Raw

Preview

Visualize

Text

1

2

3

4

5

6

7

1 Your node has been successfully added.

POST

http://localhost:8082/nodes/add

ParamsAuthorizationHeaders (9)BodyPre-request Script

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

2

3

4

5

6

7

1

2

3

4

5

6

7

Body

Cookies

Headers (10)

Test Results

Pretty

Raw

Preview

Visualize

Text

1

2

3

4

5

6

7

1 Failed to add node. A node with this url already exists.

POST

http://localhost:8082/nodes/add

ParamsAuthorizationHeaders (9)BodyPre-request Script

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

2

3

4

5

6

7

1

2

3

4

5

6

7

Body

Cookies

Headers (10)

Test Results

Pretty

Raw

Preview

Visualize

Text

1

2

3

4

5

6

7

1 Your node contains an invalid URL. Should start with a "/"

POST

http://localhost:8082/nodes/add

ParamsAuthorizationHeaders (9)BodyPre-request Script

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

2

3

4

5

6

7

1

2

3

4

5

6

7

Body

Cookies

Headers (10)

Test Results

Pretty

Raw

Preview

Visualize

Text

1

2

3

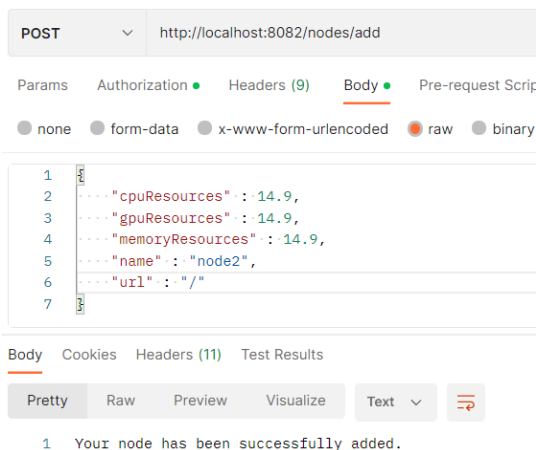
4

5

6

7

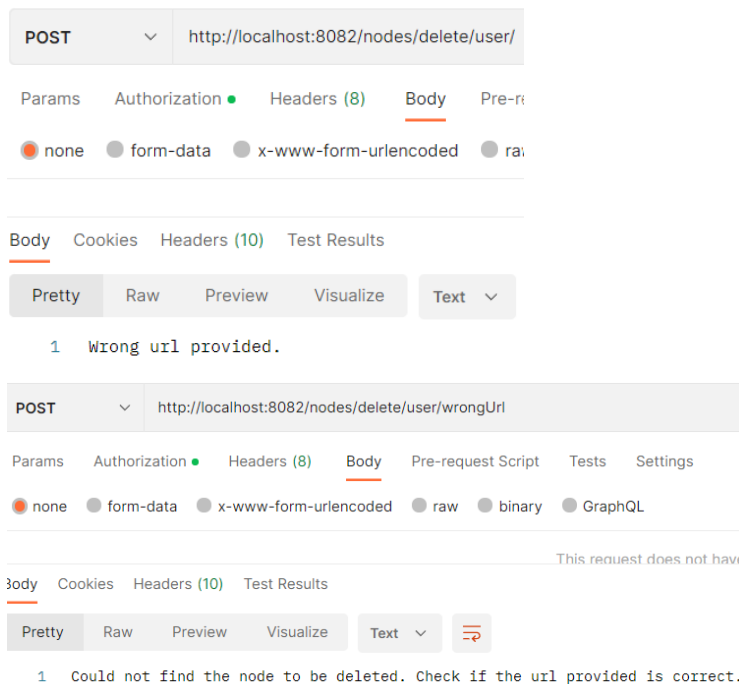
1 None of the resources can be negative.

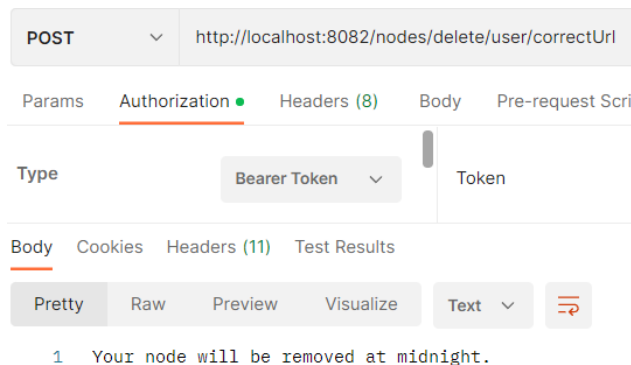
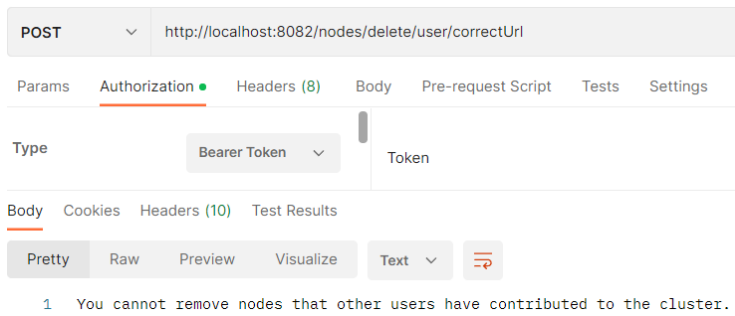


- In the last screenshot, we can see that a bug was found. Since all the urls must start with '/', an url containing only that character should not be allowed. Therefore, that is a bug and it should be fixed in the future.
- Other than that, the behavior is correct, and everything else works as expected.

Endpoint 8082/nodes/delete/user/**

- For this endpoint, we test not passing any url, passing a wrong url, and passing the url of a node that the user is not the one who contributed that node to the cluster. Finally, we pass the url of a node the user has contributed to the cluster. The last case is the only one that the removal schedulment should work.

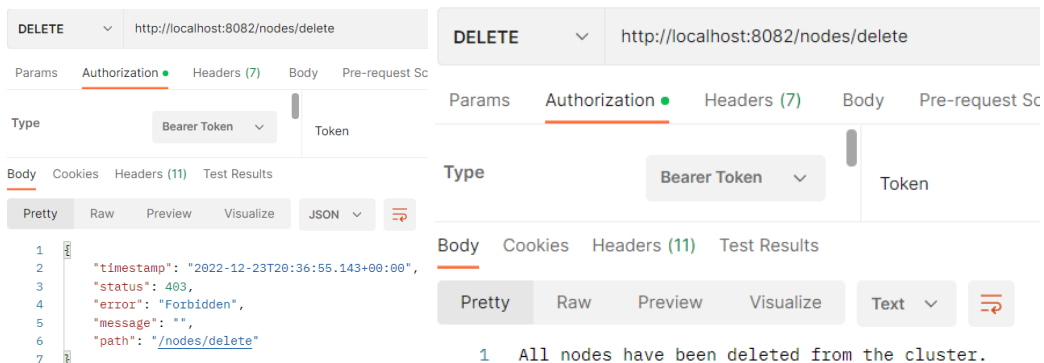




- All the cases behave as expected.

Endpoint 8082/nodes/delete/**)

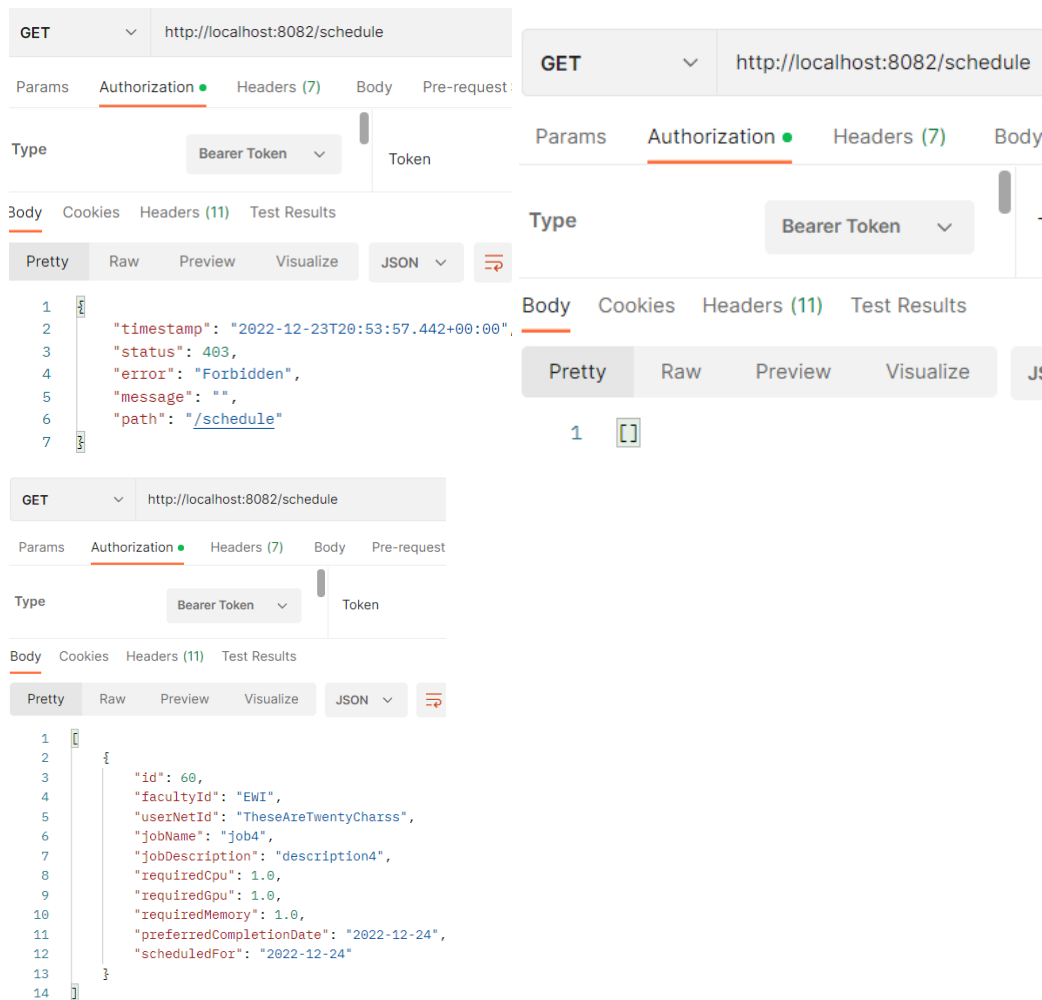
- Removes the node instantly from the cluster, should only work if the user is a sysAdmin. First, we try deleting all nodes as a normal (non admin) user, then as a sysAdmin. The nodes should only be removed in the second scenario.



- The behavior is correct.

Endpoint 8082/schedule

- Again, this can only be accessed by sysAdmins.
- We first called it from a non-admin user. After that, we tried getting the schedule from a sysAdmin account, but with no jobs scheduled. At last, we tried retrieving a filled schedule.




- The behavior for all tests was as expected.

Endpoint 8082/request


- This endpoint contains a lot of boundary cases.
- The first ones we decided to test are in terms of amount of resources, which works in the same fashion as for adding nodes. After that, we tried making a request that requires more cpu resources than what the faculty contains. We also tried making a request with no faculty specified. Furthermore, we tried making a request with a netId that is not registered, and then 3 requests with the preferred date being yesterday, today, and tomorrow, respectively. The last one should be valid to schedule.

body Cookies Headers (10) Test Results

Pretty Raw Preview Visualize Text


POST  http://localhost:8082/request

Body Cookies Headers (10) Test Results

Pretty Raw Preview Visualize Text 

POST	▼	http://localhost:8082/request
------	---	-------------------------------

Body Cookies Headers (10) Test Results

Pretty Raw Preview Visualize Text 

1 The requested job requires more resources than are assigned to the faculty.

POST http://localhost:8082/request

Params Authorization Headers (8) Body Pre-request

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

```
1 {
2   ...."facultyId": "EWI",
3   ...."userNetId": "notRegistered",
4   ...."jobName": "job4",
5   ...."jobDescription": "description4",
6   ...."requiredCpu": 1.1,
7   ...."requiredGpu": 1.1,
8   ...."requiredMemory": 1.1,
9   ...."preferredCompletionDate": "2022-12-24"
10 }
```

Body Cookies Headers (12) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "timestamp": "2022-12-23T21:14:50.754+00:00",
3   "status": 401,
4   "error": "Unauthorized",
5   "message": "",
6   "path": "/request"
7 }
```

POST http://localhost:8082/request

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

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   ...."facultyId": "EWI",
3   ...."userNetId": "TheseAreTwentyCharss",
4   ...."jobName": "job4",
5   ...."jobDescription": "description4",
6   ...."requiredCpu": 1.1,
7   ...."requiredGpu": 1.1,
8   ...."requiredMemory": 1.1,
9   ...."preferredCompletionDate": "2022-12-22"
10 }
```

Body Cookies Headers (10) Test Results

Pretty Raw Preview Visualize Text

```
1 The requested job cannot require the cluster to compute it before 2022-12-24.
```

POST http://localhost:8082/request

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

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   ...."facultyId": "EWI",
3   ...."userNetId": "TheseAreTwentyCharss",
4   ...."jobName": "job4",
5   ...."jobDescription": "description4",
6   ...."requiredCpu": 14.0,
7   ...."requiredGpu": 14.0,
8   ...."requiredMemory": 14.0,
9   ...."preferredCompletionDate": "2022-12-24"
10 }
```

Body Cookies Headers (11) Test Results

Pretty Raw Preview Visualize Text

```
1 Successfully scheduled job.
```

- The endpoint behaves correctly. One change that could be made for improvement is when trying to make a request from a non-existing faculty, return a message saying

that the faculty does not exist. Currently, we say that the resources assigned to that faculty are not enough.

Endpoint 8082/resources/assigned

- Only admins can check the assigned resources per faculties. Faculties can see their own assigned resources, but not from other faculties.
- We first tried retrieving the resources as a normal user. Then we call it from the faculty. account. After that, we call it from an admin account with resources filled and with no resources for any faculties.

The first screenshot shows a GET request to `http://localhost:8082/resources/assigned` with a Bearer Token. The response is a 403 Forbidden error with the following body:

```
{
  "timestamp": "2022-12-23T21:33:25.571+00:00",
  "status": 403,
  "error": "Forbidden",
  "message": "",
  "path": "/resources/assigned"
}
```

The second screenshot shows a GET request to `http://localhost:8082/resources/assigned` with a Bearer Token. The response is a JSON array of resource information for the faculty 'AE':

```
[
  {
    "facultyName": "AE",
    "resourceCpu": 0.0,
    "resourceGpu": 0.0,
    "resourceMemory": 0.0
  }
]
```

The third screenshot shows a GET request to `http://localhost:8082/resources/assigned/facEWI` with a Bearer Token. The response is a JSON array of resource information for the faculty 'EWI':

```
[
  {
    "facultyName": "EWI",
    "resourceCpu": 14.0,
    "resourceGpu": 14.0,
    "resourceMemory": 14.0
  }
]
```

- In the last screenshot we tried getting the resources from a faculty account (facEWI) that has resources assigned to it. Unfortunately, we found a bug. It does not show anything. This should be fixed in the future.

Endpoint 8082/resources/reserved(/date&facultyId)

- The first thing we tried was getting the resources from all faculties from a day in the past, which should not work. Then we tried getting the reserved resources from all the faculties today, which should work. The next boundary test we tried was getting the reserved resources for all days, first for a faculty that does not exist, and then for one that exists. Only the latter should work properly, as the first one should give a bad request output. Proceeding that, we specified the date and the faculty. Again, we check combinations of past dates with existent and non-existent faculties, as well as current and future days with existent and non-existent faculties. Only current and future days with existent faculties should give an output.
- NOTE: If there are no resources assigned to the specified faculty, nothing should appear. Namely, you will see an empty newline on postman's terminal.