FINÁLNÍ PROJEKT   
 č.1 MMK



Autor: Milan Mike Klos

Datum: 8.8.2024

OBSAH

[**ZADÁNÍ 3**](#_30j0zll)

[**TESTOVACÍ SCÉNÁŘE 4**](#_2et92p0)

[**EXEKUCE TESTŮ 5**](#_tyjcwt)

[**BUG REPORT 5**](#_3dy6vkm)

# ZADÁNÍ

Cílem 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..

## Přístupové údaje:

| Databáze | database: *NULL* Host: *NULL* Username: *NULL*  Password: *NULL* |
| --- | --- |
| REST-API | *NULL* |

## 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.

:

# TEST CASES

**1. Test Case: GET method in Postman**

***You will need MySQL Workbench DB (from Task table above) and Postman for API validation***

**Steps to reproduce:**

* in Postman set **GET** method and use API [**http://108.143.193.45:8080/api/v1/students/296**](http://108.143.193.45:8080/api/v1/students/296)
* **click Send button**

**Expected result:**

Postman values will match values from DB

status: 200 OK

"id": 296,

"firstName": "Karol",

"lastName": "UNTERWASSERMAN",

"email": "karolko@gmail.com",

"age": 0

**2. Test Case: GET method in Postman - status 500**

***You will need MySQL Workbench DB (from Task table above) and Postman for API validation***

**Steps to reproduce:**

* in Postman set **GET** method and use API [**http://108.143.193.45:8080/api/v1/students/280**](http://108.143.193.45:8080/api/v1/students/280)
* **click Send button**

**Expected result:**

There is no record with given ID in DB, so postman should return something like *“This student doesn’t exist”*

**3. Test Case: POST method in Postman**

***You will need MySQL Workbench DB (from Task table above) and Postman for API validation***

**Steps to reproduce:**

* in Postman select **POST method** and use API [**http://108.143.193.45:8080/api/v1/students**](http://108.143.193.45:8080/api/v1/students)
* in Body select **“raw”** from dropdown and then select **JSON**
* create new user in JSON format, for example:

{

"firstName": "Pepe",

"lastName": "Zdepe",

"email": "mail@mail.cz",

"age": 54

}

* Click Send button

**Expected result:**

New user with our parameters is created in DB

**4.** **Test Case: DELETE method in Postman**

***You will need MySQL Workbench DB (from Task table above) and Postman for API validation***

**Steps to reproduce:**

* in Postman select **DELETE method** and use API [**http://108.143.193.45:8080/api/v1/students**](http://108.143.193.45:8080/api/v1/students)**/1606**
* **Click Send button**

**Expected result:**

User with ID 1606 is removed from the DB

**5.** **Test Case: DELETE method in Postman - status 500**

***You will need MySQL Workbench DB (from Task table above) and Postman for API validation***

**Steps to reproduce:**

* in Postman select **DELETE method** and use API [**http://108.143.193.45:8080/api/v1/students**](http://108.143.193.45:8080/api/v1/students)**/1606 or already removed / non existing ID**
* **Click Send button**

**Expected result:**

There is no record with given ID in DB, so postman should return something like *“This student doesn’t exist”*

# TEST EXECUTIONS

1. **Test Case: GET method in Postman**

***You will need MySQL Workbench DB (from Task table above) and Postman for API validation***

**Steps to reproduce:**

* in Postman set **GET** method and use API [**http://108.143.193.45:8080/api/v1/students/296**](http://108.143.193.45:8080/api/v1/students/296)
* **click Send button**

**Expected result:**

Postman values will match values from DB

status: 200 OK

"id": 296,

"firstName": "Karol",

"lastName": "UNTERWASSERMAN",

"email": "karolko@gmail.com",

"age": 0

**Actual result:**

Postman values match with DB values

**Test Result: Pass**

# 

# 

1. **Test Case: GET method in Postman - status 500**

***You will need MySQL Workbench DB (from Task table above) and Postman for API validation***

**Steps to reproduce:**

* in Postman set **GET** method and use API [**http://108.143.193.45:8080/api/v1/students/280**](http://108.143.193.45:8080/api/v1/students/280)
* **click Send button**

**Expected result:**

There is no record with given ID in DB, so postman should return something like *“This student doesn’t exist”*

**Actual result:**

**"timestamp": "2024-08-20T18:56:53.025+00:00",**

**"status": 500,**

**"error": "Internal Server Error",**

**"message": "",**

**"path": "/api/v1/students/280"**

**Test Result: Failed**

**Created BUG Report (ID: 1, Test Case: GET method in Postman - status 500)**

1. **Test Case: POST method in Postman**

***You will need MySQL Workbench DB (from Task table above) and Postman for API validation***

**Steps to reproduce:**

* in Postman select **POST method** and use API [**http://108.143.193.45:8080/api/v1/students**](http://108.143.193.45:8080/api/v1/students)
* in Body select **“raw”** from dropdown and then select **JSON**
* create new user in JSON format, for example:

{

"firstName": "Pepe",

"lastName": "Zdepe",

"email": "mail@mail.cz",

"age": 54

}

* Click Send button

**Expected result:**

New user with our parameters is created in DB

**Actual result:**

New user with our parameters is created in DB with ID +1 from actual last ID

**{**

**"id": 1606,**

**"firstName": "Pepe",**

**"lastName": "ZDEPE",**

**"email": "mail@mail.cz",**

**"age": 54**

**}**

**Test Result: Pass**

1. **Test Case: DELETE method in Postman**

***You will need MySQL Workbench DB (from Task table above) and Postman for API validation***

**Steps to reproduce:**

* in Postman select **DELETE method** and use API [**http://108.143.193.45:8080/api/v1/students**](http://108.143.193.45:8080/api/v1/students)**/1606**
* **Click Send button**

**Expected result:**

User with ID 1606 is removed from the DB

**Actual result:**

User with ID 1606 is removed from the DB

**Test Result: Pass**

**5.** **Test Case: DELETE method in Postman - status 500**

***You will need MySQL Workbench DB (from Task table above) and Postman for API validation***

**Steps to reproduce:**

* in Postman select **DELETE method** and use API [**http://108.143.193.45:8080/api/v1/students**](http://108.143.193.45:8080/api/v1/students)**/1606 or already removed / non existing ID**
* **Click Send button**

**Expected result:**

There is no record with given ID in DB, so postman should return something like *“This student doesn’t exist”*

**Actual result:**

{

"timestamp": "2024-08-20T19:43:20.706+00:00",

"status": 500,

"error": "Internal Server Error",

"message": "",

"path": "/api/v1/students/1607"

}

**Test Result: Failed**

**Created BUG Report (ID: 2, Test Case: DELETE method in Postman - status 500)**

# BUG REPORT

**Test Case: GET method in Postman - status 500**

**ID: 1**

**Steps to reproduce:**

* in Postman set **GET** method and use API [**http://108.143.193.45:8080/api/v1/students/280**](http://108.143.193.45:8080/api/v1/students/280)
* **click Send button**

**Expected result:**

There is no record with given ID in DB, so postman should return something like *“This student doesn’t exist”*

**Actual result:**

**"timestamp": "2024-08-20T18:56:53.025+00:00",**

**"status": 500,**

**"error": "Internal Server Error",**

**"message": "",**

**"path": "/api/v1/students/280"**

**Test Case: DELETE method in Postman - status 500**

**ID: 2**

**Steps to reproduce:**

* in Postman select **DELETE method** and use API [**http://108.143.193.45:8080/api/v1/students**](http://108.143.193.45:8080/api/v1/students)**/1606 or already removed / non existing ID**
* **Click Send button**

**Expected result:**

There is no record with given ID in DB, so postman should return something like *“This student doesn’t exist”*

**Actual result:**

{

"timestamp": "2024-08-20T19:43:20.706+00:00",

"status": 500,

"error": "Internal Server Error",

"message": "",

"path": "/api/v1/students/1607"

}