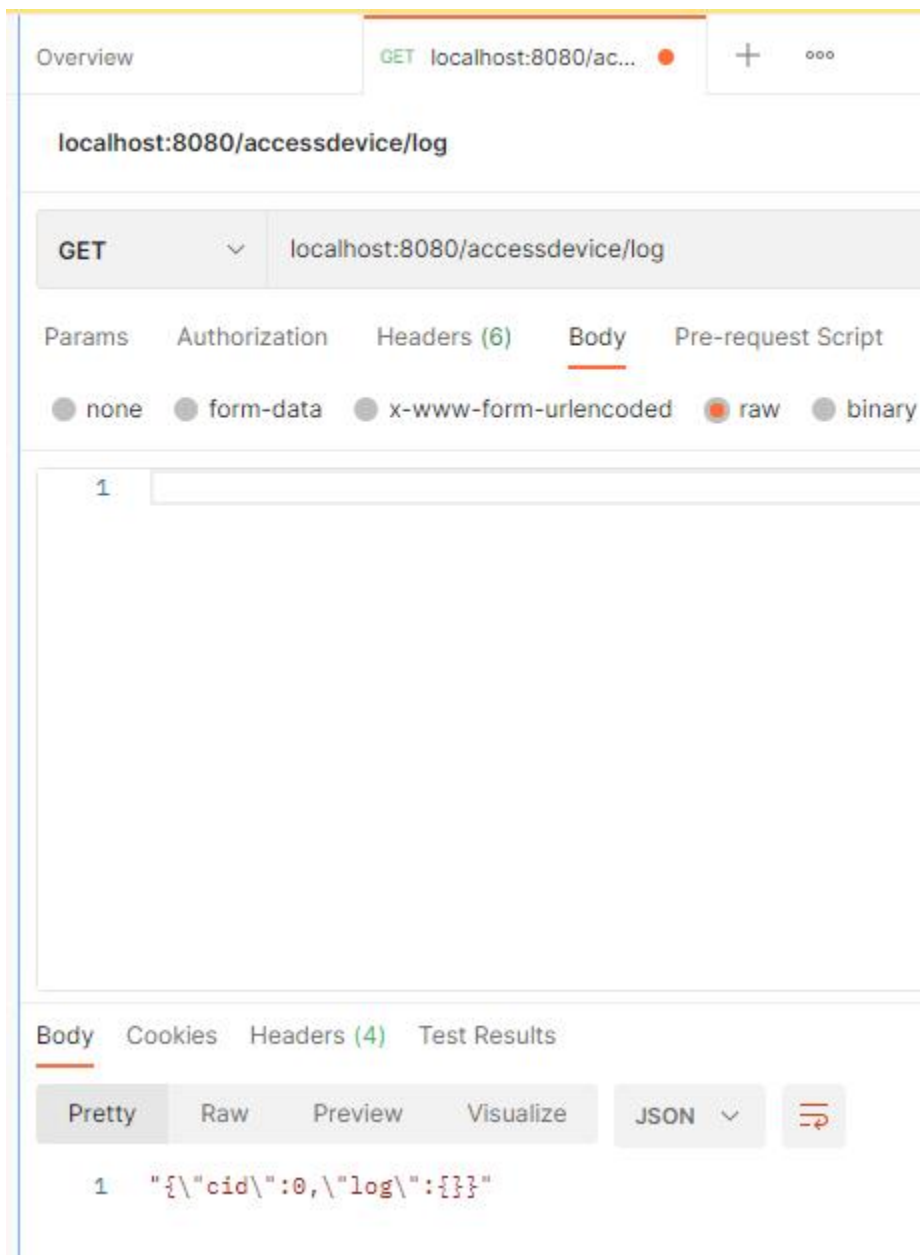


Screenshots av tester

1. Tom logg (GET på /log)



2. POST mot log

localhost:8080/accessdevice/log

POST localhost:8080/accessdevice/log

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded **raw** binary GraphQL Text

```
1 {
2   ... "message": "locked"
3 }
```

Body Cookies Headers (4) Test Results 200 C

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 0,
3   "message": "locked"
4 }
```

(kjører flere ganger for å fylle opp loggen)

3. GET mot en logg som når fått mer data

localhost:8080/accessdevice/log

Save



GET

localhost:8080/accessdevice/log

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookie

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

```
1 {
2   .. "message": "locked"
3 }
```

Body Cookies Headers (4) Test Results

200 OK 5 ms 332 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1 {\"cid\":4,\"log\":{\"0\":{\"id\":0,\"message\":\"locked\"},\"1\":{\"id\":1,\"message\":\"locked\"},
  \"2\":{\"id\":2,\"message\":\"locked\"},\"3\":{\"id\":3,\"message\":\"locked\"}}}
```

4. DELETE mot loggen. Viser en tom logg etter det

The screenshot shows a REST client interface with a DELETE request to `localhost:8080/accessdevice/log`. The request is configured with the method `DELETE` and the URL `localhost:8080/accessdevice/log`. The response is a `200 OK` status with a `11 ms` response time and a `173 B` body. The response body is displayed in JSON format, showing a JSON object with a `cid` property set to `0` and a `log` property set to an empty object `{}`.

Overview DEL localhost:8080/ac... + ... No Environment

localhost:8080/accessdevice/log Save ▼

DELETE ▼ localhost:8080/accessdevice/log

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **Text** ▼

1

Body Cookies Headers (4) Test Results 200 OK 11 ms 173 B Save

Pretty Raw Preview Visualize **JSON** ▼ ≡

1 `{"cid":0,"log":{"}}`

5. GET på /code

localhost:8080/accessdevice/code


GET localhost:8080/accessdevice/code

Params Authorization Headers (6) Body Pre-request Script Tests S

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ Graph

1

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON 

```
1 {
2   "accesscode": [
3     1,
4     2
5   ]
6 }
```

6. PUT på /code for å sette ny kode

localhost:8080/accessdevice/code Save

PUT localhost:8080/accessdevice/code

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **Text**

```
1 {
2   "accesscode": [
3     2,
4     1
5   ]
6 }
```

Body Cookies Headers (4) Test Results 200 OK 10 ms 169 B Save

Pretty Raw Preview Visualize JSON

```
1 {
2   "accesscode": [
3     2,
4     1
5   ]
6 }
```

Og en ny GET på /code etter det ->

The screenshot shows a REST client interface with the following components:

- URL Bar:** Displays `localhost:8080/accessdevice/code` and a **Save** button.
- Request Tab:** Shows a **GET** request to `localhost:8080/accessdevice/code`. Below the URL are tabs for **Params**, **Authorization**, **Headers (6)**, **Body** (selected), **Pre-request Script**, **Tests**, and **Settings**.
- Body Type:** A row of radio buttons for **none**, **form-data**, **x-www-form-urlencoded**, **raw** (selected), **binary**, and **GraphQL**. A **Text** dropdown is on the right.
- Request Body:** A text area with a line number **1** and an empty input field.
- Response Section:** Contains tabs for **Body** (selected), **Cookies**, **Headers (4)**, and **Test Results**. On the right, it shows a status of **200 OK**, a time of **5 ms**, and a size of **169 B**.
- Response Body:** A code editor with tabs for **Pretty** (selected), **Raw**, **Preview**, and **Visualize**. A **JSON** dropdown and a refresh icon are on the right. The JSON content is:

```
1 {
2   "accesscode": [
3     2,
4     1
5   ]
6 }
```

7. ny POST mot /log til test av GET med ID

localhost:8080/accessdevice/log Save

POST localhost:8080/accessdevice/log

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **Text**

```
1 {
2   "message": "HEISANN!!"
3 }
```

Body Cookies Headers (4) Test Results 200 OK 13 ms 179 B

Pretty Raw Preview Visualize **JSON**

```
1 {
2   "id": 3,
3   "message": "HEISANN!!"
4 }
```


8. GET på /log/ igjen

localhost:8080/accessdevice/log

Save

GET

localhost:8080/accessdevice/log

Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookie:

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

Text

```
1  {
2    "message": "HEISANN!!"
3  }
```

Body Cookies Headers (4) Test Results 200 OK 5 ms 335 B Save Response

Pretty

Raw


Preview

Visualize

JSON


```
1  [{"cid":4,"log":{"0":{"id":0,"message":"locked"},"1":{"id":1,"message":"locked"},
    "2":{"id":2,"message":"locked"},"3":{"id":3,"message":"HEISANN!!"}}}]
```

9. GET på /log/3 for å vise søk i log med ID


localhost:8080/accessdevice/log/3 



GET localhost:8080/accessdevice/log/3

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **Text** 

```
1 {
2   ... "message": "HEISANN!!"
3 }
```

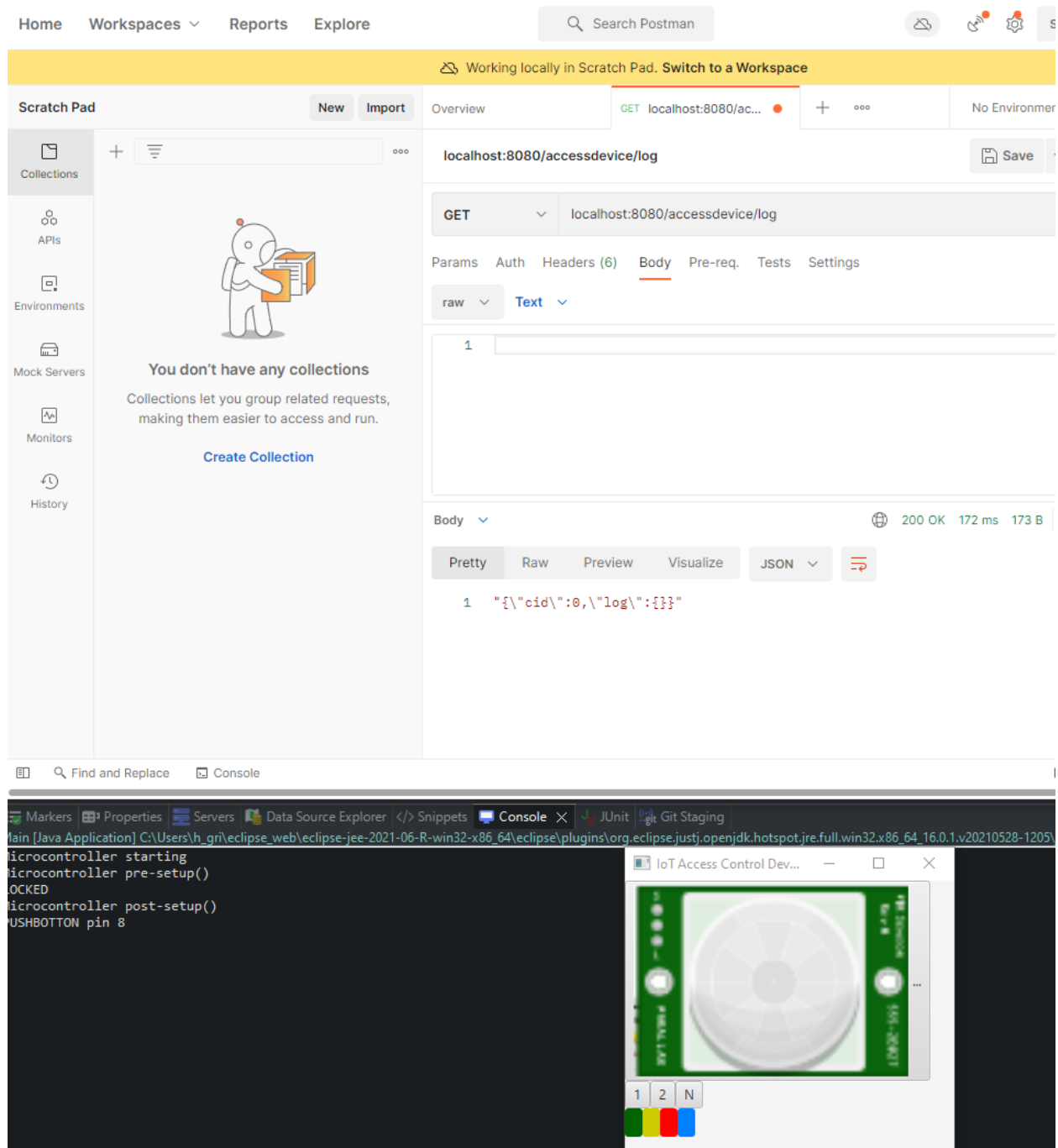
Body Cookies Headers (4) Test Results  200 OK 4 ms

Pretty Raw Preview Visualize JSON  

```
1 {
2   "id": 3,
3   "message": "HEISANN!!"
4 }
```

Screenshots Step 5: system testing

Starter REST-apiet og satt i gang javafx-guien. Koblet til nettverket (blå lys)



Aktiverer PIR og taster inn 2, 2 som kode. Det er feil kode, og tilgang gis ikke. Loggen viser også ACCESS DENIED.

Taster nå inn korrekt kode (1,2). Den åpnes, før den så automatisk lukkes igjen:

The image shows two windows from a development environment. The top window is Postman, displaying a GET request to `localhost:8080/accessdevice/log`. The response body is a JSON array with three log entries. The second entry, with `"id": 1`, has the message `"UNLOCKED"`, which is underlined in red. The third entry, with `"id": 2`, has the message `"LOCKED"`, also underlined in red. Red handwritten numbers '1' and '2' are placed below the respective log entries. The bottom window is a Java application console showing a sequence of events: `WAIT2P`, `PUSHBUTTON pin 7`, `CHECKING`, `UPDATING CODE`, `LOCKED`, `{"id":0,"message":"ACCESS DENIED"}`, `PIR SENSOR pin 2`, `WAIT1P`, `PUSHBUTTON pin 6`, `WAIT2P`, `PUSHBUTTON pin 7`, `CHECKING`, `UPDATING CODE`, `UNLOCKED`, `{"id":1,"message":"UNLOCKED"}` (with a red '1' next to it), `{"id":2,"message":"LOCKED"}` (with a red '2' next to it), and `LOCKED`. To the right of the console is a small window titled 'IoT Access Control Dev...' showing a graphical interface of a device with a circular sensor and buttons.

localhost:8080/accessdevice/log

GET localhost:8080/accessdevice/log

Params Auth Headers (6) Body Pre-req. Tests Settings Cookies

raw Text

1

Body 200 OK 5 ms 301 B Save Response

Pretty Raw Preview Visualize JSON

```
1 [{"cid":3,"log":{"0":{"id":0,"message":"ACCESS DENIED"},"1":{"id":1,"message":"UNLOCKED"},"2":{"id":2,"message":"LOCKED"}}}]
```

1 2

Markers Properties Servers Data Source Explorer Snippets Console X JUnit Git Staging

Main [Java Application] C:\Users\h_gri\workspace\j2ee-2021-06-R-win32-x86_64\workspace\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.16.0.1.v20210528-1205\jre\bin\javaw.exe (28. apr. 2021)

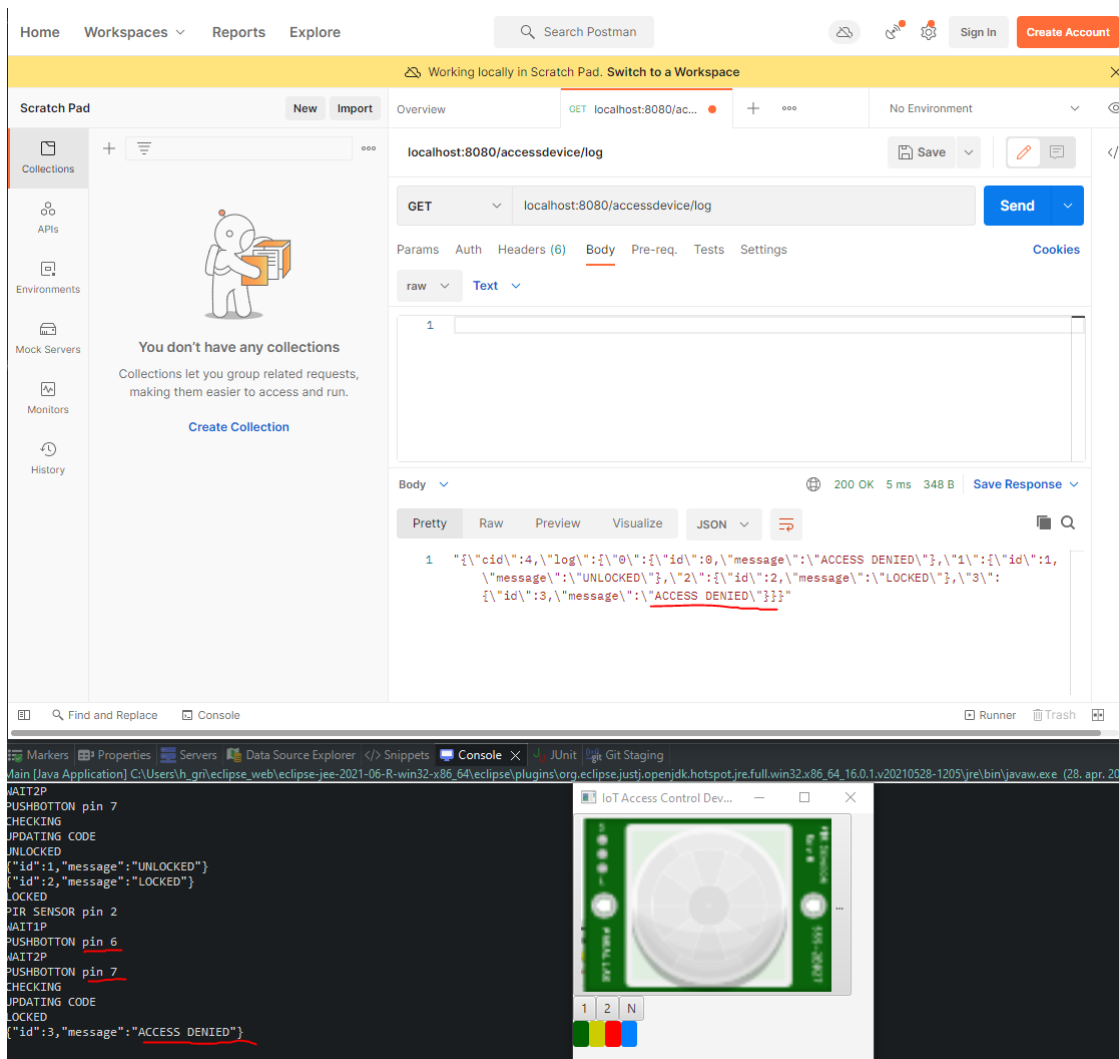
WAIT2P
PUSHBUTTON pin 7
CHECKING
UPDATING CODE
LOCKED
{ "id":0,"message":"ACCESS DENIED" }
PIR SENSOR pin 2
WAIT1P
PUSHBUTTON pin 6
WAIT2P
PUSHBUTTON pin 7
CHECKING
UPDATING CODE
UNLOCKED
{ "id":1,"message":"UNLOCKED" } 1
{ "id":2,"message":"LOCKED" } 2
LOCKED

IoT Access Control Dev...

Endre kode via PUT /code i Postman. Ny kode 2,2.



Forsøker første å taste inn gammel kode (1,2). Dette gir ACCESS DENIED, som forventet.



Prøver så med koden (2,2), og det fungerer:

The image shows a screenshot of a web application interface for an IoT Access Control Device. The interface is divided into two main sections: a left sidebar with navigation options and a main content area.

Left Sidebar:

- Collections
- APIs
- Environments
- Mock Servers
- Monitors
- History

Main Content Area:

At the top, there is a message: "You don't have any collections. Collections let you group related requests, making them easier to access and run. [Create Collection](#)".

Below this, there is a section for a REST client. The URL is `localhost:8080/accessdevice/log`. The method is `GET`. The response status is `200 OK` with a response time of `4 ms` and a size of `430 B`. The response body is displayed in JSON format:

```
1 [{"cid":6,"log":{"0":{"id":0,"message":"ACCESS DENIED"},"1":{"id":1,"message":"UNLOCKED"},"2":{"id":2,"message":"LOCKED"},"3":{"id":3,"message":"ACCESS DENIED"},"4":{"id":4,"message":"UNLOCKED"},"5":{"id":5,"message":"LOCKED"}}}]
```

Handwritten red annotations are present on the JSON response: a red '1' under the first log entry and a red '2' under the last log entry.

Below the REST client, there is a console window showing the following log output:

```
WAIT2P
PUSHBUTTON pin 7
CHECKING
UPDATING CODE
LOCKED
{"id":3,"message":"ACCESS DENIED"}
WAIT1P
PIR SENSOR pin 2
PUSHBUTTON pin 7
WAIT2P
PUSHBUTTON pin 7
CHECKING
UPDATING CODE
UNLOCKED
{"id":4,"message":"UNLOCKED"}
{"id":5,"message":"LOCKED"}
LOCKED
```

Handwritten red annotations are present on the console output: a red circle around the `"UNLOCKED"` message and a red circle around the `"LOCKED"` message.

At the bottom right, there is a small window titled "IoT Access Control Dev..." showing a graphical representation of the device with a circular sensor and a display screen.

Av loggen ser vi at den åpnes og lukkes, som den skal.

