
FINAL PROJECT 1



ENGETO

TABLE OF CONTENT

CREDENTIALS & ENVIRONMENT	3
TEST SCENARIOS	4
GET METHOD	4
POST METHOD	6
DELETE METHOD	9
TEST EXECUTION	10
GET METHOD	10
POST METHOD	13
DELETE METHOD	18
BUG REPORT	20

CREDENTIALS

DATABASE	qa_demo
HOST	aws.connect.psdb.cloud
USERNAME	*****
PASSWORD	*****
REST API URL	http://108.143.193.45:8080/api/v1/students/

ENVIRONMENT

OS Windows 10 Home

POSTMAN Version 11.9.0, Architecture x64, Platform win32 10.0.19045

MYSQL WORKBENCH Version 8.0.36 build 3737333

TEST SCENARIOS

GET METHOD

PREREQUISITES

- Workbench database set up with these [credentials](#).
- Postman set on HTTP request with this [REST API URL](#).

VALID SCENARIOS

1. GET DATA ABOUT EXISTING STUDENT

STEPS

1. Data preparation in MySQL Workbench:

```
SELECT * FROM student where id=1318;
```

	id	age	email	first_name	last_name
▶	1318	20	EricDark@Mail.Com	Eric	DARK
*	NULL	NULL	NULL	NULL	NULL

2. Select **GET** method in Postman
3. Paste URL to Postman REST API
4. Add ID „1318“ in the end of URL

EXPECTED RESULT

STATUS

200 OK

RESPONSE

```
{
  "id": 1318,
  "firstName": "Eric",
  "lastName": "DARK",
  "email": "EricDark@Mail.Com",
  "age": 20
}
```

2. GET DATA ABOUT ALL EXISTING STUDENTS

STEPS

1. Data preparation in MySQL Workbench:

```
SELECT * FROM student;
```

	id	age	email	first_name	last_name
▶	268	34		John	JOSHUA
	271	34	548	John	JOSHUA
	283	170	polusa@gmail.	PoPolUska	ČRIEVIČKOVÁ
	285	105	pnov@sda.com	KrungThepMahanakhonAmonRattana...	NOVÁK
	287	67	skupina@jedna.cz	Marek	TESTOVÁNÍNENÍŽÁDNÁRAKETOVÁVĚDAŠTEFANE
	288	67	skupina@jednac	LADISLAV	PODMOSTEM
	289	999	skupina7jedna.cz	AI	NEVĚŘÍČÍ
	290	93	amesnardi@lund1.de	A.	MESNARD
	291	11	adelsvidron@mail.com	Adel	SVIDRON
	294	30	4077@mash.com	Bejnamin Franklin	HAWKEYE PIERCE
	295	67	skupina@jedna.cz	AI	NEVĚŘÍČÍ

2. Select **GET** method in Postman
3. Paste URL to Postman REST API

EXPECTED RESULT

STATUS

200 OK

RESPONSE

List of all students in following format:

```
{
  "id": ,
  "firstName": ,
  "lastName": ,
  "email": ,
  "age":
}
```

INVALID SCENARIOS

1. GET DATA ABOUT NON-EXISTING STUDENT (using non-existing ID)

STEPS

1. Data preparation in MySQL Workbench:

```
SELECT * FROM student where id=0000;
```

#	Time	Action	Message
1	16:15:00	SELECT * FROM student where id=0000 LIMIT 0, 1000	0 row(s) returned

2. Select **GET** method in Postman
3. Paste URL to Postman REST API
4. Add ID „0000“ in the end of URL

EXPECTED RESULT

STATUS

404 NOT FOUND

RESPONSE

No data shown in Response tab

2. GET DATA ABOUT NON-EXISTING STUDENT (using invalid input as ID number)

STEPS

1. Data preparation in MySQL Workbench:

```
SELECT * FROM student where id=XXX;
```

#	Time	Action	Message
1	16:29:01	SELECT * FROM student where id=XXX LIMIT 0, 1000	Error Code: 1054, target: qa_demo.-primary:

2. Select **GET** method in Postman
3. Paste URL to Postman REST API
4. Add ID „XXX“ in the end of URL

EXPECTED RESULT

STATUS

400 BAD REQUEST

RESPONSE

No data shown in Response tab

POST METHOD

PREREQUISITES

- Workbench database set up with these [credentials](#).
- Postman set on HTTP request with this [REST API URL](#).

ATTRIBUTES

- **All fields are mandatory.**
- All values are saved as **lowercase**.

VALIDATION CRITERIA

- Value for age is **from 1 to 150**.
- Value for first_name and last_name has **from 3 to 50 characters**.
- Value for email contains at least one „@“ and „.“ character.

VALID SCENARIOS

1. ADD STUDENT TO DATABASE (check status code)

STEPS

1. Select **POST** method in Postman
2. Paste URL to Postman REST API
3. Add data to Request tab:

```
{
  "firstName": "Anna",
  "lastName": "Boyle",
  "email": "annaB@mail.com",
  "age": 30
}
```

EXPECTED RESULT

STATUS

201 CREATED

RESPONSE

Data shown in Response tab; id value to be generated automatically

```
{
  "id": ,
  "firstName": "Anna",
  "lastName": "Boyle",
  "email": "annaB@mail.com",
  "age": 30
}
```

2. CHECK IF DATABASE SAVES VALUES IN LOWERCASE

STEPS

1. Select **POST** method in Postman
2. Paste URL to Postman REST API
3. Add data to Request tab:

```
{
  "firstName": "Lisa",
  "lastName": "SMITH",
  "email": "LisaSMITH@mail.com",
  "age": 29
}
```

EXPECTED RESULT

STATUS

201 CREATED

RESPONSE

Data shown in Response tab; id value to be generated automatically

```
{
  "id": ,
  "firstName": "lisa",
  "lastName": "smith",
  "email": "lisasmith@mail.com",
  "age": 29
}
```

3. CHECK IF DATABASE SAVES VALUES CORRECTLY (in different language)

STEPS

1. Select **POST** method in Postman
2. Paste URL to Postman REST API
3. Add data to Request tab:

```
{
  "firstName": "Иванов",
  "lastName": "Иванов",
  "email": "Иванов@Иванов.ком",
  "age": 20
}
```

EXPECTED RESULT

STATUS

201 CREATED

RESPONSE

Data shown in Response tab; id value to be generated automatically

```
{
  "id": ,
  "firstName": "Иванов",
  "lastName": "Иванов",
  "email": "Иванов@Иванов.ком",
  "age": 20
}
```

INVALID SCENARIOS

1. ADD STUDENT TO DATABASE (using invalid values – lastName)

STEPS

A. SHORT LASTNAME (below 3)

1. Select **POST** method in Postman
2. Paste URL to Postman REST API
3. Add data to Request tab:

```
{
  "firstName": "Anna",
  "lastName": "Bo",
  "email": "annaB@email.com",
  "age": 15
}
```

B. LONG LASTNAME (above 50)

1. Select **POST** method in Postman
2. Paste URL to Postman REST API
3. Add data to Request tab:

```
{
  "firstName": "Anna",
  "lastName": "Loremipsumdo
  lorsijsklametconsectetu
  reficituriutjgh",
  "email": "annaB@email.com",
  "age": 20
}
```

EXPECTED RESULT

STATUS 400 BAD REQUEST

RESPONSE No data shown in Response tab

2. ADD STUDENT TO DATABASE (using invalid values – age)

STEPS

A. AGE BELOW 1

1. Select **POST** method in Postman
2. Paste URL to Postman REST API
3. Add data to Request tab:

```
{
  "firstName": "Anna",
  "lastName": "Boon",
  "email": "annaB@email.com",
  "age": 0
}
```

B. AGE OVER 150

1. Select **POST** method in Postman
2. Paste URL to Postman REST API
3. Add data to Request tab:

```
{
  "firstName": "Anna",
  "lastName": "Boon",
  "email": "annaB@email.com",
  "age": 151
}
```

EXPECTED RESULT

STATUS 400 BAD REQUEST

RESPONSE No data shown in Response tab

3. ADD STUDENT TO DATABASE (using invalid values – email without „@“)

STEPS

1. Select **POST** method in Postman
2. Paste URL to Postman REST API
3. Add data to Request tab:

```
{
  "firstName": "Anna",
  "lastName": "Boon",
  "email": "annaBemail.com",
  "age": 15
}
```

EXPECTED RESULT

STATUS

400 BAD REQUEST

RESPONSE

No data shown in Response tab

4. ADD STUDENT TO DATABASE (with missing values – firstName)

STEPS

1. Select **POST** method in Postman
2. Paste URL to Postman REST API
3. Add data to Request tab:

```
{
  "firstName": ,
  "lastName": "Boon",
  "email": "annaB@emailcom",
  "age": 15
}
```

EXPECTED RESULT

STATUS

400 BAD REQUEST

RESPONSE

No data shown in Response tab

DELETE METHOD

PREREQUISITES

- Workbench database set up with these [credentials](#).
- Postman set on HTTP request with this [REST API URL](#).

VALID SCENARIOS

1. DELETE ALL DATA ABOUT EXISTING STUDENT

STEPS

1. Data preparation in MySQL Workbench:

```
SELECT * FROM student where id=1451;
```

id	age	email	first_name	last_name
1451	1	annaBemail.com	An	BO
NULL	NULL	NULL	NULL	NULL
2. Select **DELETE** method in Postman
3. Paste URL to Postman REST API
4. Add ID „1451“ in the end of URL

EXPECTED RESULT

STATUS

200 OK

RESPONSE

All data removed (no data shown)

INVALID SCENARIOS

1. DELETE ALL DATA ABOUT NON-EXISTING STUDENT (using deleted ID)

STEPS

1. Data preparation in MySQL Workbench:

```
SELECT * FROM student where id=1451;
```

#	Time	Action	Message
✓ 1	18:00:56	SELECT * FROM student where id=1451 LIMIT 0, 1000	0 row(s) returned
2. Select **DELETE** method in Postman
3. Paste URL to Postman REST API
4. Add ID „1451“ in the end of URL

EXPECTED RESULT

STATUS

400 BAD REQUEST

RESPONSE

No data shown in Response tab

2. DELETE ALL DATA ABOUT NON-EXISTING STUDENT (using invalid values)

STEPS

1. Data preparation in MySQL Workbench:

```
SELECT * FROM student where id=xxx;
```

#	Time	Action	Message
✗ 1	18:02:13	SELECT * FROM student where id=xxx LIMIT 0, 1000	Error Code: 1054.
2. Select **DELETE** method in Postman
3. Paste URL to Postman REST API
4. Add ID „xxx“ in the end of URL

EXPECTED RESULT

STATUS

400 BAD REQUEST

RESPONSE

No data shown in Response tab

TEST EXECUTION

GET METHOD

VALID SCENARIOS

1.

GET DATA ABOUT EXISTING STUDENT

TEST STATUS

EXPECTED RESULT	STATUS 200 OK	RESPONSE { "id": 1318, "firstName": "Eric", "lastName": "DARK", "email": "EricDark@Mail.Com", "age": 20 }
ACTUAL RESULT	STATUS 200 OK	RESPONSE { "id": 1318, "firstName": "Eric", "lastName": "DARK", "email": "EricDark@Mail.Com", "age": 20 }

GET

http://108.143.193.45:8080/api/v1/students/1318

Send

ParamsAuthorizationHeaders (6)BodyPre-request ScriptTestsSettings

Query Params

Key	Value	Bulk Edit
Key	Value	

BodyCookiesHeaders (5)Test Results

Status: 200 OKTime: 164 msSize: 249 BSave Response

PrettyRawPreviewVisualizeJSON

```
1 {  
2   "id": 1318,  
3   "firstName": "Eric",  
4   "lastName": "DARK",  
5   "email": "EricDark@Mail.Com",  
6   "age": 20  
7 }
```

2. GET DATA ABOUT ALL EXISTING STUDENTS

TEST STATUS

EXPECTED RESULT

STATUS

200 OK

RESPONSE

List of all students in following format:

```
{
  "id": ,
  "firstName": ,
  "lastName": ,
  "email": ,
  "age":
},
```

ACTUAL RESULT

STATUS

200 OK

RESPONSE

List of all students in following format:

```
{
  "id": ,
  "firstName": ,
  "lastName": ,
  "email": ,
  "age":
},
```

PASSED

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

- Method:** GET
- URL:** http://108.143.193.45:8080/api/v1/students/
- Status:** 200 OK
- Time:** 192 ms
- Size:** 68.53 KB
- Response Format:** JSON
- Response Body (Pretty):**

```
1 {
2   {
3     "id": 219,
4     "firstName": "",
5     "lastName": "JOSHUA",
6     "email": "jjohn34@yourmail.com",
7     "age": 34
8   },
9   {
10    "id": 242,
11    "firstName": "Martin",
12    "lastName": "KOMÁR",
13    "email": "komaz@seznam.cz",
14    "age": 34
15  },
16  {
17    "id": 243,
18    "firstName": "",
19    "lastName": "SMITH",
20    "email": "peter21@gmail.com",
21    "age": 34
22  },
23  {
24    "id": 254,
25    "firstName": "Lucia",
26    "lastName": "LESNÁ",
27    "email": "abc@abc.com",
28    "age": 344
29  },
30  {
31    "id": 264,
```

INVALID SCENARIOS

1.

GET DATA ABOUT NON-EXISTING STUDENT (using non-existing ID)

TEST STATUS

EXPECTED RESULT	STATUS 404 NOT FOUND	RESPONSE No data shown in Response tab	FAILED
ACTUAL RESULT	STATUS 500 INTERNAL SERVER ERROR	RESPONSE "timestamp": "2024-06-30T11:32:03.456+00:00", "status": 500, "error": "Internal Server Error", "message": "", "path": "/api/v1/students/0000"	

GET

http://108.143.193.45:8080/api/v1/students/0000

Send

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

Query Params

	Key	Value	Bulk Edit
	Key	Value	

Body Cookies Headers (4) Test Results

Status: 500 Internal Server Error Time: 154 ms Size: 288 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "timestamp": "2024-06-30T11:32:03.456+00:00",
3   "status": 500,
4   "error": "Internal Server Error",
5   "message": "",
6   "path": "/api/v1/students/0000"
7 }
```

2.

GET DATA ABOUT NON-EXISTING STUDENT (using invalid input as ID number)

TEST STATUS

EXPECTED RESULT	STATUS 400 BAD REQUEST	RESPONSE No data shown in Response tab	PASSED
ACTUAL RESULT	STATUS 400 BAD REQUEST	RESPONSE "timestamp": "2024-06-30T11:35:27.782+00:00", "status": 400, "error": "Bad Request", "message": "", "path": "/api/v1/students/XXX"	

GET

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

Send

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

Query Params

	Key	Value	Bulk Edit
	Key	Value	

Body Cookies Headers (4) Test Results

Status: 400 Bad Request Time: 86 ms Size: 267 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "timestamp": "2024-06-30T11:35:27.782+00:00",
3   "status": 400,
4   "error": "Bad Request",
5   "message": "",
6   "path": "/api/v1/students/XXX"
7 }
```

POST METHOD

VALID SCENARIOS

1. ADD STUDENT TO DATABASE (check status code)

TEST STATUS

EXPECTED RESULT

STATUS

201 CREATED

RESPONSE

```
{
  "id": ,
  "firstName": "Anna",
  "lastName": "Boyle",
  "email": "annaB@mail.com",
  "age": 30
}
```

ACTUAL RESULT

STATUS

200 OK

RESPONSE

```
{
  "id": 1447,
  "firstName": "Anna",
  "lastName": "BOYLE",
  "email": "annaB@mail.com",
  "age": 30
}
```

FAILED

POST

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

Query Params

Key	Value	Bulk Edit
Key	Value	

Body Cookies Headers (5) Test Results

Status: 200 OK Time: 387 ms Size: 247 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1447,
3   "firstName": "Anna",
4   "lastName": "BOYLE",
5   "email": "annaB@mail.com",
6   "age": 30
7 }
```

2. CHECK IF DATABASE SAVES VALUES IN LOWERCASE

TEST STATUS

EXPECTED RESULT

STATUS

201 CREATED

RESPONSE

```
{
  "id": ,
  "firstName": "lisa",
  "lastName": "smith",
  "email": "lisasmith@mail.com",
  "age": 29
}
```

ACTUAL RESULT

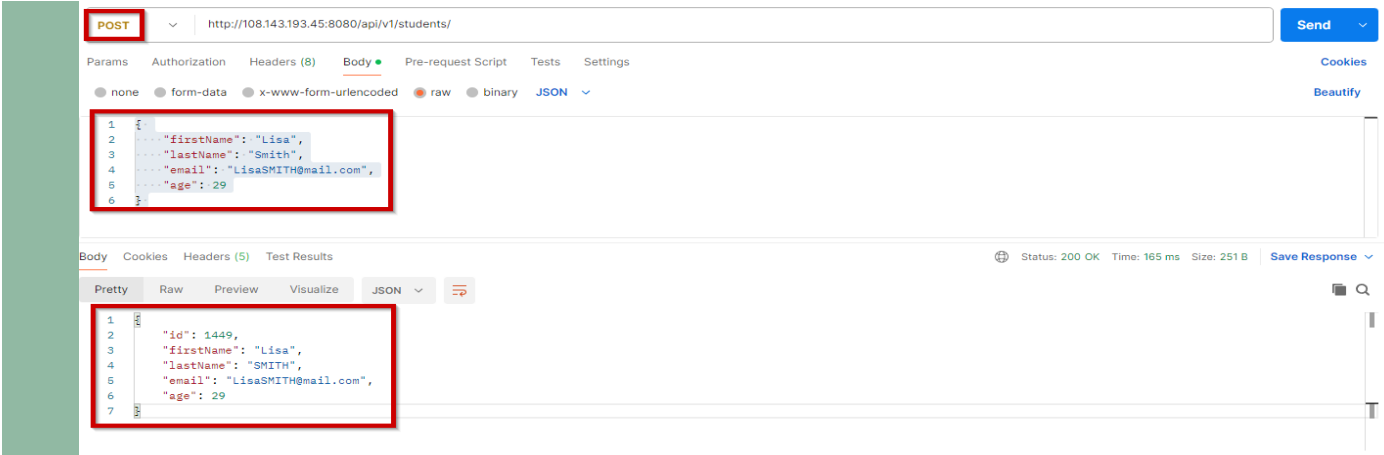
STATUS

200 OK

RESPONSE

```
{
  "id": 1449,
  "firstName": "Lisa",
  "lastName": "SMITH",
  "email": "lisasmith@mail.com",
  "age": 29
}
```

FAILED



3. CHECK IF DATABASE SAVES VALUES CORRECTLY (in different language)

TEST STATUS

EXPECTED RESULT

STATUS
201 CREATED

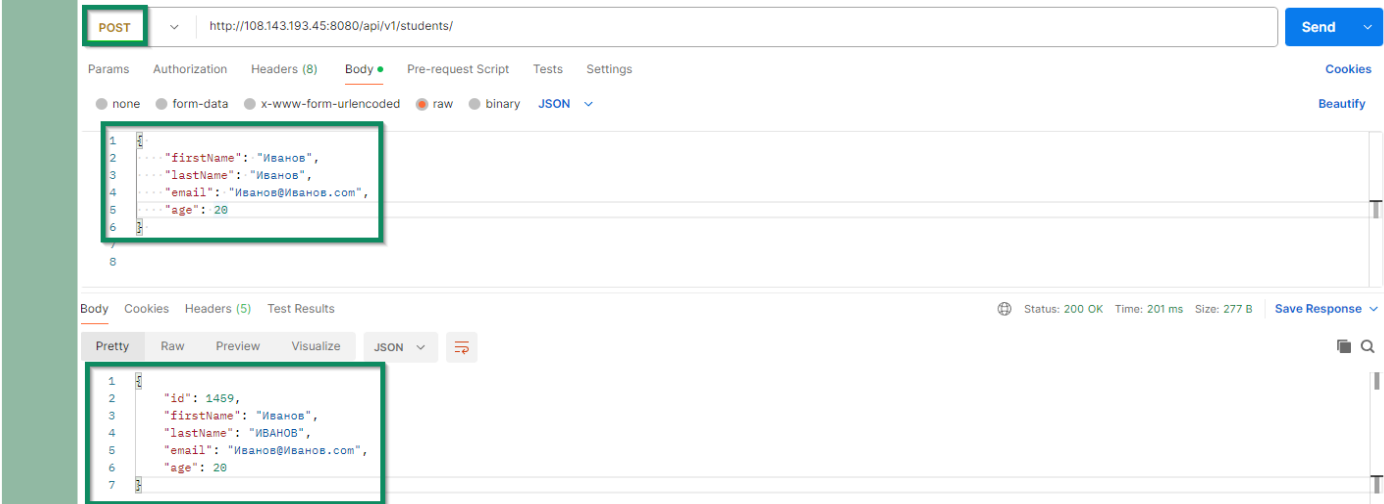
RESPONSE
{
 "id": ,
 "firstName": "ИВАНОВ",
 "lastName": "ИВАНОВ",
 "email": "ИВАНОВ@ИВАНОВ.COM",
 "age": 20
}

ACTUAL RESULT

STATUS
200 OK

RESPONSE
{
 "id": 1459,
 "firstName": "ИВАНОВ",
 "lastName": "ИВАНОВ",
 "email": "ИВАНОВ@ИВАНОВ.COM",
 "age": 20
}

PASSED



INVALID SCENARIOS

1. ADD STUDENT TO DATABASE (using invalid values – lastName)

	A. SHORT LASTNAME (below 3)	B. LONG LASTNAME (above 50)
EXPECTED RESULT	<div>STATUS 400 BAD REQUEST</div> <div>RESPONSE No data shown in Response tab</div>	<div>STATUS 400 BAD REQUEST</div> <div>RESPONSE No data shown in Response tab</div>
ACTUAL RESULT	<div>STATUS 200 OK</div> <div>RESPONSE { "id": 1481, "firstName": "Anna", "lastName": "BO", "email": "annaB@email.com", "age": 15 }</div>	<div>STATUS 200 OK</div> <div>RESPONSE { "id": 1482, "firstName": "Anna", "lastName": "LOREMIPSUMDOLORSIJKLTAMETCONSECTETUREFFICITURIUTJGH", "email": "annaB@email.com", "age": 15 }</div>
TEST STATUS	FAILED	FAILED

POST

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

Send

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

none form-data x-www-form-urlencoded raw binary JSON

```
1 {
2   "firstName": "Anna",
3   "lastName": "Bo",
4   "email": "annaB@email.com",
5   "age": 15
6 }
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1481,
3   "firstName": "Anna",
4   "lastName": "BO",
5   "email": "annaB@email.com",
6   "age": 15
7 }
```

Status: 200 OK Time: 191 ms Size: 245 B Save Response

POST

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

Send

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

none form-data x-www-form-urlencoded raw binary JSON

```
1 {
2   "firstName": "Anna",
3   "lastName": "LOREMIPSUMDOLORSIJKLTAMETCONSECTETUREFFICITURIUTJGH",
4   "email": "annaB@email.com",
5   "age": 15
6 }
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1482,
3   "firstName": "Anna",
4   "lastName": "LOREMIPSUMDOLORSIJKLTAMETCONSECTETUREFFICITURIUTJGH",
5   "email": "annaB@email.com",
6   "age": 15
7 }
```

Status: 200 OK Time: 187 ms Size: 294 B Save Response

2. ADD STUDENT TO DATABASE (using invalid values – age)

	A. AGE BELOW 1	B. AGE OVER 150
EXPECTED RESULT	STATUS 400 BAD REQUEST	STATUS 400 BAD REQUEST
	RESPONSE No data shown in Response tab	RESPONSE No data shown in Response tab
ACTUAL RESULT	STATUS 200 OK	STATUS 200 OK
	RESPONSE <pre>{ "id": 1486, "firstName": "Anna", "lastName": "BOON", "email": "annaB@email.com", "age": 0 }</pre>	RESPONSE <pre>{ "id": 1485, "firstName": "Anna", "lastName": "BOON", "email": "annaB@email.com", "age": 151 }</pre>
TEST STATUS	FAILED	FAILED

The screenshot displays two test cases in a REST client interface. Both tests are POST requests to the endpoint `http://108.143.193.45:8080/api/v1/students/`.

Test Case 1 (Top):

- Request Body (JSON):**

```
{
  "firstName": "Anna",
  "lastName": "BOON",
  "email": "annaB@email.com",
  "age": 0
}
```
- Response:** Status: 200 OK, Time: 155 ms, Size: 246 B. The response body is

```
{
  "id": 1486,
  "firstName": "Anna",
  "lastName": "BOON",
  "email": "annaB@email.com",
  "age": 0
}
```
- Test Status:** FAILED

Test Case 2 (Bottom):

- Request Body (JSON):**

```
{
  "firstName": "Anna",
  "lastName": "BOON",
  "email": "annaB@email.com",
  "age": 151
}
```
- Response:** Status: 200 OK, Time: 159 ms, Size: 248 B. The response body is

```
{
  "id": 1485,
  "firstName": "Anna",
  "lastName": "BOON",
  "email": "annaB@email.com",
  "age": 151
}
```
- Test Status:** FAILED

3. ADD STUDENT TO DATABASE (using invalid values – email without „@“)

TEST STATUS

EXPECTED RESULT

STATUS

400 BAD REQUEST

RESPONSE

No data shown in Response tab

ACTUAL RESULT

STATUS

200 OK

RESPONSE

```
{
  "id": 1455,
  "firstName": "Anna",
  "lastName": "BOON",
  "email": "annaBemail.com",
  "age": 15
}
```

FAILED

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

Body

```
{
  "firstName": "Anna",
  "lastName": "Boon",
  "email": "annaBemail.com",
  "age": 15
}
```

Status: 200 OK Time: 203 ms Size: 246 B Save Response

```
{
  "id": 1455,
  "firstName": "Anna",
  "lastName": "BOON",
  "email": "annaBemail.com",
  "age": 15
}
```

4. ADD STUDENT TO DATABASE (with missing values – firstName)

TEST STATUS

EXPECTED RESULT

STATUS

400 BAD REQUEST

RESPONSE

No data shown in Response tab

ACTUAL RESULT

STATUS

400 BAD REQUEST

RESPONSE

```
{
  "timestamp": "2024-07-27T16:49:30.098+00:00",
  "status": 400,
  "error": "Bad Request",
  "message": "",
  "path": "/api/v1/students/"
}
```

PASSED

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

Body

```
{
  "firstName": "x",
  "lastName": "Boon",
  "email": "annaBemail.com",
  "age": 15
}
```

Status: 400 Bad Request Time: 70 ms Size: 264 B Save Response

```
{
  "timestamp": "2024-07-27T16:49:30.098+00:00",
  "status": 400,
  "error": "Bad Request",
  "message": "",
  "path": "/api/v1/students/"
}
```

DELETE METHOD

VALID SCENARIOS

1.

DELETE ALL DATA ABOUT EXISTING STUDENT

TEST STATUS

EXPECTED RESULT	STATUS	RESPONSE	PASSED
	200 OK	All data removed (no data shown)	
ACTUAL RESULT	STATUS	RESPONSE	
	200 OK	All data removed (no data shown)	

DELETE

http://108.143.193.45:8080/api/v1/students/1451

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Query Params

Key	Value	Bulk Edit
Key	Value	

body

Cookies

Headers (4)

Test Results

Status: 200 OK

Time: 136 ms

Size: 123 B

Save Response

Pretty

Raw

Preview

Visualize

Text

1

INVALID SCENARIOS

1.

DELETE ALL DATA ABOUT NON-EXISTING STUDENT (using deleted ID)

TEST STATUS

EXPECTED RESULT	STATUS	RESPONSE	FAILED
	400 BAD REQUEST	No data shown in Response tab	
ACTUAL RESULT	STATUS	RESPONSE	
	500 INTERNAL SERVER ERROR	<pre>"timestamp": "2024-07-27T17:13:23.0796+00:00", "status": 500, "error": "Internal Server Error", "message": "", "path": "/api/v1/students/1451"</pre>	

DELETE

http://108.143.193.45:8080/api/v1/students/1451

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Query Params

Key	Value	Bulk Edit
Key	Value	

Body

Cookies

Headers (4)

Test Results

Status: 500 Internal Server Error

Time: 142 ms

Size: 288 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

1

2

3

4

5

6

7

2. DELETE ALL DATA ABOUT NON-EXISTING STUDENT (using invalid values)

TEST STATUS

EXPECTED RESULT

STATUS

400 BAD REQUEST

RESPONSE

No data shown in Response tab

ACTUAL RESULT

STATUS

400 BAD REQUEST

RESPONSE

```
"timestamp": "2024-07-27T17:18:41.885+00:00",
"status": 400,
"error": "Bad Request",
"message": "",
"path": "/api/v1/students/xxx"
```

PASSED

DELETE Send

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

Query Params

Key	Value	Bulk Edit
Key	Value	

Body Cookies Headers (4) Test Results Status: 400 Bad Request Time: 73 ms Size: 267 B Save Response

Pretty Raw Preview Visualize JSON

```
1  [
2    "timestamp": "2024-07-27T17:18:41.885+00:00",
3    "status": 400,
4    "error": "Bad Request",
5    "message": "",
6    "path": "/api/v1/students/xxx"
7  ]
```

BUG REPORT

PREREQUISITES

- Workbench database set up with these [credentials](#).
- Postman set on HTTP request with this [REST API URL](#).

VALIDATION CRITERIA

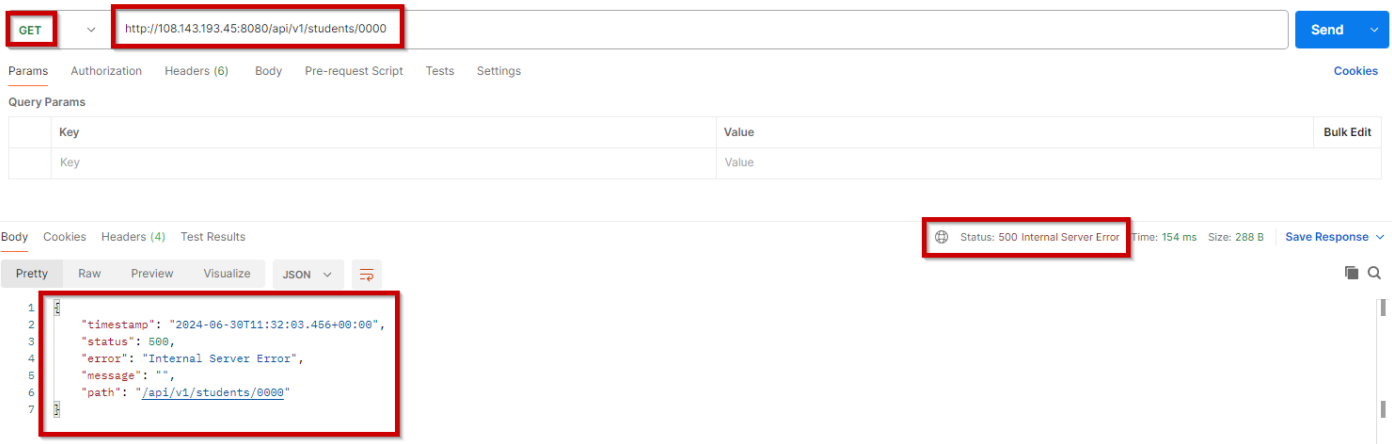
- Value for age is **from 1 to 150**.
- Value for first_name and last_name has **from 3 to 50 characters**.
- Value for email contains at least one „@“ and „.“ character.

ATTRIBUTES

- **All fields are mandatory**.
- All values are saved as **lowercase**.

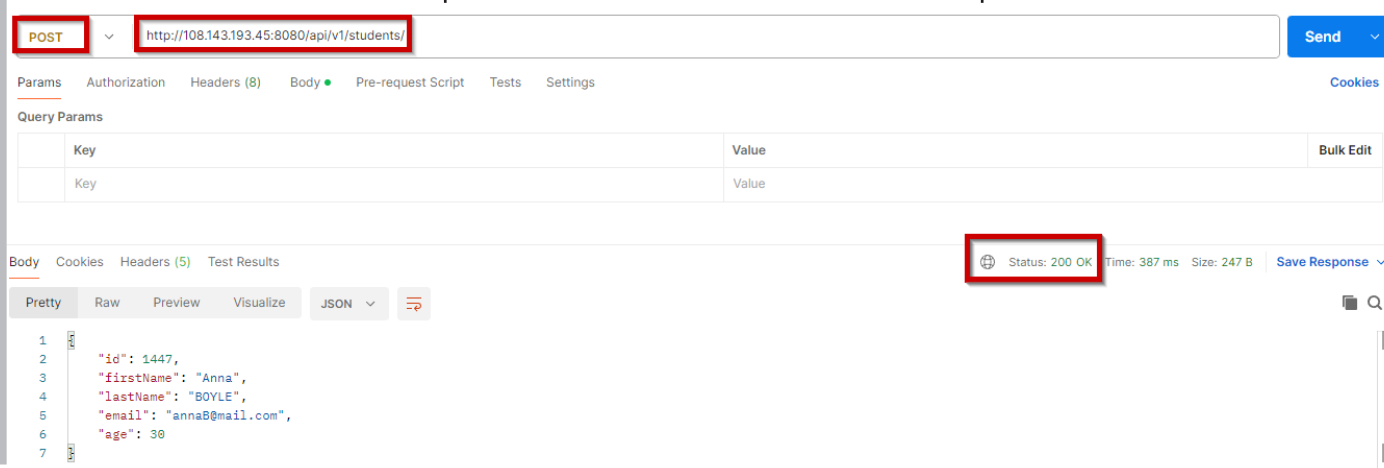
1. GET DATA ABOUT NON-EXISTING STUDENT (using non-existing ID)

STEPS TO REPRODUCE	DATA INPUT	EXPECTED RESULT	ACTUAL RESULT	PRIORITY
<div>1. Data preparation in MySQL Workbench</div> <div>2. Select GET method in Postman</div> <div>3. Paste URL to Postman REST API</div> <div>4. Add ID „0000“ in the end of URL</div> <div>5. Press Send button</div>	<div>MYSQL WORKBENCH REQUEST</div> <div><code>SELECT * FROM student where id=0000;</code></div> <div>POSTMAN URL</div> <div>http://108.143.193.45:8080/api/v1/students/0000</div>	<div>STATUS</div> <div>400 BAD REQUEST</div> <div>RESPONSE</div> <div>No data shown in Response tab</div>	<div>STATUS</div> <div>500 INTERNAL SERVER ERROR</div> <div>RESPONSE</div> <div><code>"timestamp": "2024-06-30T11:32:03.456+00:00",</code> <code>"status": 500,</code> <code>"error": "Internal Server Error",</code> <code>"message": "",</code> <code>"path": "/api/v1/students/0000"</code></div>	<div>Medium</div>



2. ADD STUDENT TO DATABASE (check status code)

STEPS TO REPRODUCE	DATA INPUT	EXPECTED RESULT	ACTUAL RESULT	PRIORITY
<ol style="list-style-type: none">1. Select POST method in Postman2. Paste URL to REST API3. Add data to Request tab4. Press Send button	POSTMAN URL http://108.143.193.45:8080/api/v1/students/ POSTMAN REQUEST TAB <pre>{ "firstName": "Anna", "lastName": "Boyle", "email": "annaB@mail.com", "age": 30 }</pre>	STATUS 201 CREATED RESPONSE <pre>{ "id": , "firstName": "Anna", "lastName": "Boyle", "email": "annaB@mail.com", "age": 30 }</pre>	STATUS 200 OK RESPONSE <pre>{ "id": 1447, "firstName": "Anna", "lastName": "BOYLE", "email": "annaB@mail.com", "age": 30 }</pre>	Low



The screenshot shows the Postman interface for a POST request. The method is set to POST, and the URL is http://108.143.193.45:8080/api/v1/students/. The request body is a JSON object with the following fields: firstName (Anna), lastName (Boyle), email (annaB@mail.com), and age (30). The response status is 200 OK, and the response body is a JSON object with the following fields: id (1447), firstName (Anna), lastName (BOYLE), email (annaB@mail.com), and age (30).

3. CHECK IF DATABASE SAVES VALUES IN LOWERCASE

STEPS TO REPRODUCE	DATA INPUT	EXPECTED RESULT	ACTUAL RESULT	PRIORITY
<ol style="list-style-type: none"> 1. Select POST method in Postman 2. Paste URL to REST API 3. Add data to Request tab 4. Press Send button 	POSTMAN URL http://108.143.193.45:8080/api/v1/students/ POSTMAN REQUEST TAB <pre>{ "firstName": "Lisa", "lastName": "SMITH", "email": "LisaSMITH@mail.com", "age": 29 }</pre>	STATUS 201 CREATED RESPONSE <pre>{ "id": , "firstName": "lisa", "lastName": "smith", "email": "lisasmith@mail.com", "age": 29 }</pre>	STATUS 200 OK RESPONSE <pre>{ "id": 1449, "firstName": "Lisa", "lastName": "SMITH", "email": "lisaSMITH@mail.com", "age": 29 }</pre>	Low

The screenshot shows the Postman interface for a POST request to `http://108.143.193.45:8080/api/v1/students/`. The request method is **POST**. The request body is a JSON object:

```
{
  "firstName": "Lisa",
  "lastName": "SMITH",
  "email": "LisaSMITH@mail.com",
  "age": 29
}
```

The response status is **200 OK**. The response body is a JSON object:

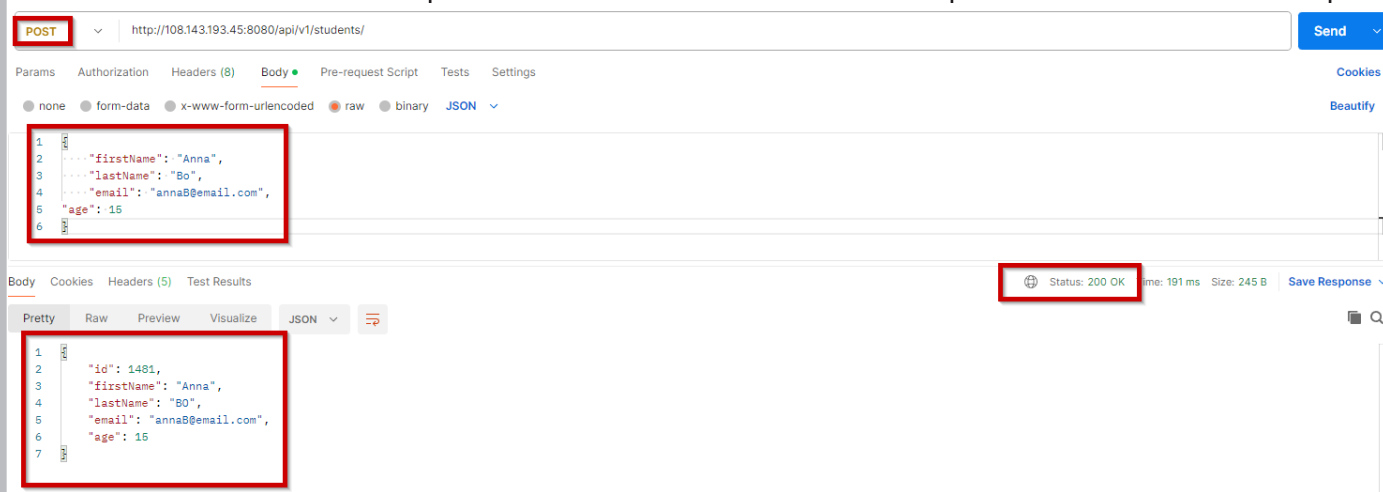
```
{
  "id": 1449,
  "firstName": "Lisa",
  "lastName": "SMITH",
  "email": "lisaSMITH@mail.com",
  "age": 29
}
```

Red boxes in the original image highlight the request and response bodies.

4. ADD STUDENT TO DATABASE (using invalid values – lastName)

A. SHORT (below 3)

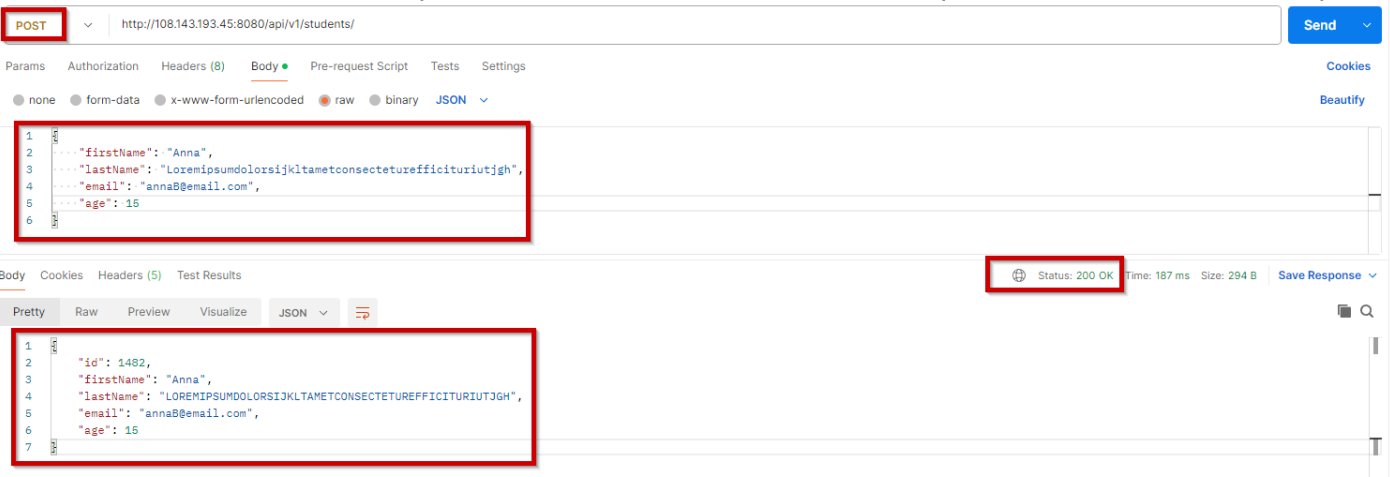
STEPS TO REPRODUCE	DATA INPUT	EXPECTED RESULT	ACTUAL RESULT	PRIORITY
<ol style="list-style-type: none"> Select POST method in Postman Paste URL to REST API Add data to Request tab Press Send button 	POSTMAN URL http://108.143.193.45:8080/api/v1/students/ POSTMAN REQUEST TAB <pre>{ "firstName": "Anna", "lastName": "Bo", "email": "annaB@email.com", "age": 15 }</pre>	STATUS 400 BAD REQUEST RESPONSE No data shown in Response tab	STATUS 200 OK RESPONSE <pre>{ "id": 1481, "firstName": "Anna", "lastName": "BO", "email": "annaB@email.com", "age": 15 }</pre>	High



5. ADD STUDENT TO DATABASE (using invalid values – lastName)

B. LONG (above 50)

STEPS TO REPRODUCE	DATA INPUT	EXPECTED RESULT	ACTUAL RESULT	PRIORITY
<ol style="list-style-type: none">Select POST method in PostmanPaste URL to REST APIAdd data to Request tabPress Send button	<p>POSTMAN URL</p> <p>http://108.143.193.45:8080/api/v1/students/</p> <p>POSTMAN REQUEST TAB</p> <pre>{ "firstName": "Anna", "lastName": "Loremipsumdolorsijkltametconsectetur efficituriutjgh", "email": "annaB@email.com", "age": 20 }</pre>	<p>STATUS</p> <p>400 BAD REQUEST</p> <p>RESPONSE</p> <p>No data shown in Response tab</p>	<p>STATUS</p> <p>200 OK</p> <p>RESPONSE</p> <pre>{ "id": 1482, "firstName": "Anna", "lastName": "LOREMIPSUMDOLORSI JKLTAMETCONSECTETUREFFICITURI UTJGH", "email": "annaB@email.com", "age": 15 }</pre>	High



6. ADD STUDENT TO DATABASE (using invalid values – age)

A. AGE BELOW 1

STEPS TO REPRODUCE

1. Select **POST** method in Postman
2. Paste URL to REST API
3. Add data to Request tab
4. Press Send button

DATA INPUT

POSTMAN URL

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

POSTMAN REQUEST TAB

```
{
  "firstName": "Anna",
  "lastName": "Boon",
  "email": "annaB@email.com",
  "age": 0
}
```

EXPECTED RESULT

STATUS

400 BAD REQUEST

RESPONSE

No data shown in Response tab

ACTUAL RESULT

STATUS

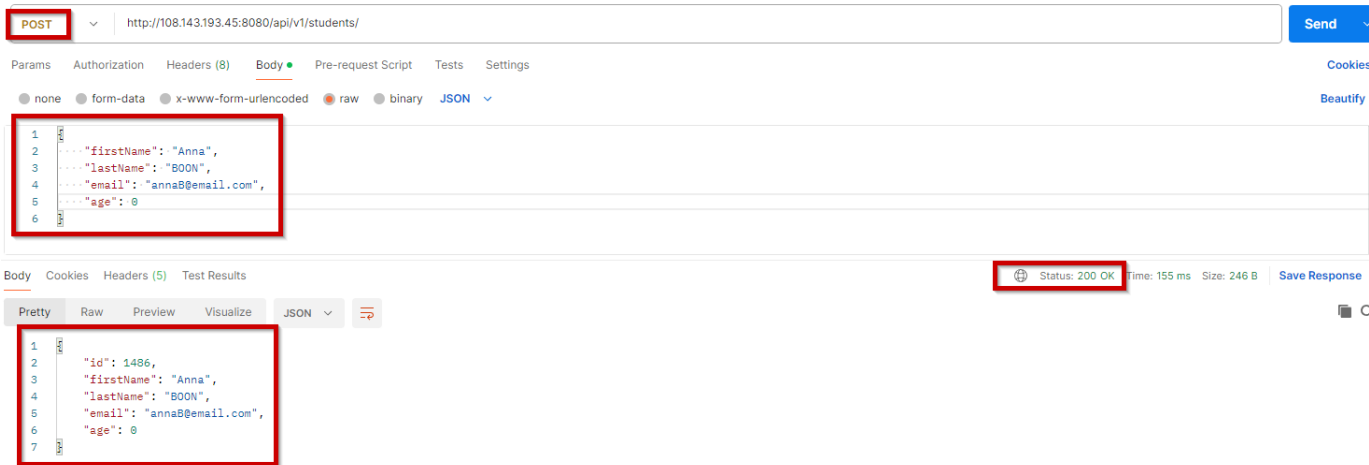
200 OK

RESPONSE

```
{
  "id": 1486,
  "firstName": "Anna",
  "lastName": "BOON",
  "email": "annaB@email.com",
  "age": 0
}
```

PRIORITY

High



7. ADD STUDENT TO DATABASE (using invalid values – age)

B. AGE OVER 150

STEPS TO REPRODUCE	DATA INPUT	EXPECTED RESULT	ACTUAL RESULT	PRIORITY
<ol style="list-style-type: none"> 1. Select POST method in Postman 2. Paste URL to REST API 3. Add data to Request tab 4. Press Send button 	POSTMAN URL http://108.143.193.45:8080/api/v1/students/ POSTMAN REQUEST TAB <pre>{ "firstName": "Anna", "lastName": "Boon", "email": "annaB@email.com", "age": 151 }</pre>	STATUS 400 BAD REQUEST RESPONSE No data shown in Response tab	STATUS 200 OK RESPONSE <pre>{ "id": 1485, "firstName": "Anna", "lastName": "BOON", "email": "annaB@email.com", "age": 151 }</pre>	High

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

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

none form-data x-www-form-urlencoded raw binary JSON

```
1 {
2   "firstName": "Anna",
3   "lastName": "BOON",
4   "email": "annaB@email.com",
5   "age": 151
6 }
```

Body Cookies Headers (5) Test Results

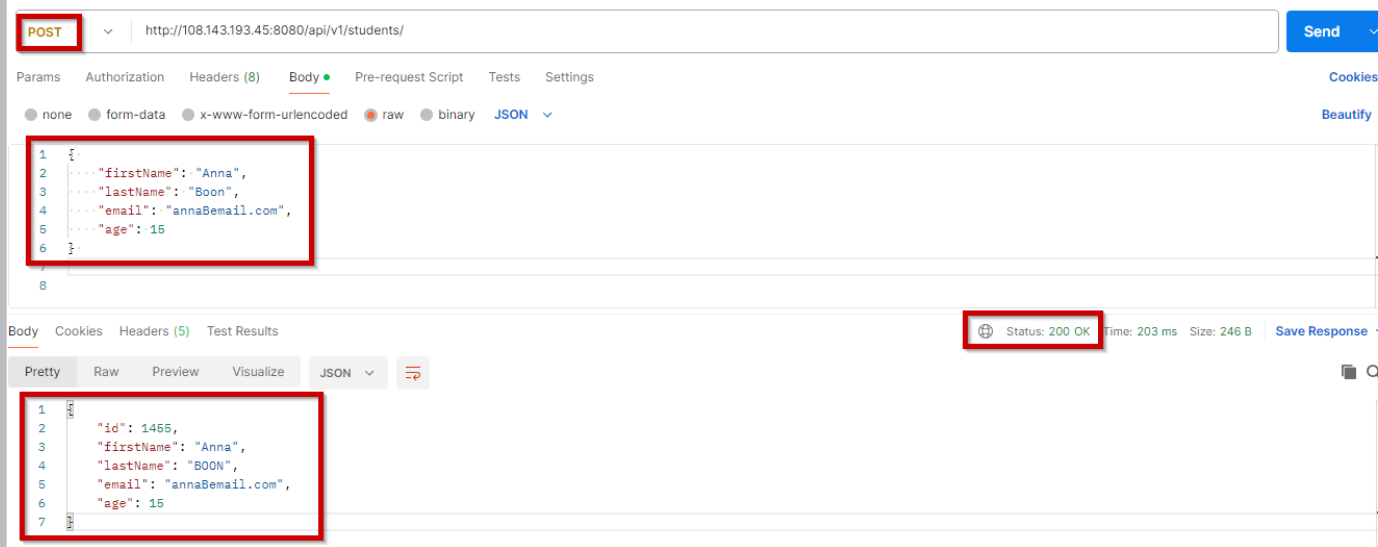
Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1485,
3   "firstName": "Anna",
4   "lastName": "BOON",
5   "email": "annaB@email.com",
6   "age": 151
7 }
```

Status: 200 OK Time: 159 ms Size: 248 B Save Response

8. ADD STUDENT TO DATABASE (using invalid values – email without „@“)

STEPS TO REPRODUCE	DATA INPUT	EXPECTED RESULT	ACTUAL RESULT	PRIORITY
<ol style="list-style-type: none"> Select POST method in Postman Paste URL to REST API Add data to Request tab Press Send button 	<p>POSTMAN URL</p> <p>http://108.143.193.45:8080/api/v1/students/</p> <p>POSTMAN REQUEST TAB</p> <pre>{ "firstName": "Anna", "lastName": "Boon", "email": "annaBemall.com", "age": 15 }</pre>	<p>STATUS</p> <p>400 BAD REQUEST</p> <p>RESPONSE</p> <p>No data shown in Response tab</p>	<p>STATUS</p> <p>200 OK</p> <p>RESPONSE</p> <pre>{ "id": 1455, "firstName": "Anna", "lastName": "BOON", "email": "annaBemall.com", "age": 15 }</pre>	<p>High</p>



9. DELETE ALL DATA ABOUT NON-EXISTING STUDENT (using deleted ID)

STEPS TO REPRODUCE	DATA INPUT	EXPECTED RESULT	ACTUAL RESULT	PRIORITY
<ol style="list-style-type: none">1. Data preparation in MySQL Workbench2. Select DELETE method in Postman3. Paste URL to Postman REST API4. Add ID „1451“ in the end of URL5. Press Send button	POSTMAN URL http://108.143.193.45:8080/api/v1/students/1451	STATUS 400 BAD REQUEST RESPONSE No data shown in Response tab	STATUS 500 INTERNAL SERVER ERROR RESPONSE <pre>"timestamp": "2024-07-27T17:13:23.0796+00:00", "status": 500, "error": "Internal Server Error", "message": "", "path": "/api/v1/students/1451"</pre>	High

DELETE

http://108.143.193.45:8080/api/v1/students/1451

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Query Params

	Key	Value	Bulk Edit
	Key	Value	

Body

Cookies

Headers (4)

Test Results

Status: 500 Internal Server Error

Time: 142 ms

Size: 288 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1  "timestamp": "2024-07-27T17:13:23.796+00:00",
2  "status": 500,
3  "error": "Internal Server Error",
4  "message": "",
5  "path": "/api/v1/students/1451"
```