FINÁLNÍ PROJEKT
č.1

Autor:Jaroslav Pátek (Discord: patekxd |Server name: Jaroslav P.)
Datum: 1.7.2024

OBSAH

ZADÁNÍ

Čílem finálního projektu je otestovat funkčnost aplikace, která slouží k manipulaci s daty o studentech. Aplikace má rozhraní REST-API, které umožňuje vytvoření, smazání a získání dat..

Poznámky:

Nezapomeňte, že v IT se data musí někde uložit a poté získat. Proto ověřte, že data jsou správně uložena a získávána z databáze.
Nezapomeňte do testovacích scénářů uvést testovací data, očekávaný výsledek včetně těla odpovědi a stavových kódů.

TESTOVACÍ SCÉNÁŘE

Na základě uvedených testovacích scénářů jsem ověřil(a) funkčnost aplikace.

Test #1

Abstract: Get data about existing student

Test data preparation:

Existing student id=242

SELECT * FROM student WHERE id=242;
‘242’ ‘34’ ‘komar@seznam.cz’ ‘Martin’ ‘KOMÁR’

Steps:

GET http://108.143.193.45:8080/api/v1/students/242

Expected result:

1) GET http://108.143.193.45:8080/api/v1/students/242
2) Press send button
3) status code: 200 OK
4) response:

{
“id”: 242
“first_name”: Martin
“last_name”: KOMÁR
“email”: komar@seznam.cz
“age”: 34
}

Test #2

Abstract: Get data about nonexistent student

Test data preparation:

Nonexistent student id=-1

SELECT * FROM student WHERE id=-1;
‘null’ ‘null’ ‘null’ ‘null’ ‘null’

Steps:

GET http://108.143.193.45:8080/api/v1/students/-1

Expected result:

1) GET http://108.143.193.45:8080/api/v1/students/-1
2) Press send button
3) status code: 404 Not found

Test #3

Abstract: Get data using a non integer value

Test data preparation:

Non existing student id=null

SELECT * FROM student WHERE id=null;
'null' 'null' 'null' 'null' 'null'

Steps:

GET http://108.143.193.45:8080/api/v1/students/null

Expected result:

1) GET http://108.143.193.45:8080/api/v1/students/null
2) Press send button
3) status code: 400 Bad request

Test #4

Abstract: Delete all student data of an existing student

Test data preparation:

Existing student id=342

SELECT * FROM student WHERE id=342;
342 25 petrA@sova.cz Petra SOVA

Steps:

DELETE http://108.143.193.45:8080/api/v1/students/342

Expected result:

1) DELETE http://108.143.193.45:8080/api/v1/students/342
2) Press send button
3) All data about particular student are removed

Test #5

Abstract: Delete all student data of a non existing student

Test data preparation:

Non existing student id=3

SELECT * FROM student WHERE id=3;

null null null null null

Steps:

DELETE http://108.143.193.45:8080/api/v1/students/3

Expected result:

1) DELETE http://108.143.193.45:8080/api/v1/students/3
2) Press send button
3) status: 404 not found

Test #6

Abstract: Delete data of student whose id is not an integer value

Test data preparation:

Non existing student id=null

SELECT * FROM student WHERE id=null;

'null' 'null' 'null' 'null' 'null'

Steps:

DELETE http://108.143.193.45:8080/api/v1/students/null

Expected result:

1) DELETE http://108.143.193.45:8080/api/v1/students/null
2) Press send button
3) status code: 400 Bad request

Test #7

Abstract: Create new student using post method

Test data preparation:

```
{

    "firstName":"Willy",
    "lastName":"Zaney",
    "email":"willz95@gmail.com",
    "age":30
}
```

Steps:

POST http://108.143.193.45:8080/api/v1/students/

Body -> raw -> JSON:

```
{

    "firstName":"Willy",
    "lastName":"Zaney",
    "email":"willz95@gmail.com",
    "age":30
}
```

Expected result:

1) POST http://108.143.193.45:8080/api/v1/students/ | Body -> raw -> JSON:
   ```
   {
       "firstName":"Willy",
       "lastName":"Zaney",
       "email":"willz95@gmail.com",
       "age":30
   }
   ```
2) Press send button
3) status code: 201 created | database record matches input | REST API response:
   ```
   {
       "firstName":"Willy",
       "lastName":"Zaney",
       "email":"willz95@gmail.com",
       "age":30
   }
   ```

Test #8

Abstract: Creating new student with the POST method without mandatory data

Test data preparation:

```
{

    "firstName":"Don",
    "lastName":"Key",

    "age":30
}
```

Steps:

POST http://108.143.193.45:8080/api/v1/students/

Body -> raw -> JSON:

```
{

    "firstName":"Don",
    "lastName":"Key",

    "age":30
}
```

Expected result:

1) POST http://108.143.193.45:8080/api/v1/students/ | Body -> raw -> JSON:

```
{

    "firstName":"Don",
    "lastName":"Key",

    "age":30
}
```

2) Press send button
3) status code: 400 bad request, error message: “email, lastName are mandatory”

Test #9

Abstract: Student records are saved as lowerCase() in the database

Test data preparation:

```
{

    "firstName":"ChRiS p.",
    "lastName":"BaCoN",
    "email": "eBlAn123@yourmail.com",
    "age":20
}
```

Steps:

POST http://108.143.193.45:8080/api/v1/students/

Body -> raw -> JSON:

```
{

    "firstName":"ChRiS p.",
    "lastName":"BaCoN",
    "email": "eBlAn123@yourmail.com",
    "age":20
}
```

Expected result:

1) POST http://108.143.193.45:8080/api/v1/students/ | Body -> raw -> JSON:
   ```
   {

       "firstName":"ChRiS p.",
       "lastName":"BaCoN",
       "email": "eBlAn123@yourmail.com",
       "age":20
   }
   ```
2) Press send button
3) status code: 201 created | new student information matches user input and is saved as lowercase in the database | REST API response:

```
{
        "id": 1331,
        "firstName": "ChRiS p.",
        "lastName": "BACON",
        "email": "eBlAn123@yourmail.com",
        "age": 20
}
```


Test #10

Abstract: Age of new student is in range between 1-150

Test data preparation:


```
{

        "firstName":"Easton",
        "lastName":"West",
        "email": "eastonW@gmail.com",
        "age": 151
}
```


Steps:

POST http://108.143.193.45:8080/api/v1/students/

Body -> raw -> JSON:


```
{

        "firstName":"Easton",
        "lastName":"West",
        "email": "eastonW@gmail.com",
        "age": 151
}
```


Expected result:

1) POST http://108.143.193.45:8080/api/v1/students/ | Body -> raw -> JSON:


```
{

        "firstName":"Easton",
        "lastName":"West",
        "email": "eastonW@gmail.com",
        "age": 151
 }
```

2) Press send button
3) status code: 400 bad request | error message: “age must be a number in range of 1-150

Test #11

Abstract: Email is in invalid format when creating new student using POST method

Test data preparation:

```
{

    "firstName":"Jack",
    "lastName":"Pott",
    "email": "123456789",
    "age": 20
}
```

Steps:

POST http://108.143.193.45:8080/api/v1/students/

Body -> raw -> JSON:

```
{

    "firstName":"Jack",
    "lastName":"Pott",
    "email": "123456789",
    "age": 20
```

Expected result:

1) POST http://108.143.193.45:8080/api/v1/students/ | Body -> raw -> JSON:

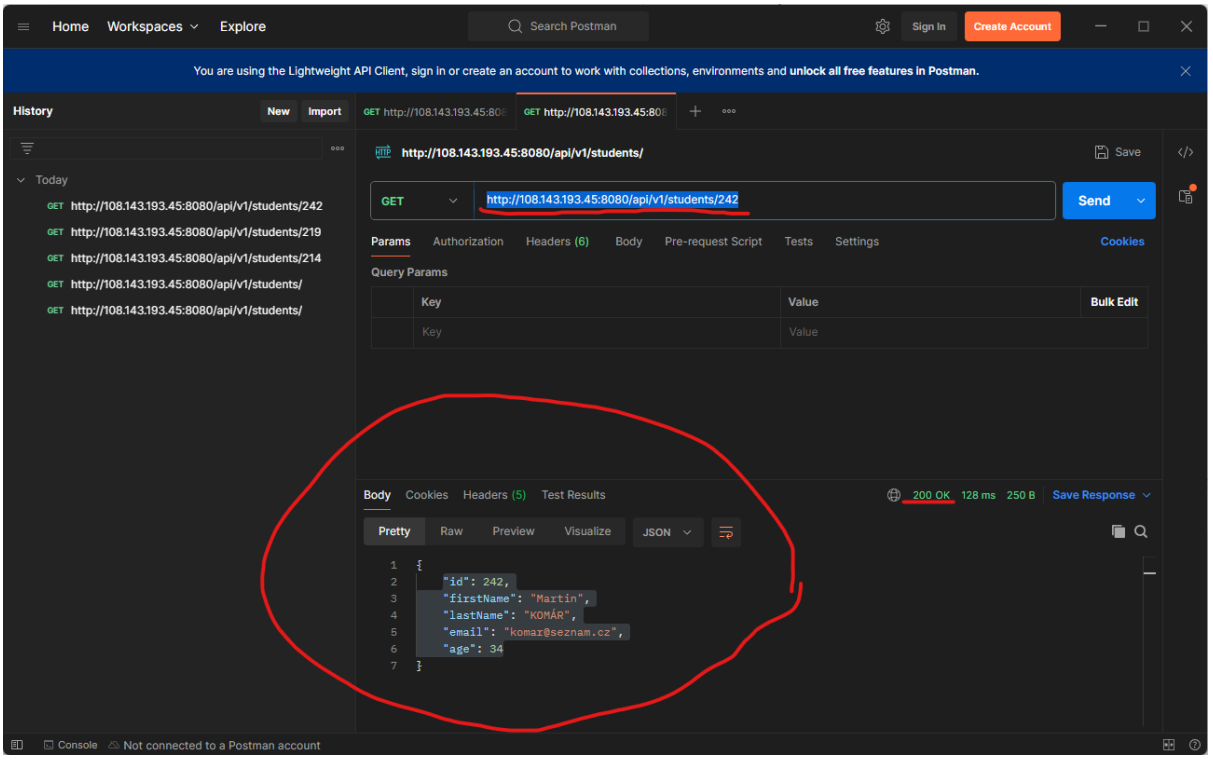
```
{

    "firstName":"Jack",
    "lastName":"Pott",
    "email": "123456789",
    "age": 20
}
```

2) Press send button
3) status code: 400 bad request | error message: “email must include @ and a valid domain”

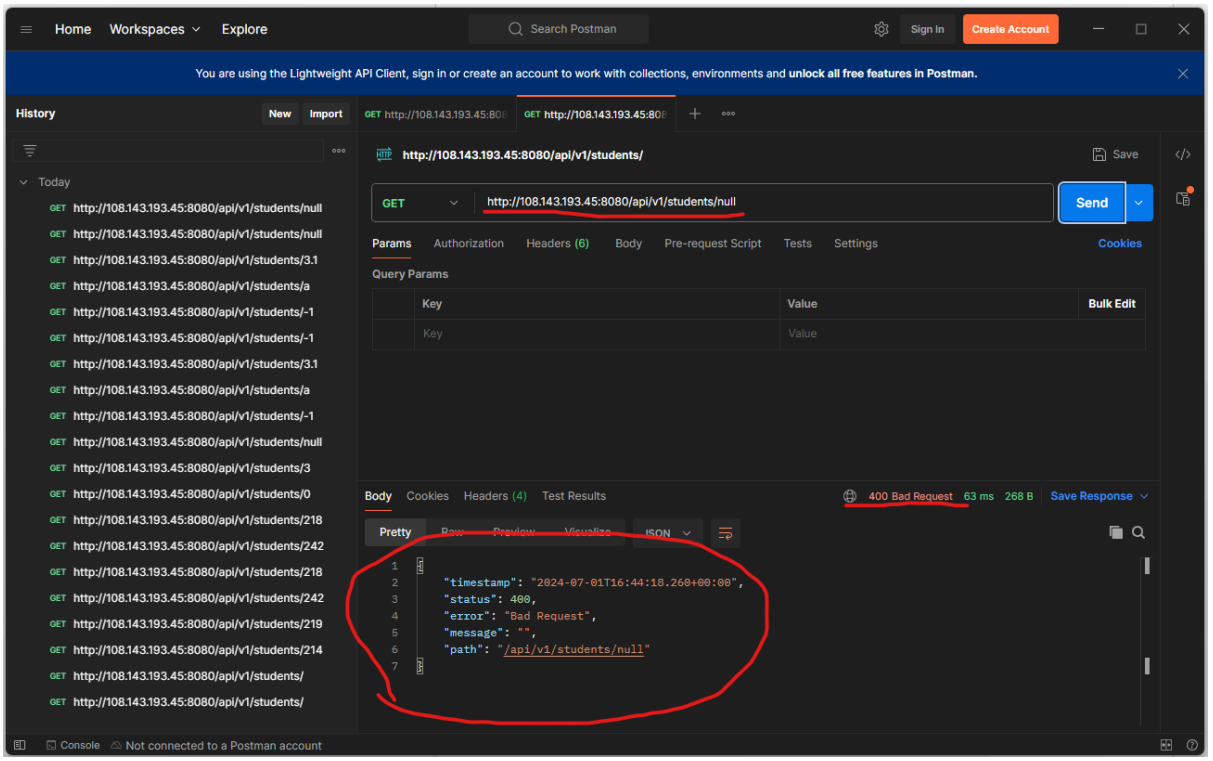
EXEKUCE TESTŮ

Testovací scénáře jsem provedl(a), přikládám výsledky testů.

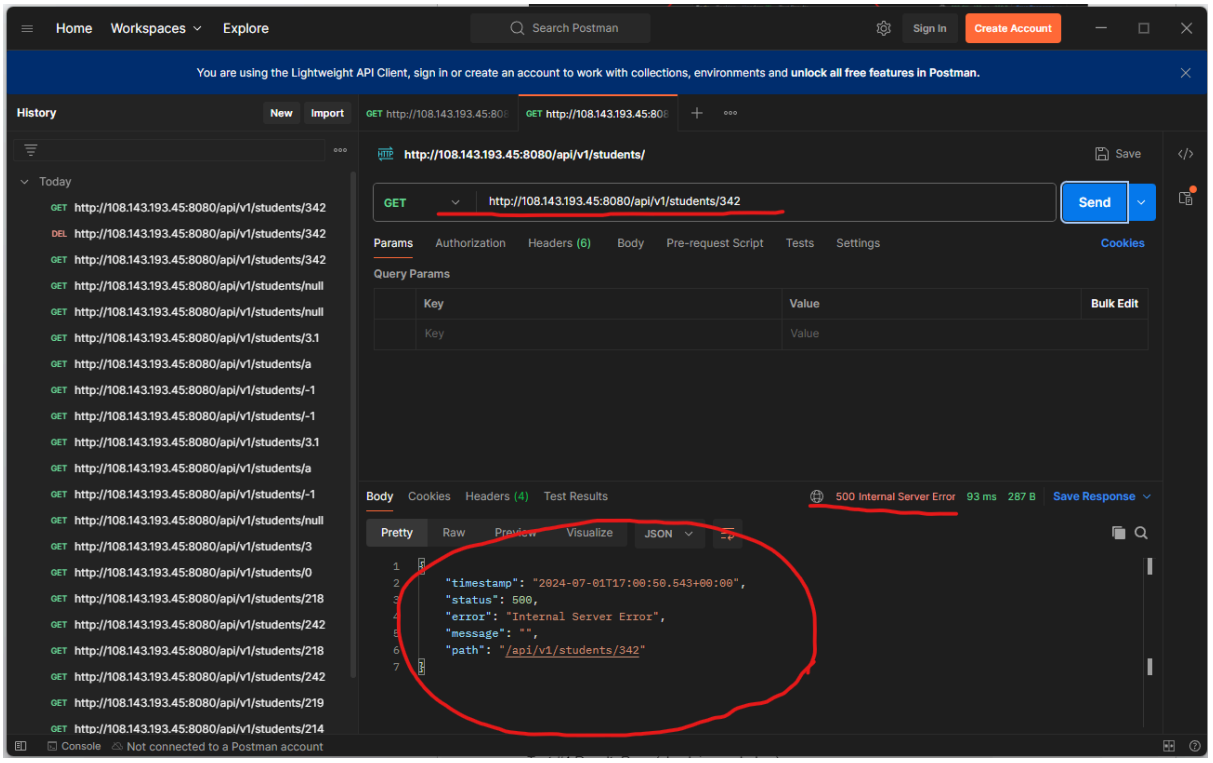
Test #1 Result: Pass (check image below)



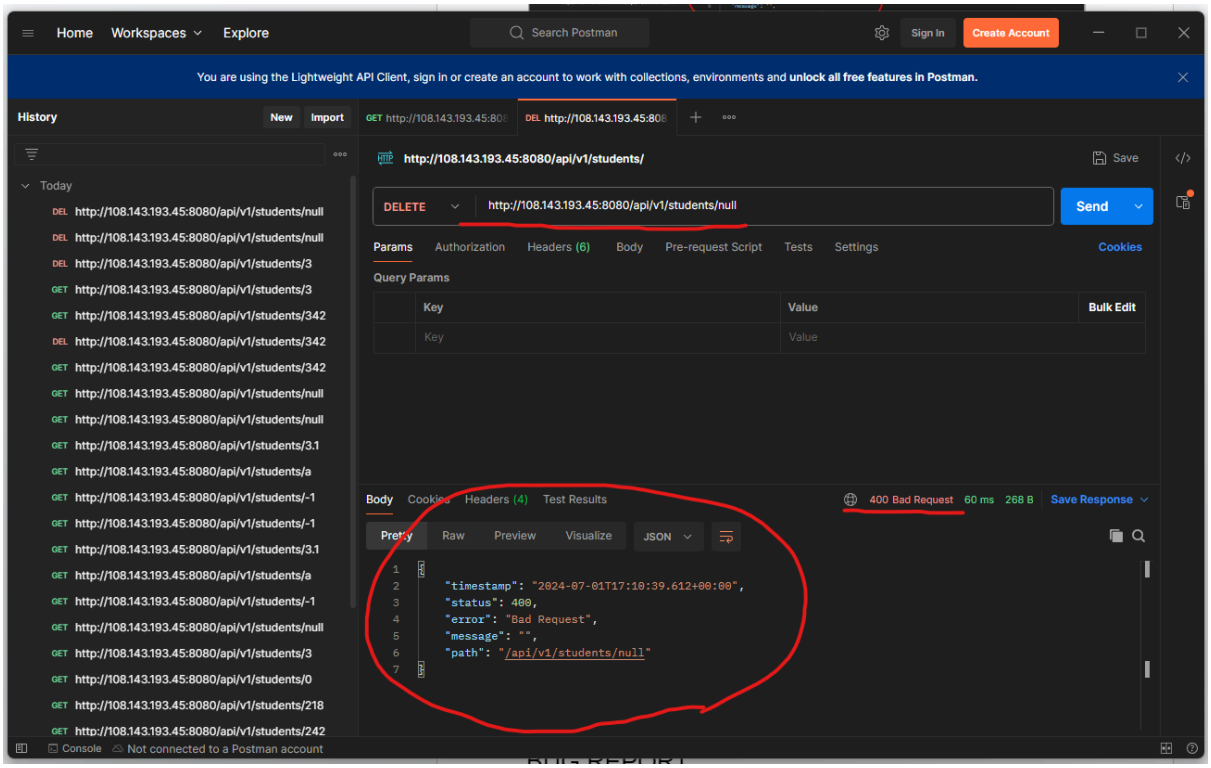
Test #3 Result: Pass (check image below)



Test #4 Result: Pass (check image below)



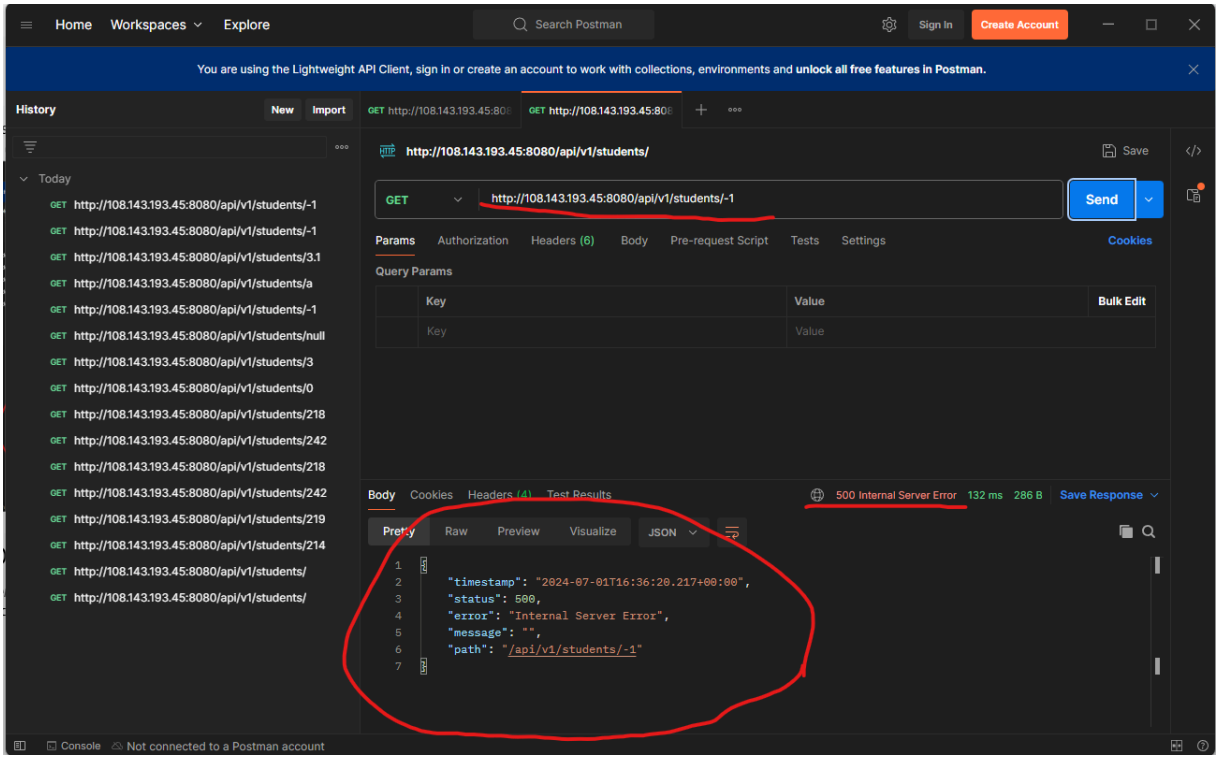
Test #6 Result: Pass (check image below)



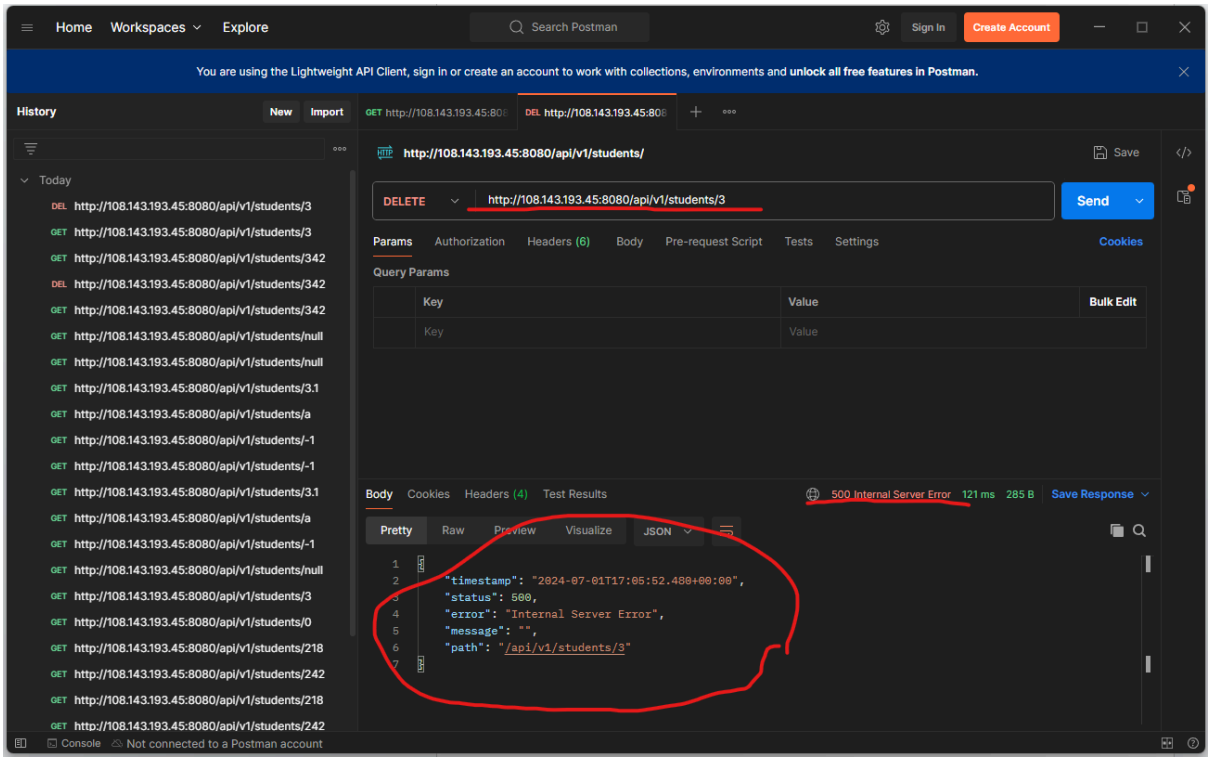
BUG REPORT

Na základě provedených scénářů jsem objevil(a) uvedené chyby aplikace.

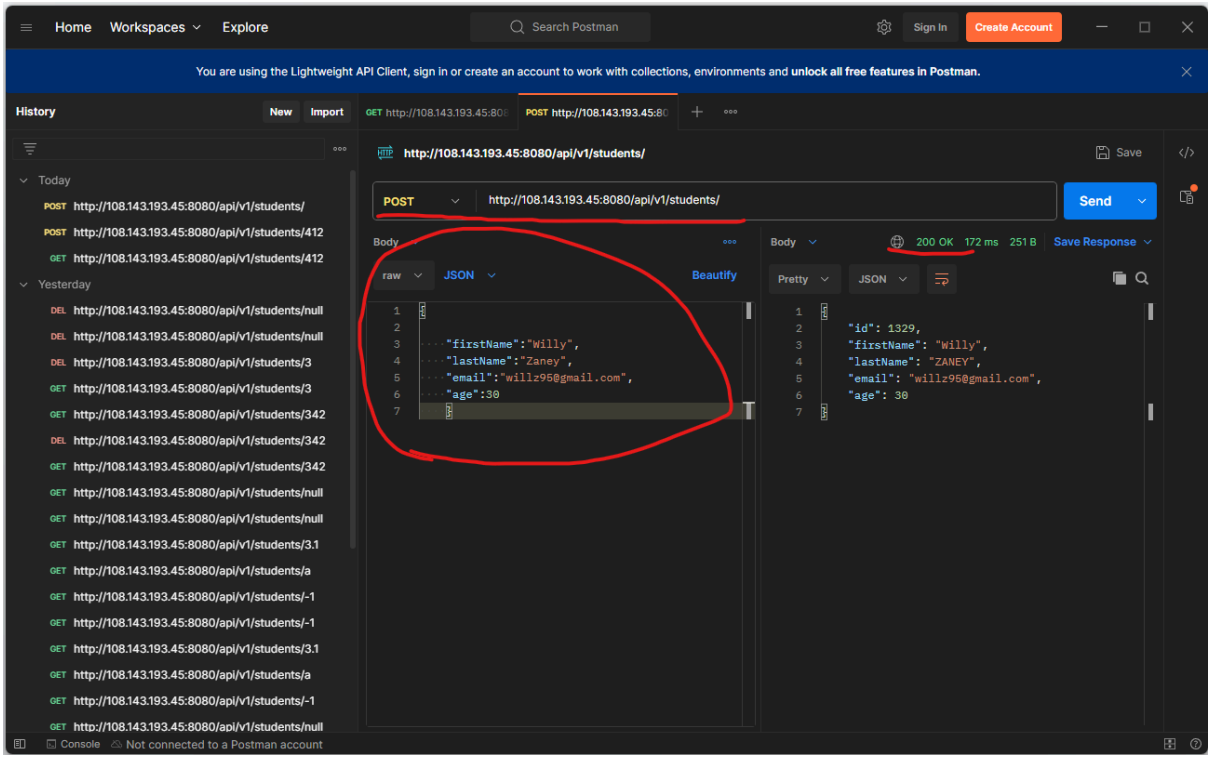
Test #2 Result: Fail (check image below)

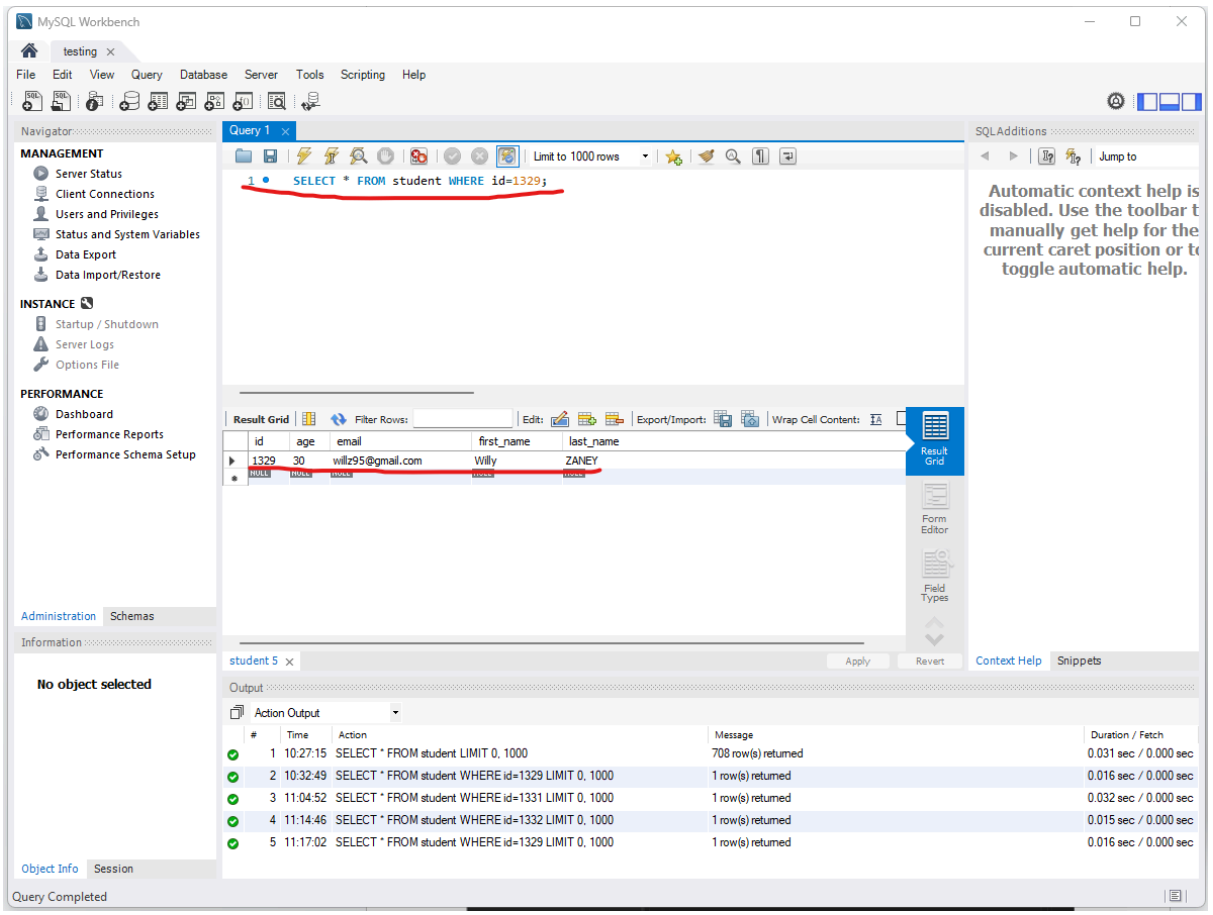


Test #5 Result: Fail (check image below)

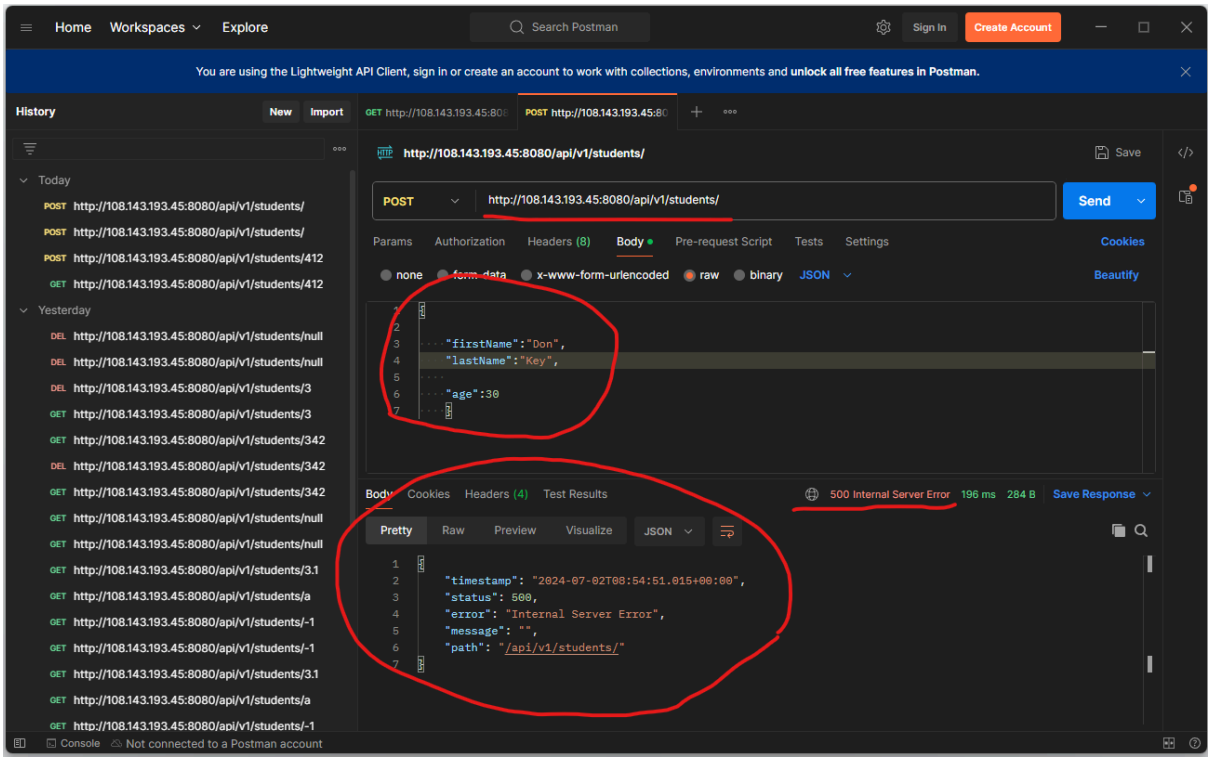


Test #7 Result: Fail (check images below)

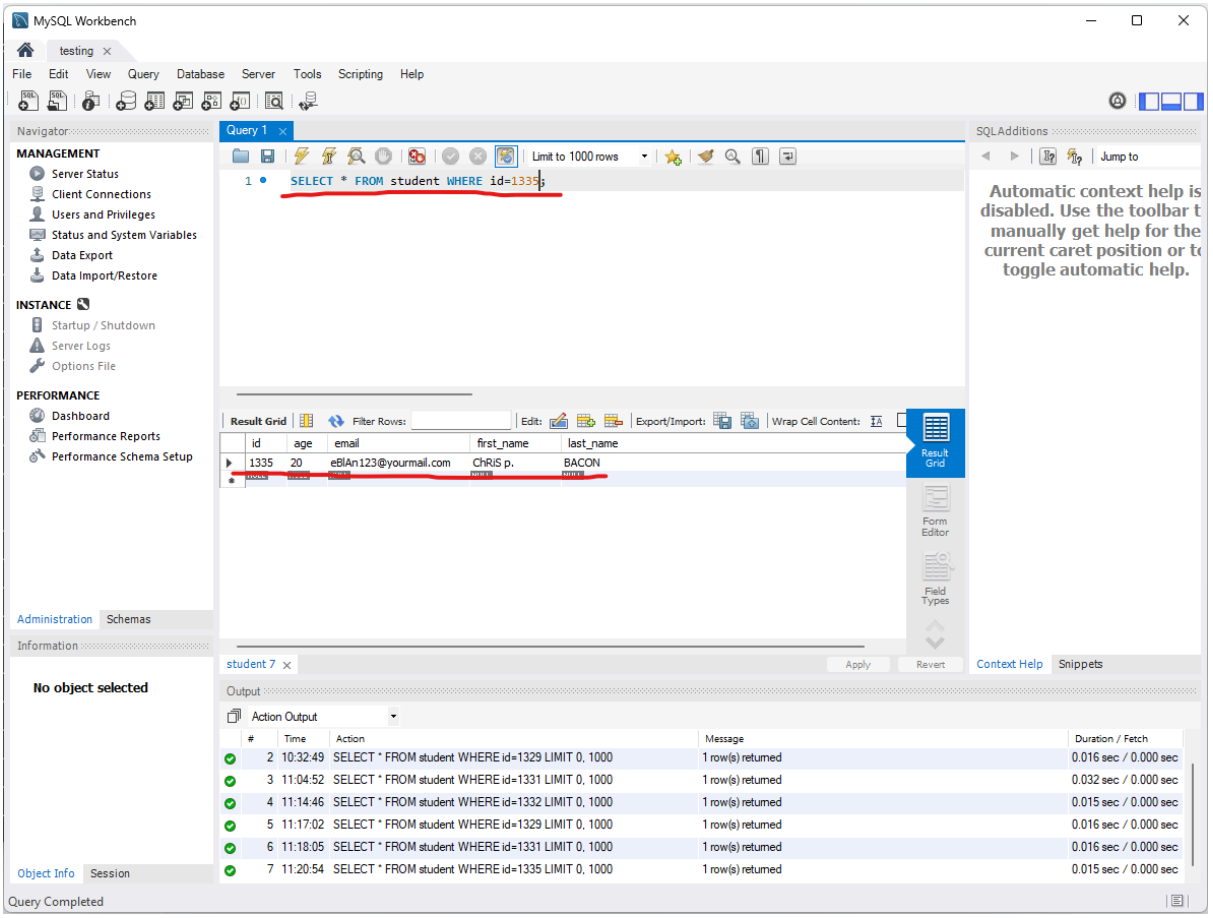


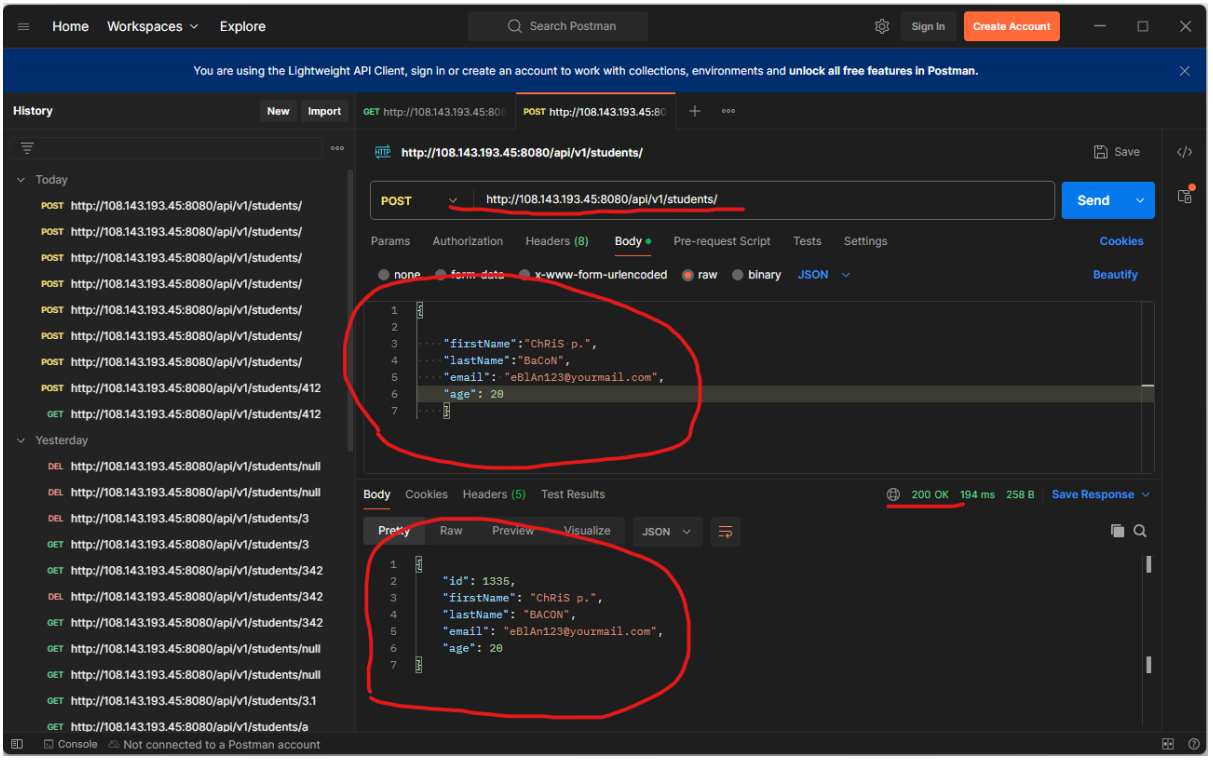


Test #8 Result: Fail (check image below)

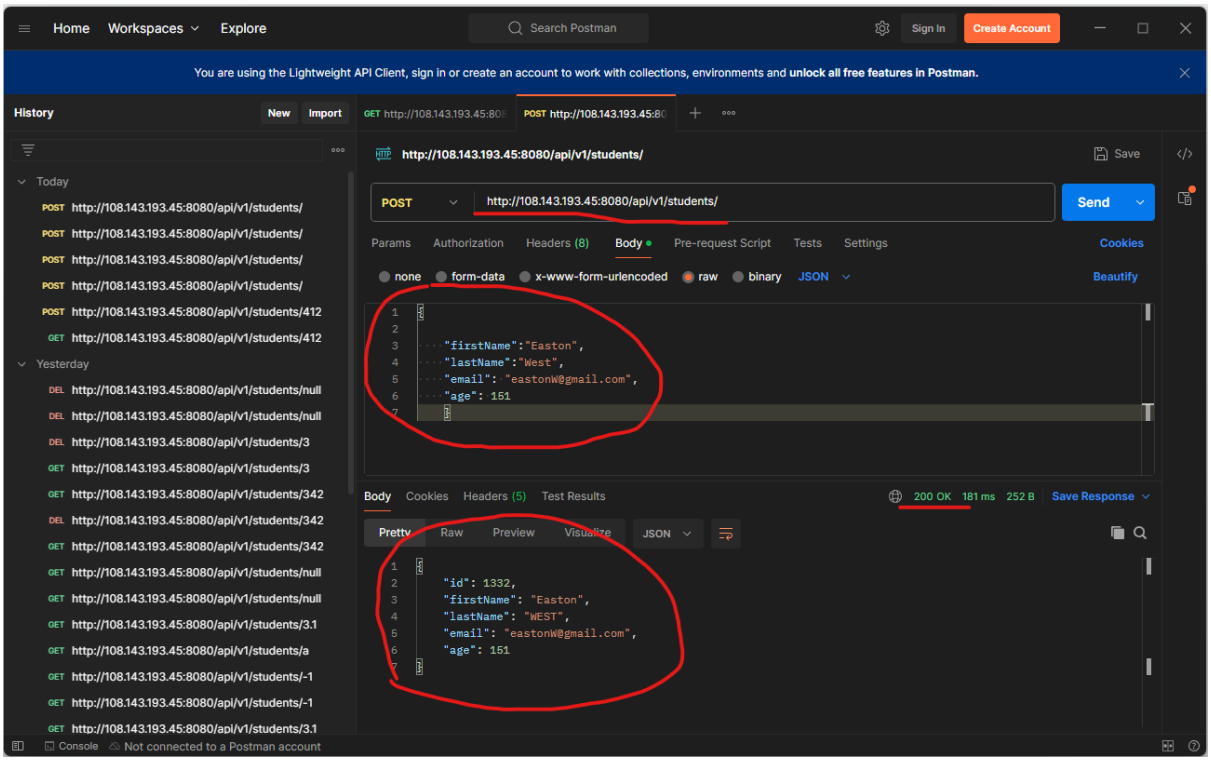


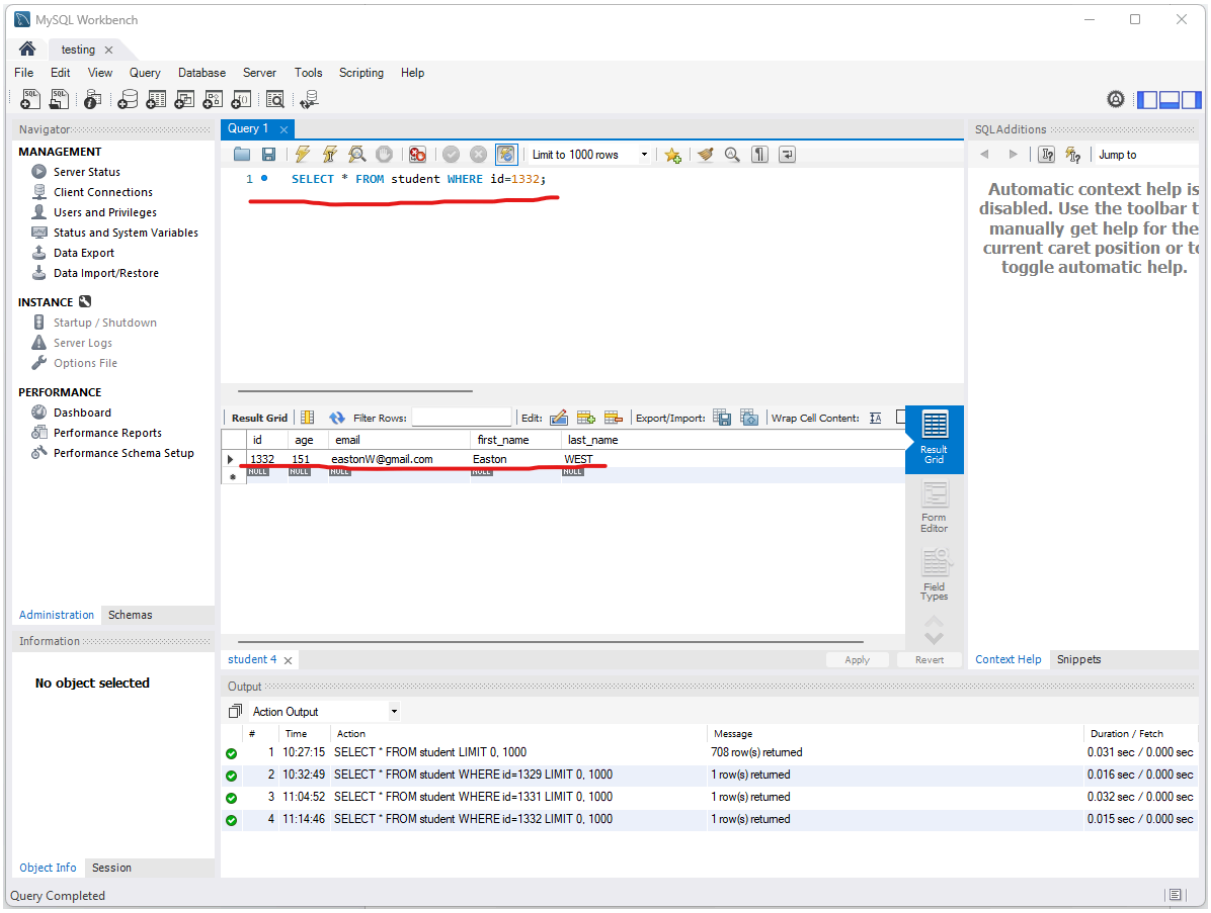
Test #9 Result: Fail (check images below)





Test #10 Result: Fail (check images below)





Test #11 Result: Fail (check images below)