

FINAL PROJECT

n.1



ENGETO

Author: Daniel Pecka
Date: 10.11.2024

LIST OF CONTAINS

PROJECT REQUIREMENTS	3
Access credentials:	3
Comments:	3
TEST CASES (TC)	4
TC 1: Retrieve Student Data Using Path Parameter (Standard RESTful)	4
TC 2: Retrieve Student Data Using Query Parameter (Potential Bug Check)	5
TC 3: Handle Invalid Student Data (GET Method for Non-Existent Student)	6
TC 4: Create a New Student Record (POST Method with Valid Data)	7
TC 5: Create Student with Invalid Data (POST Method with Long Strings)	9
TC 6: Create Student with Invalid Data (POST Method with invalid email adress)	10
TC 7: Delete Existing Student (DELETE Method)	13
TESTS EXECUTION	14
TC 1: Retrieve Student Data Using Path Parameter (Standard RESTful)	14
TC 2: Retrieve Student Data Using Query Parameter (Potential Bug Check)	16
TC 3: Handle Invalid Student Data (GET Method for Non-Existent Student)	18
TC 4: Create a New Student Record (POST Method with Valid Data)	19
TC 5: Create Student with Invalid Data (POST Method with Long Strings)	21
TC 6: Create Student with Invalid Data (POST Method with invalid email adress)	23
TC 7: Delete Existing Student (DELETE Method)	25
TESTS OVERVIEW	27
BUG REPORT	32

PROJECT REQUIREMENTS

The goal of the final project is to test the functionality of the application, which is used to manipulate data about students. The application has a REST-API interface that allows creating, deleting and retrieving data..

Access credentials:

Database	database: qa_demo Host: aws.connect.psdb.cloud Username: k9ik3tg4x7hdxk68nlqy Password: pscale_pw_8Nc3bfP1t3ECEJm8cGPze47M egBUWQzgH7J8XsLghBi
REST-API	http://108.143.193.45:8080/api/v1/students/

Comments:

Remember that in IT, data has to be stored somewhere and then retrieved. Therefore, verify that the data is correctly stored and retrieved from the database.

Don't forget to include test data, expected results including response body and status codes in test scenarios.

TEST CASES (TC)

I have tested the functionality of the application based on the listed test cases.

TC 1: Retrieve Student Data Using Path Parameter (Standard RESTful)

Title: Verify GET request retrieves student data correctly using path parameter.

Precondition: Student with ID 564 exists in the database.

Steps:

1. Open a workspace on Postman (<https://web.postman.co/>).
2. Insert a GET request <http://108.143.193.45:8080/api/v1/students/564> and send it.
3. Verify the response status code.
4. Open MySQL Workbench.
5. Insert the following SQL query:

```
SELECT *  
FROM student  
WHERE id = 564  
;
```

6. Compare the output of the GET request with output of the SQL query.

Expected Result:

- Status code: 200 OK
- Postman response body:

```
{  
  "id": 564,  
  "firstName": "Petra",  
  "lastName": "RADILOVA",  
  "email": "radilova@yourmail.com",  
  "age": 19  
}
```

- Database output matches the Postman response.

TC 2: Retrieve Student Data Using Query Parameter (Potential Bug Check)

Title: Verify GET request retrieves data correctly using query parameter (id=564).

Precondition: Student with ID 564 exists in the database, while the parameter search being available is an intended design.

Steps:

1. Open a workspace on Postman (<https://web.postman.co/>).
2. Insert a GET request <http://108.143.193.45:8080/api/v1/students?id=564> and send it.
3. Verify the response status code.
4. Open MySQL Workbench.
5. Insert the following SQL query:

```
SELECT *  
FROM student  
WHERE id = 564  
;
```

6. Verify the response status.
7. Compare the output of the GET request with output of the SQL query.

Expected Result:

- Status code: *200 OK*
- Postman response body:

```
{  
  "id": 564,  
  "firstName": "Petra",  
  "lastName": "RADILOVA",  
  "email": "radilova@yourmail.com",  
  "age": 19  
}
```

- Database output matches the Postman response.

TC 3: Handle Invalid Student Data (GET Method for Non-Existent Student)

Title: Verify GET request for a non-existent student returns a 404 Not Found status using path parameter.

Precondition: No student exists with ID 9999. There is a pre-designed error message.

Steps:

1. Open a workspace on Postman (<https://web.postman.co/>).
2. Insert a GET request <http://108.143.193.45:8080/api/v1/students/9999> and send it.
3. Verify the response status code and body.

Expected Result:

- Status code: 404 Not Found
- Response body:

```
{  
  "error": "Student not found"  
}
```

TC 4: Create a New Student Record (POST Method with Valid Data)

Title: Verify POST request creates a new student record successfully.

Precondition: Ensure a student with ID 1000 does not exist.

Steps:

1. Open a workspace on Postman (<https://web.postman.co/>).
2. Insert a POST request to <http://108.143.193.45:8080/api/v1/students/>.
3. Set the request body as:

```
{  
  "id": 1000,  
  "firstName": "Petr",  
  "lastName": "Novák",  
  "email": "petr.novak@example.com",  
  "age": 30  
}
```

4. Send the request.
5. Verify the response code and body.
6. Check the database to ensure the new record was added:

```
SELECT *  
FROM student  
WHERE id = 1000  
;
```

Expected Result:

- Status code: *200 OK*
- Postman response body:

```
{  
  "id": 1000,  
  "firstName": "Petr",
```

```
"lastName": "Novák",  
"email": "petr.novak@example.com",  
"age": 30  
}
```

- Database output matches the Postman response.

TC 5: Create Student with Invalid Data (POST Method with Long Strings)

Title: Verify POST request fails with long string inputs.

Precondition: Ensure API is supposed to have validation for input data length according documentation. There is a pre-designed error message. No student exists with ID 1002.

Steps:

Steps:

1. Open a workspace on Postman (<https://web.postman.co/>).
2. Insert a POST request to <http://108.143.193.45:8080/api/v1/students/>.
3. Set the Pre-request script as:

```
const longString = 'A'.repeat(300); pm.environment.set("longString", longString);
```

4. Set the request body as:

```
{
  "id": 1005,
  "firstName": "{{longString}}",
  "lastName": "{{longString}}",
  "email": "mail@mail.com",
  "age": 20
}
```

5. Send the request.
6. Verify the response code and body.
7. Check the database to ensure no record was added:

```
SELECT *
FROM student
WHERE id = 1002
;
```

Expected Result:

- Status code: *400 Bad Request*
- Postman response body:

```
{  
  "error": "Input data too long or invalid format"  
}
```

- There is no record with ID 1002 in the database..

TC 6: Create Student with Invalid Data (POST Method with invalid email adress)

Title: Verify API handles invalid email format correctly.

Precondition: Ensure API should have validation for email input according to documentation. There is a pre-designed error message. No student exists with ID 1005.

Steps:

1. Open a workspace on Postman (<https://web.postman.co/>).
2. Insert a POST request to <http://108.143.193.45:8080/api/v1/students/>.
3. Set the request body as:

```
{  
  "id": 1005,  
  "firstName": "Petr",  
  "lastName": "Novák",  
  "email": "petrnovakexample.com",  
  "age": 30  
}
```

4. Send the request.
5. Verify the response code and body.
6. Check the database to ensure no record was added:

```
SELECT *  
FROM student  
WHERE id = 1005  
;
```

Expected Result:

- Status code: *400 Bad Request*
- Postman response body:

```
{
```

```
"error": "Invalid email format"  
}
```

- There is no record with ID 1002 in the database..

TC 7: Delete Existing Student (DELETE Method)

Title: Verify DELETE request removes a student record.

Precondition: Student exists with ID 2241 in the database.

Steps:

1. Open a workspace on Postman (<https://web.postman.co/>).
2. Insert a POST request to <http://108.143.193.45:8080/api/v1/students/2241>.
3. Send the request.
4. Verify the response code.
5. Check the database to ensure the record was deleted:

```
SELECT *  
FROM student  
WHERE id = 2241  
;
```

Expected Result:

- Status code: *200 OK*
- There is no record with ID 2241 in the database..

TESTS EXECUTION

I have completed the test cases, I list the test results in this section.

TC 1: Retrieve Student Data Using Path Parameter (Standard RESTful)

Following the scenario in the previous section I have achieved the results in the supplements below:

Supplement 1.1: Postman results

The screenshot displays the Postman application interface. At the top, the request method is set to **GET** and the URL is `http://108.143.193.45:8080/api/v1/students/564`. The **Params** tab is selected, showing a table for Query Parameters.

Key	Value	Description
Key	Value	Description

Below the table, the **Body** tab is active, showing the response in **JSON** format. The status is **200 OK** with a response time of 650 ms and a size of 257 B. The response body is a JSON object:

```
1 {
2   "id": 564,
3   "firstName": "Petra",
4   "lastName": "RADILOVA",
5   "email": "radilova@yourmail.com",
6   "age": 19
7 }
```

Supplement 1.2: Database results

The screenshot shows the MySQL Workbench interface. On the left, the 'Navigator' pane displays the 'qa_demo' database schema, including tables like 'hibernate_sequence' and 'student', and columns like 'id', 'age', 'email', 'first_name', and 'last_name'. The 'Query 1' editor in the center contains the following SQL query:

```
1 SELECT *
2 FROM student
3 WHERE id = 564
4 ;
5
```

Below the query editor, the 'Result Grid' shows the query results in a table format:

id	age	email	first_name	last_name
564	19	radlova@yourmail.com	Petra	RADIOLOVA

On the right, the 'SQLAdditions' pane displays a message: 'Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.'

At the bottom, the 'Output' pane shows the execution details for the query:

#	Time	Action	Message	Duration / Fetch
1	13:45:32	SELECT * FROM student WHERE id = 564 LIMIT 0, 1000	1 row(s) returned	0.016 sec / 0.000 sec

TC 2: Retrieve Student Data Using Query Parameter (Potential Bug Check)

Following the scenario in the previous section I have achieved the results in the supplements below:

Supplement 2.1: Postman results

GET http://108.143.193.45:8080

HTTP

http://108.143.193.45:8080/api/v1/students?id=564

Save

Share

GET

http://108.143.193.45:8080/api/v1/students?id=564

Send

Params

Authorization

Headers (5)

Body

Scripts

Tests

Settings

Cookies

Query Params

<input checked="" type="checkbox"/>	Key	Value	Description	...	Bulk Edit
<input checked="" type="checkbox"/>	id	564			
	Key	Value	Description		

Body

Cookies

Headers (5)

Test Results

200 OK

771 ms

105.22 KB

Pretty

Raw

Preview

Visualize

JSON

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

[

{

"id": 350,

"firstName": "Jo",

"lastName": "DOE",

"email": "john.doe@gmail.com",

"age": 34

}

,

{

"id": 351,

"firstName": "al",

"lastName": "HITCHCOCK",

"email": "alfred@mail.com",

"age": 80

}

,

{

"id": 352,

"firstName": "John",

"lastName": "DOE",

"email": "john.doe@gmail.com",

"age": 17

}

,

{

"id": 354,

"firstName": "alfred",

Supplement 2.2: Database results

The screenshot displays the MySQL Workbench interface. On the left, the 'Navigator' pane shows the 'qa_demo' database schema with tables like 'hibernate_sequence' and 'student'. The 'student' table is expanded, showing columns: id, age, email, first_name, and last_name. The main query editor shows a SQL query: `SELECT * FROM student WHERE id = 564`. The 'Result Grid' pane displays the query results for the student with id 564.

id	age	email	first_name	last_name
564	19	radiova@yourmail.com	Petra	RADIOLOVA

On the right, the 'SQLAdditions' pane contains a message: 'Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.'

At the bottom, the 'Output' pane shows the 'Action Output' for the query execution:

#	Time	Action	Message	Duration / Fetch
1	13:45:32	SELECT * FROM student WHERE id = 564 LIMIT 0, 1000	1 row(s) returned	0.016 sec / 0.000 sec

TC 3: Handle Invalid Student Data (GET Method for Non-Existent Student)

Following the scenario in the previous section I have achieved the result in the supplement below:

Supplement 3.1: Postman results

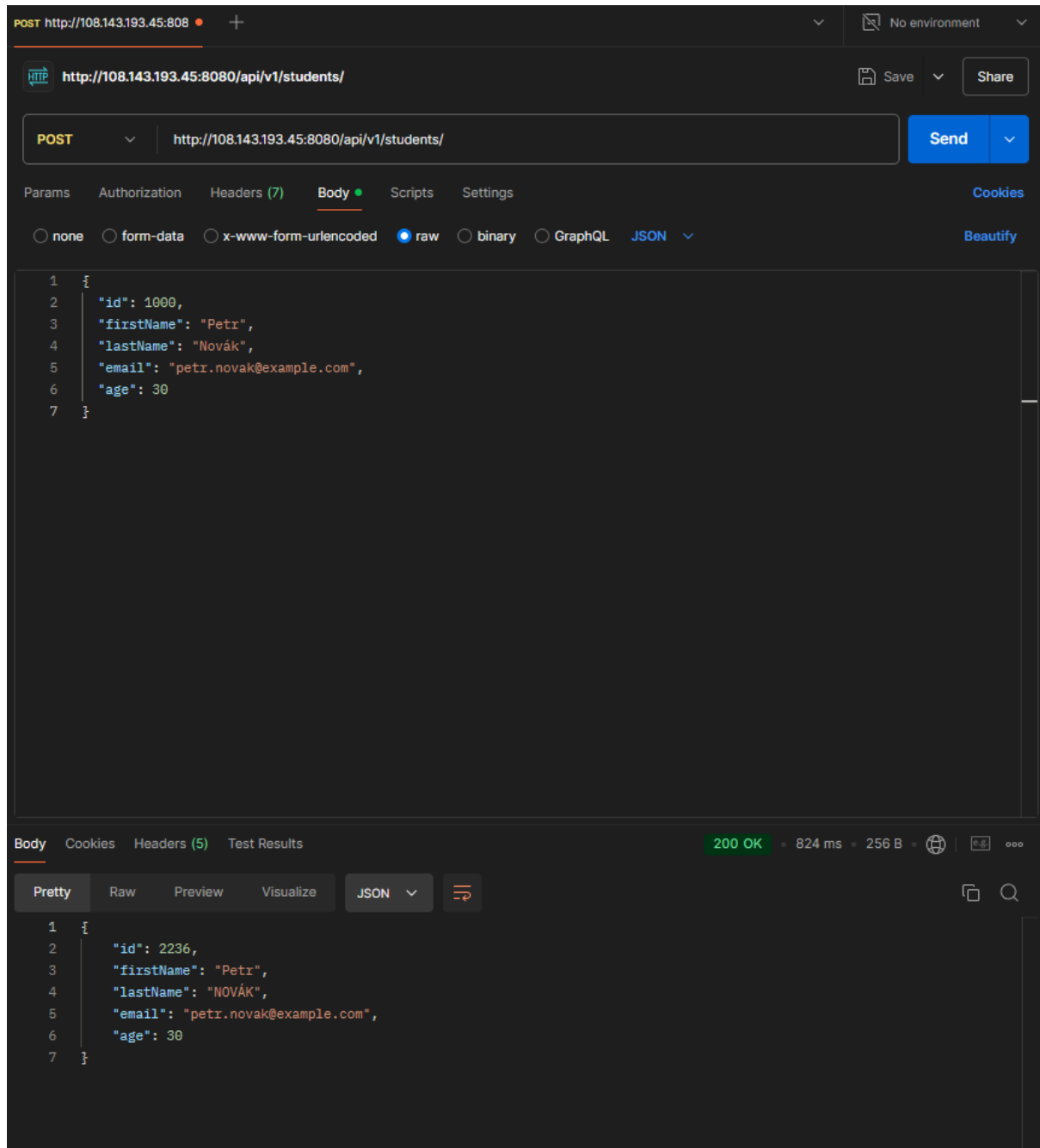
The screenshot shows a Postman interface with a GET request to `http://108.143.193.45:8080/api/v1/students/9999`. The request is saved and ready to be sent. The response is a 500 Internal Server Error, which is displayed in the Body tab. The error message is as follows:

```
1 {
2   "timestamp": "2024-11-10T10:33:11.851+00:00",
3   "status": 500,
4   "error": "Internal Server Error",
5   "message": "",
6   "path": "/api/v1/students/9999"
7 }
```

TC 4: Create a New Student Record (POST Method with Valid Data)

Following the scenario in the previous section I have achieved the result in the supplement below:

Supplement 4.1: Postman results



Supplement 4.2: Database result for id = 1000

The screenshot shows the MySQL Workbench interface with the 'qa_demo' database selected. The 'student' table is expanded in the Navigator. The SQL editor contains the following query:

```
1 SELECT *
2 FROM student
3 WHERE id = 1000
4 ;
```

The 'Result Grid' is empty, indicating no rows were returned. The 'Output' pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	12:10:08	SELECT * FROM student WHERE id = 1000 LIMIT 0, 1000	0 row(s) returned	0.016 sec / 0.000 sec

On the right, a message states: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

Supplement 4.2: Database result for id = 2236

The screenshot shows the MySQL Workbench interface with the 'qa_demo' database selected. The 'student' table is expanded in the Navigator. The SQL editor contains the following query:

```
1 SELECT *
2 FROM student
3 WHERE id = 2236
4 ;
```

The 'Result Grid' displays one row of data:

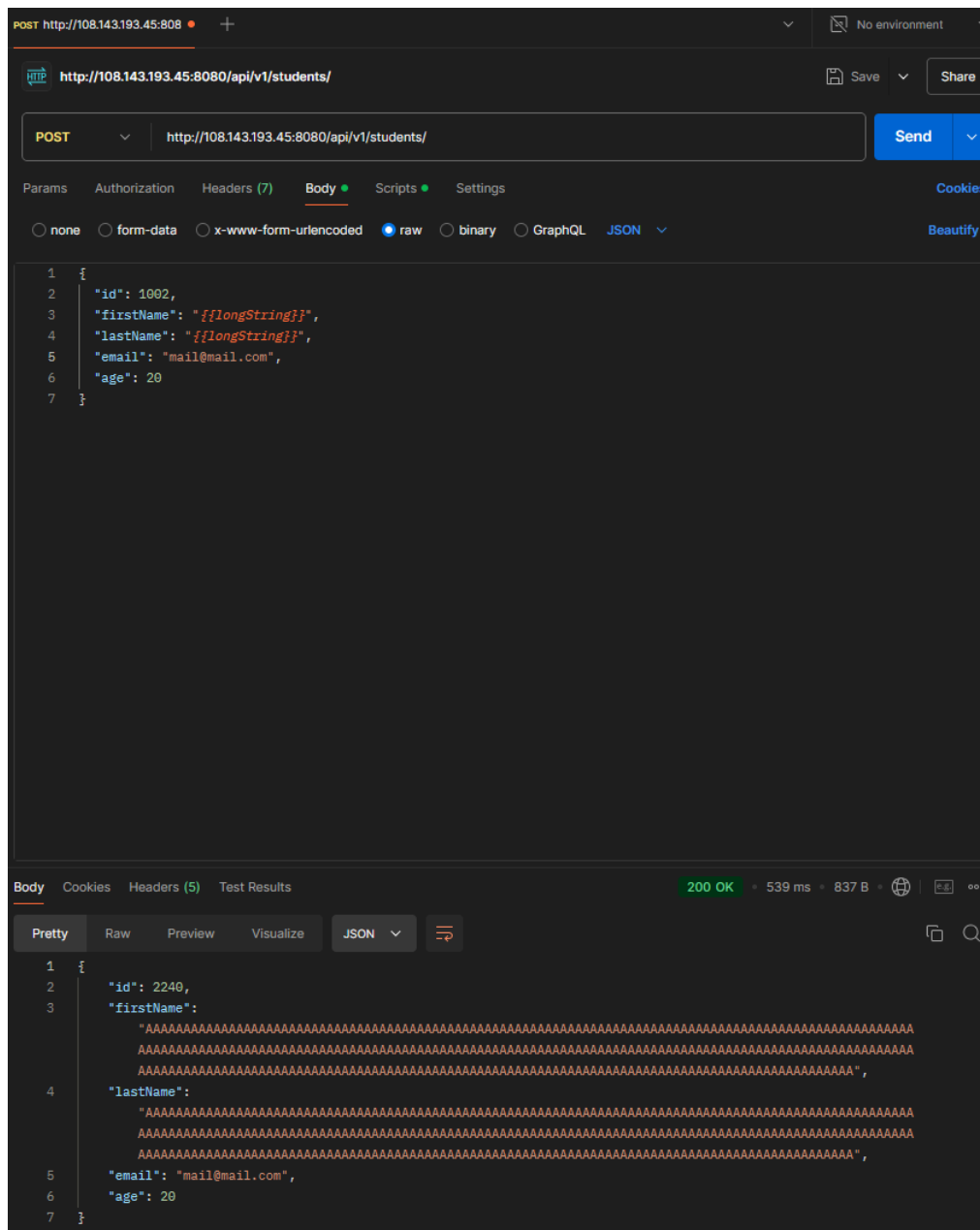
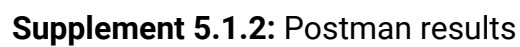
id	age	email	first_name	last_name
2236	30	petr.novak@example.com	Petr	NOVÁK

The 'Output' pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	12:10:08	SELECT * FROM student WHERE id = 1000 LIMIT 0, 1000	0 row(s) returned	0.016 sec / 0.000 sec
2	12:10:45	SELECT * FROM student WHERE id = 2236 LIMIT 0, 1000	1 row(s) returned	0.015 sec / 0.000 sec

On the right, a message states: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

Supplement 5.1.1: Postman Pre-request



Supplement 5.2: DB output

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

qa_demo

Tables

hibernate_sequence

student

Columns

id

age

email

first_name

last_name

Indexes

PRIMARY

Foreign Keys

Triggers

Views

Stored Procedures

Functions

sys

Query 1

```
1 SELECT *
2 FROM student
3 WHERE id = 2240
4 ;
```

Limit to 1000 rows

SQLAdditions

Jump to

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid

id	age	email	first_name	last_name
2240	20	mail@mail.com	AAAAAAAAAAAAAAAAAAAAAAAAAAAA...	AAAAAAAAAAAAAAAAAAAAAAAAAAAA...

student 13

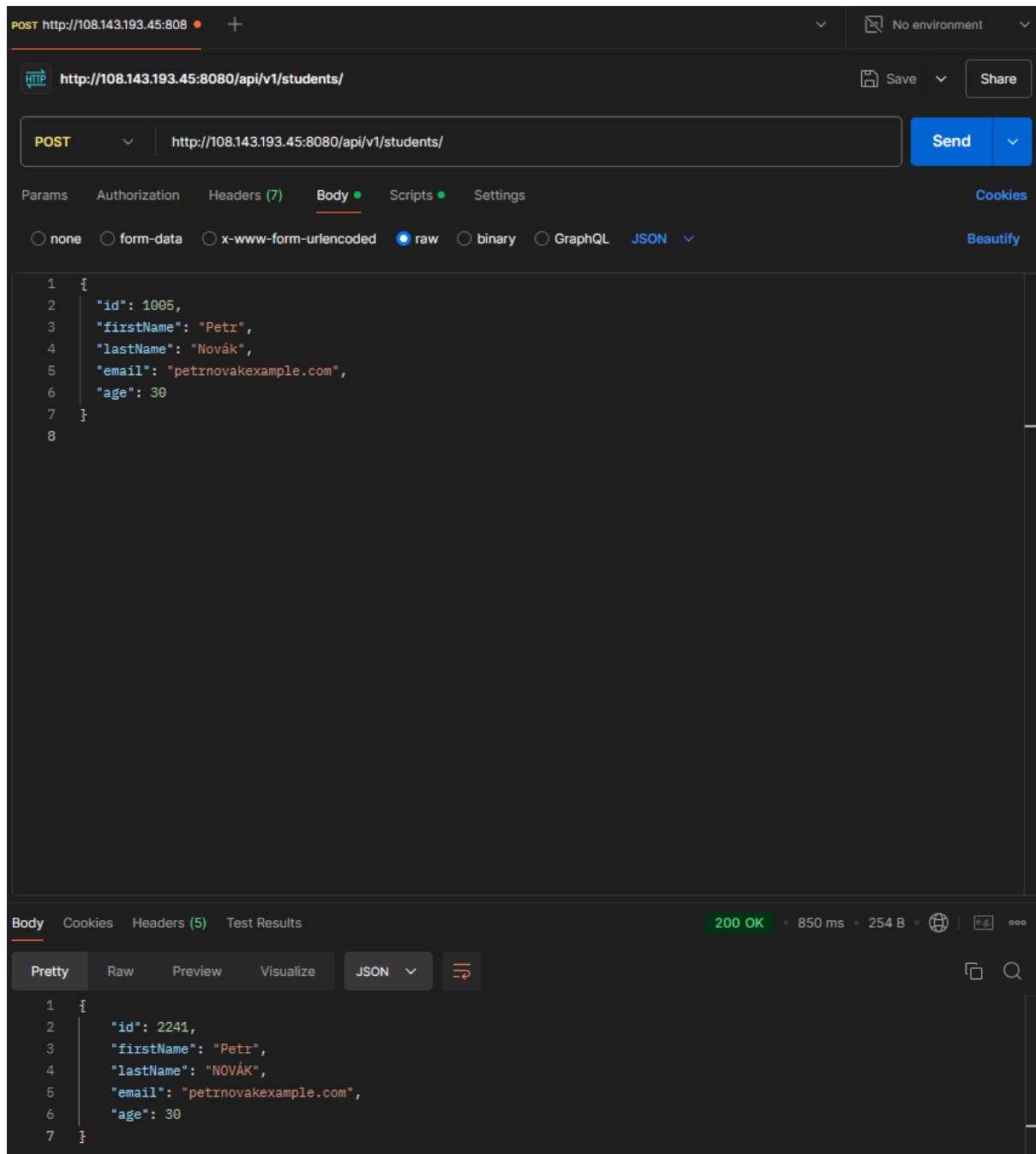
Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	14:45:27	SELECT * FROM student WHERE id = 2240 LIMIT 0, 1000	1 row(s) returned	0.031 sec / 0.000 sec

TC 6: Create Student with Invalid Data (POST Method with invalid email adress)

Supplement 6.1: Postman results



Supplement 6.2: DB output

MySQL Workbench

qa_demo

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

qa_demo

Tables

hibernate_sequence

student

Columns

id

age

email

first_name

last_name

Indexes

PRIMARY

Foreign Keys

Triggers

Views

Stored Procedures

Functions

sys

Query 1

Limit to 1000 rows

1 SELECT *

2 FROM student

3 WHERE id = 2241

4 ;

SQLAdditions

Jump to

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid

Filter Rows:

Exports

Wrap Cell Contents

id age email first_name last_name

2241 30 petnovakexample.com Petr NOVÁK

student 14

Read Only

Context Help

Snippets

Output

Action Output

Time Action Message Duration / Fetch

1 14:50:09 SELECT * FROM student WHERE id = 2241 LIMIT 0, 1000 1 row(s) returned 0.015 sec / 0.016 sec

TC 7: Delete Existing Student (DELETE Method)

Supplement 7.1: Postman results

The image shows the Postman application interface. At the top, the URL bar displays 'DEL http://108.143.193.45:8080' with a dropdown arrow and a 'No environment' label. Below this, the request details are shown: 'HTTP' icon, 'http://108.143.193.45:8080/api/v1/students/2241', 'Save' button, and 'Share' button. The method is set to 'DELETE' and the URL is 'http://108.143.193.45:8080/api/v1/students/2241'. A 'Send' button is visible. The 'Body' tab is selected, showing 'raw' as the format. The body content is empty. The status bar at the bottom indicates a '200 OK' response with a time of '1091 ms' and a size of '123 B'. The 'Body' tab is also selected in the bottom panel, showing 'Pretty' as the format. The response content is empty.

DEL http://108.143.193.45:8080 +

HTTP http://108.143.193.45:8080/api/v1/students/2241 Save Share

DELETE http://108.143.193.45:8080/api/v1/students/2241 Send

Params Authorization Headers (7) Body Scripts Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON Beautify

1
2

Body Cookies Headers (4) Test Results 200 OK • 1091 ms • 123 B

Pretty Raw Preview Visualize Text 1

Supplement 7.2: DB output

MySQL Workbench

qa_demo

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

qa_demo

Tables

hibernate_sequence

student

Columns

id

age

email

first_name

last_name

Indexes

PRIMARY

Foreign Keys

Triggers

Views

Stored Procedures

Functions

sys

Query 1

Limit to 1000 rows

1 SELECT *

2 FROM student

3 WHERE id = 2241

4 ;

SQLAdditions

Jump to

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid

Filter Rows:

Export: Wrap Cell Content: I

id age email first_name last_name

student 18

Read Only

Context Help

Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	16:21:03	SELECT * FROM student WHERE id = 2241 LIMIT 0, 1000	0 row(s) returned	0.015 sec / 0.000 sec

TESTS OVERVIEW

Below is an overview of all test scenarios I have done, 2 passed and 5 failed. For those failed a bug report with additional information is included in the next section.

Test Case	Scenario	Expected Result	Actual Result	Status	Execution Timestamp	Environment
TC1	Retrieve Student Data Using Path Parameter (Standard RESTful)	200 OK; DB output matches Postman response	200 OK; DB output matches Postman response	PASS	2024-11-09 13:40:26	MySQL Workbench 8.0.32 build 3737333 CE (64-bit); Firefox Browser 132.0.1 (64-bit); Postman for Web Version 11.19.0-241023-0 434; Windows 10 Pro Version 22H2 OS build 19045.4894
TC2	Retrieve Student Data Using Query Parameter (Potential Bug Check)	200 OK; DB output matches Postman response.	200 OK; DB output does not match postman. Postman returns different data	FAIL	2024-11-10 13:45:32	MySQL Workbench 8.0.32 build 3737333 CE (64-bit); Firefox Browser 132.0.1 (64-bit); Postman for Web Version

			than requested: https://drive.google.com/file/d/1X6K0uBi3b0ur39TqIGtpC9AyPwEG4-GA/view?usp=sharing			11.19.0-241023-0434; Windows 10 Pro Version 22H2 OS build 19045.4894
TC3	Handle Invalid Student Data (GET Method for Non-Existent Student)	404 Not Found; Error message is displayed: { "error": "Student not found" }	500 Internal Server Error; Postman output: https://drive.google.com/file/d/18Yv5sleUaQCVdbutHsgCspex6qdYVtrU/view?usp=sharing	FAIL	2024-11-10 10:33:11	Firefox Browser 132.0.1 (64-bit); Postman for Web Version 11.19.0-241023-0434; Windows 10 Pro Version 22H2 OS build 19045.4894
TC4	Create a New Student Record (POST Method with Valid Data)	200 OK; Postman response: { "id": 2236, "firstName":	200 OK; Postman output: https://drive.google.com/file/d/1M3ipnKU3kraMvKR74TpUtqAAIBeiJ2Hf/view?usp=sharing	FAIL	2024-11-10 12:10:45	MySQL Workbench 8.0.32 build 3737333 CE (64-bit); Firefox Browser 132.0.1 (64-bit); Postman for Web Version 11.19.0-241023-0

		"Petr", "lastName": "Novák", "email": "petr.novak@exa mple.com", "age": 30 } DB output matches Postman response	The record is created with a different ID and capitalized surname, can be found in DB but does not fulfill the test requirements.			434; Windows 10 Pro Version 22H2 OS build 19045.4894
TC5	Create Student with Invalid Data (POST Method with Long Strings)	400 Bad Request; Postman response: { "error": "Input data too long or invalid format" }	200 OK; Postman output: https://drive.google.com/file/d/1-JjPCvBmRCUt7AJFhCQUXEM2F7mzTE-c/view?usp=sharing The record is created with invalid data.	FAIL	2024-11-10 14:45:27	MySQL Workbench 8.0.32 build 3737333 CE (64-bit); Firefox Browser 132.0.1 (64-bit); Postman for Web Version 11.19.0-241023-0 434; Windows 10 Pro Version 22H2 OS build 19045.4894

		No record is created in the DB.				
TC6	Create Student with Invalid Data (POST Method with invalid email adress)	<p>400 Bad Request;</p> <p>Postman response:</p> <pre>{ "error": "Invalid email format" }</pre> <p>No record is created in the DB.</p>	<p>200 OK;</p> <p>Postman output: https://drive.google.com/file/d/1JpMh76pm1M-SYYBUZh-tVVaaHpiEI4K/view?usp=sharing</p> <p>The record is created with invalid data.</p>	FAIL	2024-11-10 14:50:09	<p>MySQL Workbench 8.0.32 build 3737333 CE (64-bit);</p> <p>Firefox Browser 132.0.1 (64-bit);</p> <p>Postman for Web Version 11.19.0-241023-0434;</p> <p>Windows 10 Pro Version 22H2 OS build 19045.4894</p>
TC7	Delete Existing Student (DELETE Method)	<p>200 OK;</p> <p>DB output will return empty for id 2241.</p>	<p>200 OK;</p> <p>DB output returns empty for id 2241.</p>	PASS	2024-11-10 16:21:03	<p>MySQL Workbench 8.0.32 build 3737333 CE (64-bit);</p> <p>Firefox Browser 132.0.1 (64-bit);</p> <p>Postman for Web Version 11.19.0-241023-0434;</p> <p>Windows 10 Pro Version 22H2 OS</p>

						build 19045.4894
--	--	--	--	--	--	------------------

BUG REPORT

Based on the performed scenarios, I discovered the mentioned application errors. I know it is ideal to include individual steps in the bug report, but this format does not make it very readable if I add them. For this purpose I list the "Related Test Case" so the information can be looked up.

Bug ID			Tracking			Bug Details				Environment			
ID Number	Reported by	Submission date	Assigned to	Severity	Priority	Related Test Case	Scenario	Expected Result	Actual Result	DB Tool	Browser	Postman	Operating system
001	Daniel Pecka	2024-11-10	-	Major	High	TC2	Retrieve Student Data Using Query Parameter (Potential Bug Check)	200 OK; DB output matches Postman response.	200 OK; DB output does not match postman. Postman returns different data than requested: https://drive.google.com/file/d/1X6K0uBi3b0ur39TqIGtpC9AyPwEG4-GA/view?usp=sharing	MySQL Workbench 8.0.32 build 373733 3 CE (64-bit)	Firefox Browser 132.0.1 (64-bit)	Postman for Web Version 11.19.0-241023-0434	Windows 10 Pro Version 22H2 OS build 19045.4894

002	Daniel Pecka	2024-11-10	-	Minor	Low	TC3	Handle Invalid Student Data (GET Method for Non-Existent Student)	404 Not Found; Error message is displayed: { "error": "Student not found" }	500 Internal Server Error; Postman output: https://drive.google.com/file/d/18Yv5sleUaQCVdbutHsgCspex6qdYVtrU/view?usp=sharing	-	Firefox Browser 132.0.1 (64-bit)	Postman for Web Version 11.19.0-241023-0434	Windows 10 Pro Version 22H2 OS build 19045.4894
003	Daniel Pecka	2024-11-10	-	Average	Medium	TC4	Create a New Student Record (POST Method with Valid Data)	200 OK; Postman response: { "id": 2236, "firstName": "Petr", "lastName": "Novák", "email": "petr.novak@example.com", "age": 30 } DB output matches Postman response	200 OK; Postman output: https://drive.google.com/file/d/1M3ipnKU3kraMvKR74TpUtqAAIBeiJ2Hf/view?usp=sharing The record is created with a different ID and capitalized surname, can be found in DB but does not fulfill the test requirements.	MySQL Workbench 8.0.32 build 3737333 CE (64-bit)	Firefox Browser 132.0.1 (64-bit)	Postman for Web Version 11.19.0-241023-0434	Windows 10 Pro Version 22H2 OS build 19045.4894

004	Daniel Pecka	2024-11-10	-	Major	High	TC5	Create Student with Invalid Data (POST Method with Long Strings)	<p>400 Bad Request;</p> <p>Postman response:</p> <pre>{ "error": "Input data too long or invalid format" }</pre> <p>No record is created in the DB.</p>	<p>200 OK;</p> <p>Postman output:</p> <p>https://drive.google.com/file/d/1-JjPCvBmRCUt7AJFhCQUXEM2F7mzTE-c/view?usp=sharing</p> <p>The record is created with invalid data.</p>	MySQL Workbench 8.0.32 build 373733 3 CE (64-bit)	Firefox Browser 132.0.1 (64-bit)	Postman for Web Version 11.19.0-241023-0434	Windows 10 Pro Version 22H2 OS build 19045.4894
005	Daniel Pecka	2024-11-10	-	Major	High	TC6	Create Student with Invalid Data (POST Method with invalid email adress)	<p>400 Bad Request;</p> <p>Postman response:</p> <pre>{ "error": "Invalid email format" }</pre> <p>No record is created in the DB.</p>	<p>200 OK;</p> <p>Postman output:</p> <p>https://drive.google.com/file/d/1JpMh76pm1M-SYYBUZh-tVVaaHpiEII4K/view?usp=sharing</p> <p>The record is created with invalid data.</p>	MySQL Workbench 8.0.32 build 373733 3 CE (64-bit)	Firefox Browser 132.0.1 (64-bit)	Postman for Web Version 11.19.0-241023-0434	Windows 10 Pro Version 22H2 OS build 19045.4894