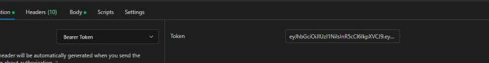


The screenshot shows the Burp Suite interface with the 'POST' tab selected for the request to `http://localhost:3000/book`. The 'Params' tab is active, displaying a table of request parameters. The table has two columns: 'Key' and 'Value'. The parameters are: 'name' (Lemonmellow's kitchen Martini), 'author' (Douglas Adams), 'isbn' (978-951-0-11766-3), and 'tag' (None). The 'Body' tab is also visible, showing a JSON payload.

Key	Value
<input checked="" type="checkbox"/> name	Lemonmellow's kitchen Martini
<input checked="" type="checkbox"/> author	Douglas Adams
<input checked="" type="checkbox"/> isbn	978-951-0-11766-3
<input type="checkbox"/> tag	None



The screenshot shows the AWS IAM console interface. The 'Users' tab is selected, displaying a list of users. The user 'cg9AG03UJ2M2N004845C86gK8E9p...' is highlighted. The 'Authentication' tab is also visible, showing a list of authentication keys. The 'Test Results' tab is also visible, showing a successful test result for the 'cg9AG03UJ2M2N004845C86gK8E9p...' user.

The screenshot shows the Burp Suite interface with the following details:

- URL:** http://localhost:3000/book/5
- Method:** POST
- Headers:**
 - name: form-data
 - Content-Type: application/x-www-form-urlencoded
 - Accept: */*
 - Origin: http://localhost:3000
 - Referer: http://localhost:3000/book/5
- Body:**

```
{
  "name": "Jaggschulcher peksika",
  "author": "Julia Vuorenen",
  "isbn": "956-555-444-353",
  "city": "Helsinki"
}
```
- Response:**
 - Status: 200 OK
 - Time: 10 ms
 - Size: 203 B
 - Content-Type: application/json
- Body:**

```
{
  "id": 5,
  "idCount": "1",
  "insertId": "1",
  "info": "book matched",
  "1 Changed": "1",
  "warningId": "0",
  "serverState": "1",
  "warningState": "0",
  "changedInfo": "1"
}
```

```
mysql> select * from book;
```

	id_book	name	author	isbn
1	1	PHP Basic	Bob Jones	123-456-789-111-x
2	2	Statistics	Lisa Smith	222-333-444-555-y
3	3	The C Programming Language	Denis Richie	222-333-222-333
4	4	The C++ Programmin Language	Bjarne Stroustrup	222-777-666-111
5	5	Juoppohullun päiväkirja	Juha Vuorinen	666-555-444-333
6	6	Linnunradan käsikirja liftareille	Douglas Adams	978-951-0-11756-3

6 rows in set (0.00 sec)

The screenshot shows the Burp Suite interface with the HTTP history tab active. The selected request is a POST to `/api/book/5` with a `Content-Type: application/json` header. The body contains a JSON object representing a book. The response status is 200 OK. The interface includes a top bar with the URL, a search bar, and a 'Send' button. Below the URL bar are tabs for Params, Authentication, Headers, Body, Scripts, and Settings. The main area is divided into three sections: Type (Bearer Token), Token, and a large text area for the request body. The bottom section shows the raw request and response details, including the request body and the response body.

The screenshot displays the VS Code editor with a REST client file named `books.http` in the `src` directory. The file contains a `GET` request to `http://localhost:3000/book` with an `Authorization` header. The response is a JSON array of book objects.

REST Client Request:

```
GET http://localhost:3000/book

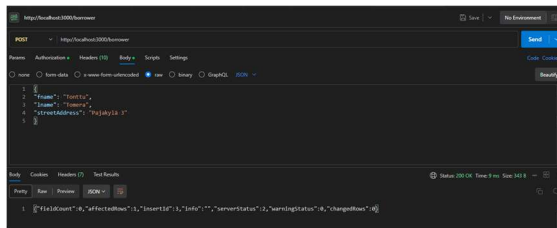
Authorization: Bearer token
```

REST Client Response:

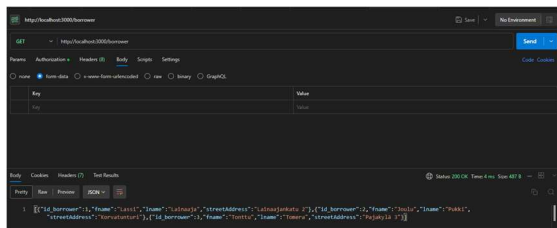
```
[{"id_book":1,"name":"PHP Basic","author":"Bob Jones","isbn":"123-456-789-111-1","id_book":2,"name":"Statistics","author":"Lisa Smith","isbn":"222-333-444-555-7"}, {"id_book":3,"name":"The C Programming Language","author":"Dennis Ritchie","isbn":"222-333-222-333"}, {"id_book":4,"name":"The C++ Programming Language","author":"Bjarne Stroustrup","isbn":"222-777-666-111"}, {"id_book":5,"name":"Linearvarden knikikirje liltareville","author":"Douglas Adams","isbn":"999-999-0-11796-9"}]
```

The `Output` panel shows the raw JSON response of the REST client. The status bar at the bottom indicates the file is `src/books/001-1.562 mb - 479`.

Borroweriin luodana POST komennolla uusi lainaaja käytetään tällä kertaan raw jsonia, eikä x-www-form-urlencoded:

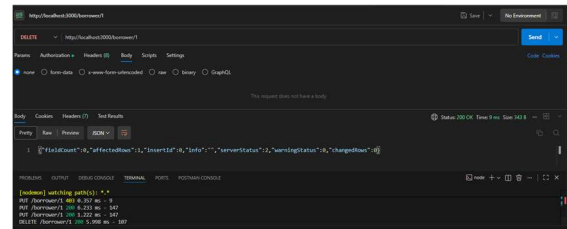


Tsekataan lainaajat GET:

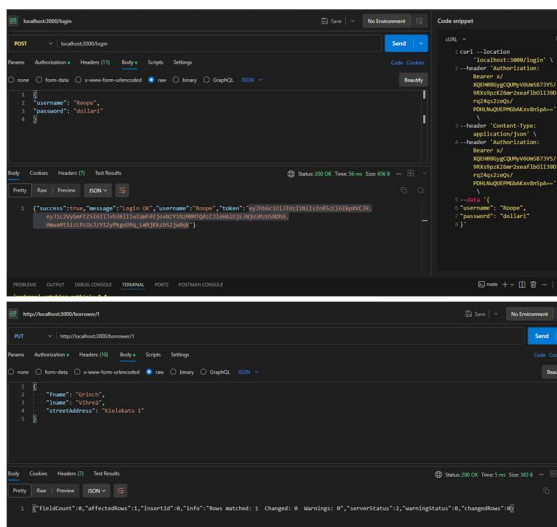
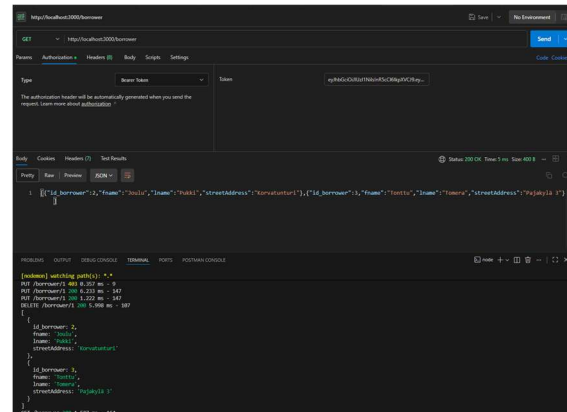


Huomaattiin että Lasse Lainaja olikin oikeasti joulun riesa Grinch ja tokenikin ehti vanhentua tässä välissä. Uusittiin Tokeni ja päivitettiin PUT komennolla id1

Grinch ei pilaa tätä joulua, niin annetaan sille DELETEä:



Status check GET:



```
mysql> select * from borrower;
```

id_borrower	fname	lname	streetAddress
1	Grinch	Vihreä	Kielekatu 1
2	Joulu	Pukki	Korvatunturi
3	Tonttu	Tomera	Pajakylä 3

3 rows in set (0.00 sec)