

FINÁLNÍ PROJEKT  
č.1



Autor: Kulykova Siuzanna Volodymyrivna  
Datum: 11.03.2024

## **OBSAH**

ZADÁNÍ	3
TESTOVACÍ SCÉNÁŘE	4
EXEKUCE TESTŮ	12
BUG REPORTS	13

# 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.

REST rozhraní běží na adrese <http://108.143.193.45:8080/api/v1/students/>

Otestujte metody GET, POST a DELETE.

Platí, že:

1. metoda GET zobrazí data o existujícím studentovi
2. metoda POST založí záznam o studentovi
3. metoda DELETE smaže data o studentovi

# TESTOVACÍ SCÉNÁŘE

*Na základě uvedených testovacích scénářů jsem ověřila funkčnost aplikace.*

**Testovací scénář 1 Zkontroluji metodu GET, která zobrazí data o existujícím studentovi**

## **Testovací případ 1**

**Název:** Chování systému při zadání platného id studenta.

**Testovací data:** platné id studenta 352

### **Steps to reproduce:**

1. Do requestu GET zadám <http://108.143.193.45:8080/api/v1/students/352>
2. Zmačknu tlačítko Send

### **Očekávaný výsledek:**

Vrací se stavový kód 200. Karta Body zobrazí JSON s údaji studenta.

**Výsledek testu: Test proběhl úspěšně.**

## **Testovací případ 2**

**Název:** Chování systému při zadání neplatného id studenta

**Testovací data:** neplatné id studenta 888

### **Steps to reproduce:**

1. Do requestu GET zadám <http://108.143.193.45:8080/api/v1/students/888>
2. Zmačknu tlačítko Send

### **Očekávaný výsledek:**

Vrací se stavový kod 404.

**Výsledek testu: Test proběhl neúspěšně.**

### ***Testovací případ 3***

**Název:** Získání dat všech studentů

**Testovací data:**

**Steps to reproduce:**

1. Do requestu GET zadám <http://108.143.193.45:8080/api/v1/students/>
2. Zmačknu tlačítko Send

**Očekávaný výsledek:**

Vrací se stavový kód 200. Karta Body vydá JSON všech studentů z databáze.

**Výsledek testu:** **Test proběhl úspěšně.**

### ***Testovací případ 4***

**Název:** Chování systému při zadání mezer na místo id studenta

**Testovací data:** mezera

**Steps to reproduce:**

1. Do requestu GET zadám <http://108.143.193.45:8080/api/v1/students/>
2. Zmačknu tlačítko Send

**Očekávaný výsledek:**

Vrací se stavový kód 400.

**Výsledek testu:** **Test proběhl neúspěšně.**

**Testovací scénář 2** *Zkontroluj metodu POST, která založí záznam o studentovi*

### ***Testovací případ 5***

**Název:** Založení dat studenta, když je zadáno údaje do všech vstupů.

**Testovací data:**

```
"firstName": "Alexa",  
"lastName": "Doležal",  
"email": "a_doležal@gmail.com",
```

"age": 30

#### Steps to reproduce:

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data
3. Zmačknu tlačítko Send

#### Očekávaný výsledek:

Vrací se stavový kód 200. Karta Body obsahuje JSON studenta, struktura kterého:

```
{  
  "id":_____,  
  "firstName": "Alexa",  
  "lastName": "Doležal",  
  "email": "a_dolezal@gmail.com",  
  "age": 30  
}
```

Výsledek testu: **Test proběhl neúspěšně.**

#### Testovací případ 6

**Název:** Odmítnutí vložení údajů do databáze, když není zadáno pouze firstName

#### Testovací data:

```
"lastName": "Doležal",  
"email": "a_dolezal@gmail.com",  
"age": 30
```

#### Steps to reproduce:

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data
3. Zmačknu tlačítko Send

**Očekávaný výsledek:** Vrací se stavový kód 400.

Výsledek testu: **Test proběhl neúspěšně.**

#### Testovací případ 7

**Název:** Odmítnutí vložení údajů do databáze, když není zadáno pouze lastName

#### Testovací data:

```
"firstName": "Alexa",
```

```
"email": "a_doležal@gmail.com",  
"age": 30
```

**Steps to reproduce:**

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data
3. Zmačknu tlačítko Send

**Očekávaný výsledek:**

Vrací se stavový kód 400.

**Výsledek testu: Test proběhl neúspěšně.**

***Testovací případ 8***

**Název:** Odmítnutí vložení údajů do databáze, když není zadán pouze email

**Testovací data:**

```
"firstName": "Alexa",  
"lastName": "Doležal",  
"age": 30
```

**Steps to reproduce:**

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data
3. Zmačknu tlačítko Send

**Očekávaný výsledek:**

Vrací se stavový kód 400.

**Výsledek testu: Test proběhl neúspěšně.**

***Testovací případ 9***

**Název:** Odmítnutí vložení údajů do databáze, když není zadán věk

**Testovací data:**

```
"firstName": "Alexa",  
"lastName": "Doležal",  
"email": "a_doležal@gmail.com"
```

**Steps to reproduce:**

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data
3. Zmačknu tlačítko Send

**Očekávaný výsledek:** Vrací se stavový kód 400.

**Testovací případ 10**

**Název:** Odmítnutí vložení údajů do databáze, když nejsou zadány žádné údaje

**Testovací data:**

**Steps to reproduce:**

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw ne zadám testovací data
3. Zmačknu tlačítko Send

**Očekávaný výsledek:**

Vrací se stavový kód 400.

**Výsledek testu:** **Test proběhl neúspěšně.**

**Testovací případ 11**

**Název:** Odmítnutí vložení údajů do databáze, když je zadáno speciální symboly do políčka firstName

**Testovací data:**

```
"firstName": "@#$%^&*()",  
"lastName": "Doležal",  
"email": "a_doležal@gmail.com",  
"age": 30
```

**Steps to reproduce:**

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data
3. Zmačknu tlačítko Send

**Očekávaný výsledek:**



Vrací se stavový kód 400.

**Výsledek testu: Test proběhl neúspěšně.**

### ***Testovací případ 12***

**Název:** Odmítnutí vložení údajů do databáze, když je zadáno mezery do políčka firstName

**Testovací data:**

```
"firstName": " ",  
"lastName": "Doležal",  
"email": "a_doležal@gmail.com",  
"age": 30
```

**Steps to reproduce:**

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data
3. Zmačknu tlačítko Send

**Očekávaný výsledek:** Vrací se stavový kód 400.

**Výsledek testu: Test proběhl neúspěšně.**

### ***Testovací případ 13***

**Název:** Odmítnutí vložení údajů do databáze, když je zadáno číslíce do políčka firstName

**Testovací data:**

```
"firstName": "12345",  
"lastName": "Doležal",  
"email": "a_doležal@gmail.com",  
"age": 30
```

**Steps to reproduce:**

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data
3. Zmačknu tlačítko Send

**Očekávaný výsledek:**

Vrací se stavový kód 400.

**Výsledek testu: Test proběhl neúspěšně.**

### ***Testovací případ 14***

**Název:** Odmítnutí vložení údajů do databáze, když je zadáno údaj délkou 256 symbolů do políčka firstName

**Testovací data:**

```
"firstName": "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa  
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa  
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa  
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa",  
  "lastName": "Doležal",  
  "email": "a_dolezal@gmail.com",  
  "age": 30
```

### Steps to reproduce:

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data
3. Zmačknu tlačítko Send

**Očekávaný výsledek:**

Vrací se stavový kód 400.

**Výsledek testu: Test proběhl neúspěšně.**

### Testovací scénář 3 Zkontrolují metodu DELETE, která smaže data o studentovi

### Testovací případ 15

**Název:** Chování systému při smazání data platného id studenta

**Testovací data:** platné id studenta na moment provedení testu

### Steps to reproduce:

1. Do requestu DELETE zadám <http://108.143.193.45:8080/api/v1/students/>“platné id studenta na moment provedení testu“
2. Zmačknú tlačítko Send

**Očekávaný výsledek:**

Vrací se stavový kód 200. Parametry JSON Body jsou prázdné.

**Výsledek testu: Test proběhl úspěšně.**

### ***Testovací případ 16***

**Název:** Chování systému při pokusu smazat data neplatného id studenta

**Testovací data:** neplatné id studenta 888

#### **Steps to reproduce:**

1. Do requestu DELETE zadám <http://108.143.193.45:8080/api/v1/students/888>
2. Zmačknu tlačítko Send

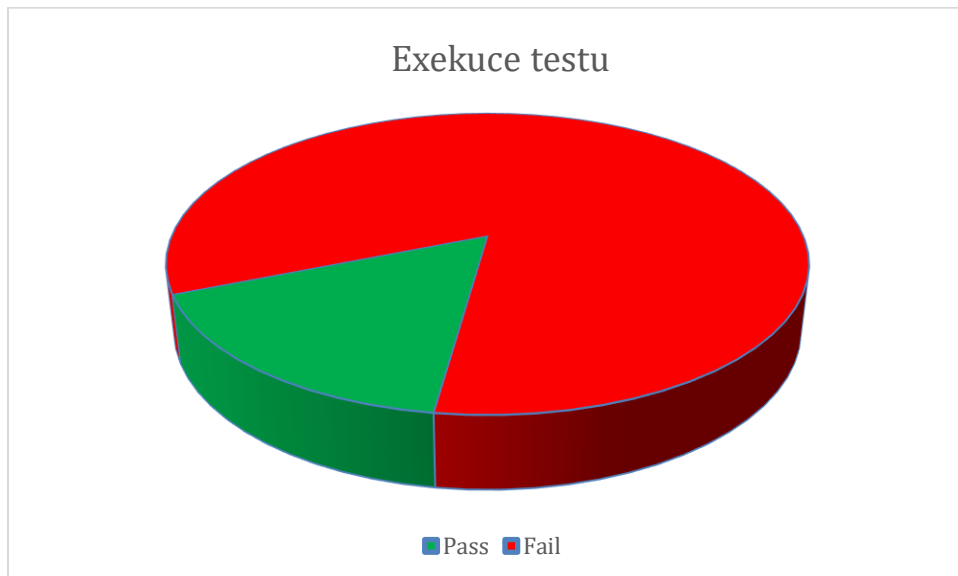
**Očekávaný výsledek:** Vrací se stavový kód 404.

**Výsledek testu:** **Test proběhl neúspěšně.**

## EXEKUCE TESTŮ

*Testovací scénáře jsem provedla, přikládám výsledky testů.*

Bylo provedeno 16 testů, z nichž 13 bylo neúspěšných a 3 úspěšných.



K chybám můžou být přiřazeny následující druhy závažnosti (severity):

- **Blocker:** Chyba, která brání dalšímu testování nebo použití systému. Musí být vyřešena okamžitě.
- **Critical:** Chyba, která způsobuje selhání hlavních funkcí aplikace.
- **Major:** Chyba, která významně ovlivňuje funkčnost, ale neblokuje celkovou práci.
- **Minor:** Chyba, která má malý dopad na funkčnost.
- **Trivial:** Chyba, která je kosmetická nebo má minimální vliv na uživatele.

# BUG REPORTS

Na základě provedených scénářů jsem objevila uvedené chyby aplikace.

bug	probability	severity	test case number	mitigace
1. Vrací se kód 500 při zadání neplatného id studenta	Deterministic ká	<b>Major</b>	2	Místo toho by měl být vrácen kód 404 (Not Found). Na straně serveru je třeba přidat validaci, která zkontroluje formát a strukturu ID před provedením dotazu do databáze. Poskytování jasných a specifických chybových zpráv pomůže uživatelům i vývojářům lépe porozumět problému a rychleji jej vyřešit.
2. Vrací se kód 500 při zadání mezer na místo id studenta	Deterministic ká	<b>Major</b>	4	Přidat validaci, která zkontroluje, zda ID neobsahuje pouze mezery. Pokud jsou zjištěny mezery, vrátí kód 400.
3. Příjmení se v databázi ukládá velkými písmeny.	Deterministic ká	<b>Minor</b>	5	Ujisti se, že v SQL dotazu není žádná funkce, která by převáděla text na velká písmena a opravit to.
4. Vrací se kód 500 při vložení údajů do databáze, když není zadáno pouze firstName	Deterministic ká	<b>Major</b>	6	Přidat kontrolu, která zajistí, že všechny požadované údaje jsou přítomny před vložení do databáze. Pokud chybí firstName nebo jakýkoliv jiný povinný vstup, vrátí kód 400 (Bad Request).
5. Vrací se kód 500 při vložení údajů do databáze, když není zadáno pouze lastName	Deterministic ká	<b>Major</b>	7	Přidat kontrolu, která zajistí, že všechny požadované údaje jsou přítomny před vložení do databáze. Pokud chybí lastName nebo jakýkoliv jiný povinný vstup, vrátí kód 400 (Bad Request).
6. Vrací se kód 500 při vložení údajů do databáze, když není zadán pouze email	Deterministic ká	<b>Major</b>	8	Přidat kontrolu, která zajistí, že všechny požadované údaje jsou přítomny před vložení do databáze. Pokud chybí email nebo jakýkoliv jiný povinný vstup, vrátí kód 400 (Bad Request).
7. Vrací se kód 500 při vložení údajů do databáze, když není zadán věk	Deterministic ká	<b>Major</b>	9	Přidat kontrolu, která zajistí, že všechny požadované údaje jsou přítomny před vložení do databáze. Pokud chybí věk nebo jakýkoliv jiný povinný vstup, vrátí kód 400 (Bad Request).

8. Vrací se kód 500 při vložení údajů do databáze, když nejsou zadány žádné údaje	Deterministic ká	<b>Major</b>	10	Přidat kontrolu, která zajistí, že všechny požadované údaje jsou přítomny před vložení do databáze. Pokud chybí jakýkoliv jiný povinný vstup, vrátí kód 400 (Bad Request).
9. Vložení údajů do databáze, když je zadáno speciální symboly do políčka firstName	Deterministic ká	<b>Major</b>	11	V SQL je nutné omezit ukládání čísel mezer a spec symbolu do polí s textovým datovým typem pomocí kontrolních omezení (constraints) při vytváření tabulek. Konkrétně lze použít CHECK constraints, které zajistí, že do daného pole budou ukládána pouze data, která splňují určitá kritéria.
10. Vložení údajů do databáze, když je zadáno mezery do políčka firstName	Deterministic ká	<b>Major</b>	12	Přidat kontrolu, která zajistí, že všechny požadované údaje jsou přítomny před vložení do databáze. Pokud chybí jakýkoliv jiný povinný vstup, vrátí kód 400 (Bad Request).
11. Vložení údajů do databáze, když je zadáno číslice do políčka firstName	Deterministic ká	<b>Major</b>	13	V SQL je nutné omezit ukládání čísel mezer a spec symbolu do polí s textovým datovým typem pomocí kontrolních omezení (constraints) při vytváření tabulek. Konkrétně lze použít CHECK constraints, které zajistí, že do daného pole budou ukládána pouze data, která splňují určitá kritéria. Omezit maximální délku pro textové sloupce v SQL tabulek při pomoci atributu VARCHAR. Prakticky je důležité vyřešit a stanovit jasná kritéria pro vstupní informace. Bude se uvažovat o zavedení dvojitych jmen nebo příjmení apod? Podle toho stanovit maximální počet znaků, povolené znaky nebo mezery. Stejně tak pro všechna vstupní políčka.
12. Vložení údajů do databáze, když je zadán údaj délkou 256 symbolu do políčka firstName.	Deterministic ká	<b>Minor</b>	14	
13. Vrací se kód 500 při pokusu smazat data neplatného id studenta	Deterministic ká	<b>Major</b>	16	Namísto toho by měl být vrácen kód 404 (Not Found). Na straně serveru je třeba přidat validaci, která zkontroluje formát a strukturu ID před provedením dotazu do databáze.

## Bug report 1

**Název:** Vrací se kód 500 při zadání neplatného id studenta

**Závažnost:** Major

### Kroky k reprodukcí chyby:

1. Do requestu GET zadám <http://108.143.193.45:8080/api/v1/students/888>
2. Zmačknu tlačítko Send

### Očekávaný výsledek:

Vrací se stavový kod 404.

### Skutečný výsledek:

Vrací se kód 500

### Příloha:

The screenshot shows a REST client interface. At the top, a GET request is entered for the URL `http://108.143.193.45:8080/api/v1/students/888`. Below the URL bar, tabs for Params, Authorization, Headers (7), Body, Scripts, Tests, and Settings are visible. The 'Body' tab is selected, showing 'none' as the content type. A 'Send' button is on the right. Below the request area, a response section is shown with tabs for Body, Cookies, Headers (5), and Test Results. The 'Body' tab is active, displaying a JSON response. A status bar at the top of the response section indicates '500 Internal Server Error' with a response time of 240 ms and a size of 323 B. The JSON body contains the following data:

```
1 {
2   "timestamp": "2024-11-21T11:37:27.289+00:00",
3   "status": 500,
4   "error": "Internal Server Error",
5   "message": "",
6   "path": "/api/v1/students/888"
7 }
```

## Bug report 2

**Název:** Vrací se kód 500 při zadání mezer na místo id studenta

**Závažnost:** Major

### Kroky k reprodukcí chyby:

1. Do requestu GET zadám <http://108.143.193.45:8080/api/v1/students/>
2. Zmačknu tlačítko Send

### Očekávaný výsledek:

Vrací se stavový kód 400.

**Skutečný výsledek:** Vrací se kód 500

### Příloha:

The screenshot shows a REST client interface. At the top, a GET request is configured for the URL `http://108.143.193.45:8080/api/v1/students/`. Below the URL bar, tabs for Params, Authorization, Headers (7), Body, Scripts, Tests, and Settings are visible. The 'Body' tab is selected, showing a message: 'This request does not have a body'. Below this, radio buttons for content types are shown: `none` (selected), `form-data`, `x-www-form-urlencoded`, `raw`, `binary`, and `GraphQL`. A 'Send' button is on the right. Below the request configuration, the response is displayed. The status bar shows '500 Internal Server Error' in red, with a response time of 48 ms and a size of 326 B. The response body is shown in JSON format, displaying an error object with the following fields: `timestamp`, `status` (500), `error` (Internal Server Error), `message` (empty string), and `path` (the requested URL with encoded spaces).

```
{
  "timestamp": "2024-11-21T11:37:27.289+00:00",
  "status": 500,
  "error": "Internal Server Error",
  "message": "",
  "path": "/api/v1/students/%20%20"
}
```



## Bug report 3

**Název:** Příjmení se v databázi ukládá velkými písmeny.

**Závažnost:** Minor

**Kroky k reprodukcí chyby:**

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data  
"firstName": "Alexa",  
"lastName": "Doležal",  
"email": "a\_doležal@gmail.com",  
"age": 30
3. Zmačknu tlačítko Send

**Očekávaný výsledek:**

Vrací se stavový kód 200. Karta Body obsahuje JSON studenta, struktura kterého:

```
{  
  "id": _____,  
  "firstName": "Alexa",  
  "lastName": "Doležal",  
  "email": "a_doležal@gmail.com",  
  "age": 30  
}
```

**Skutečný výsledek:** Vrací se stavový kód 200, ale příjmení se v databázi ukládá velkými písmeny

```
{  
  "id": _____,  
  "firstName": "Alexa",  
  "lastName": "DOLEŽAL",  
  "email": "a_doležal@gmail.com",  
  "age": 30  
}
```

**Příloha:**

The screenshot shows a REST client interface. At the top, a POST request is configured to `http://108.143.193.45:8080/api/v1/students/`. The 'Body' tab is selected, and the request body is in 'raw' mode, containing the following JSON:

```
1 {  
2   "firstName": "Alexa",  
3   "lastName": "Doležal",  
4   "email": "a_doležal@gmail.com",  
5   "age": 30  
6 }
```

Below the request, the response is shown with a status of `200 OK`, a response time of `197 ms`, and a body size of `252 B`. The response body is displayed in 'Pretty' JSON format:

```
1 {  
2   "id": 2370,  
3   "firstName": "Alexa",  
4   "lastName": "DOLEŽAL",  
5   "email": "a_doležal@gmail.com",  
6   "age": 30  
7 }
```

## Bug report 4

**Název:** Vrací se kód 500 při vložení údajů do databáze, když není zadáno pouze firstName

**Závažnost:** Major

### Kroky k reprodukcí chyby:

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data  
"lastName": "Doležal",  
"email": "a\_doležal@gmail.com",  
"age": 30
3. Zmačknu tlačítko Send

### Očekávaný výsledek:

Vrací se stavový kód 400.

**Skutečný výsledek:** Vrací se kód 500.

### Příloha:

The screenshot displays a REST client interface. At the top, a POST request is configured to the URL `http://108.143.193.45:8080/api/v1/students/`. The request body is in raw JSON format, containing the following data: `{ "lastName": "Doležal", "email": "a_doležal@gmail.com", "age": 30 }`. Below the request configuration, the response is shown. It is a **500 Internal Server Error** with a status of 500, a timestamp of `"2024-11-21T11:28:19.950+00:00"`, and an error message of `"Internal Server Error"`. The response body is also in JSON format: `{ "timestamp": "2024-11-21T11:28:19.950+00:00", "status": 500, "error": "Internal Server Error", "message": "", "path": "/api/v1/students/" }`.

## Bug report 5

**Název:** Vrací se kód 500 při vložení údajů do databáze, když není zadáno pouze lastName

**Závažnost:** Major

### Kroky k reprodukcí chyby:

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data  
"firstName": "Alexa",  
"email": "a\_doležal@gmail.com",  
"age": 30
3. Zmačknu tlačítko Send

### Očekávaný výsledek:

Vrací se stavový kód 400.

**Skutečný výsledek:** Vrací se kód 500.

### Příloha:

The screenshot shows a REST client interface. At the top, a POST request is configured for the URL `http://108.143.193.45:8080/api/v1/students/`. The request body is in raw JSON format, containing the following data:

```
1 {
2   "firstName": "Alexa",
3   "email": "a_doležal@gmail.com",
4   "age": 30
5 }
```

Below the request, the response is displayed. It is a 500 Internal Server Error with a status of 500. The response body is in raw JSON format, containing the following data:

```
1 {
2   "timestamp": "2024-11-21T11:30:22.251+00:00",
3   "status": 500,
4   "error": "Internal Server Error",
5   "message": "",
6   "path": "/api/v1/students/"
7 }
```

## Bug report 6

**Název:** Vrací se kód 500 při vložení údajů do databáze, když není zadán pouze email

**Závažnost:** Major

### Kroky k reprodukcí chyby:

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data  
"firstName": "Alexa",  
"lastName": "Doležal",  
"age": 30
3. Zmačnu tlačítko Send

**Očekávaný výsledek:** Vrací se stavový kód 400.

**Skutečný výsledek:** Vrací se kód 500.

**Příloha:**

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

- Method:** POST
- URL:** <http://108.143.193.45:8080/api/v1/students/>
- Body (raw):**

```
1 {  
2   "firstName": "Alexa",  
3   "lastName": "Doležal",  
4   "age": 30  
5 }
```
- Response:** 500 Internal Server Error (204 ms, 284 B)
- Response Body (Pretty):**

```
1 {  
2   "timestamp": "2024-11-21T11:34:44.202+00:00",  
3   "status": 500,  
4   "error": "Internal Server Error",  
5   "message": "",  
6   "path": "/api/v1/students/"  
7 }
```

## Bug report 7

**Název:** Vrací se kód 500 při vložení údajů do databáze, když není zadán věk

**Závažnost:** Major

### Kroky k reprodukci chyby:

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data  
"firstName": "Alexa",  
"lastName": "Doležal",  
"email": "a\_doležal@gmail.com"
3. Zmačknu tlačítko Send

**Očekávaný výsledek:** Vrací se stavový kód 400.

**Skutečný výsledek:** Vrací se kód 500.

**Příloha:**

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

- Request:**
  - Method: POST
  - URL: <http://108.143.193.45:8080/api/v1/students/>
  - Body (raw):

```
1 {  
2   "firstName": "Alexa",  
3   "lastName": "Doležal",  
4   "email": "a_doležal@gmail.com"  
5 }
```
- Response:**
  - Status: 500 Internal Server Error
  - Time: 403 ms
  - Size: 284 B
  - Body (pretty):

```
1 {  
2   "timestamp": "2024-11-21T11:37:27.289+00:00",  
3   "status": 500,  
4   "error": "Internal Server Error",  
5   "message": "",  
6   "path": "/api/v1/students/"  
7 }
```

## Bug report 8

**Název:** Vrací se kód 500 při vložení údajů do databáze, když nejsou zadány žádné údaje

**Závažnost:** Major

### Kroky k reprodukcí chyby:

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw ne zadám testovací data
3. Zmačknou tlačítko Send

**Očekávaný výsledek:** Vrací se stavový kód 400.

**Skutečný výsledek:** Vrací se kód 500.

**Příloha:**

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

- Method:** POST
- URL:** <http://108.143.193.45:8080/api/v1/students/>
- Body:** Empty (raw format)
- Status:** 500 Internal Server Error (68 ms, 284 B)
- Response Body (JSON):**

```
1 {
2   "timestamp": "2024-11-21T11:40:55.740+00:00",
3   "status": 500,
4   "error": "Internal Server Error",
5   "message": "",
6   "path": "/api/v1/students/"
7 }
```

## Bug report 9

**Název:** Vložení údajů do databáze, když je zadáno speciální symboly do políčka firstName  
**Závažnost:** Major

### Kroky k reprodukcí chyby:

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data

```
"firstName": "@#$%^&*()",  
"lastName": "Doležal",  
"email": "a_doležal@gmail.com",  
"age": 30
```

3. Zmačknu tlačítko Send

### Očekávaný výsledek:

Vrací se stavový kód 400.

**Skutečný výsledek:** Vrací se stavový kód 200.

### Příloha:

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

- Method:** POST
- URL:** <http://108.143.193.45:8080/api/v1/students/>
- Body (raw):**

```
"firstName": "@#$%^&*()",  
"lastName": "Doležal",  
"email": "a_doležal@gmail.com",  
"age": 30
```
- Status:** 200 OK
- Response (Pretty):**

```
{  
  "id": 2380,  
  "firstName": "@#$%^&*()",  
  "lastName": "DOLEŽAL",  
  "email": "a_doležal@gmail.com",  
  "age": 30  
}
```

## Bug report 10

**Název:** Vložení údajů do databáze, když je zadáno mezery do políčka firstName

**Závažnost:** Major

### Kroky k reprodukcí chyby:

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data  
"firstName": " ",  
"lastName": "Doležal",  
"email": "a\_doležal@gmail.com",  
"age": 30
3. Zmačknu tlačítko Send

**Očekávaný výsledek:** Vrací se stavový kód 400.

**Skutečný výsledek:** Vrací se stavový kód 200.

### Příloha:

The screenshot displays a REST client interface. At the top, a POST request is configured to `http://108.143.193.45:8080/api/v1/students/`. The 'Body' tab is selected, showing raw JSON data: `{ "firstName": " ", "lastName": "Doležal", "email": "a_doležal@gmail.com", "age": 30 }`. Below the request, the response is shown with a status of `200 OK`, a response time of `216 ms`, and a size of `253 B`. The response body is formatted as JSON: `{ "id": 2378, "firstName": " ", "lastName": "DOLEŽAL", "email": "a_doležal@gmail.com", "age": 30 }`.



## Bug report 11

**Název:** Vložení údajů do databáze, když je zadáno číslíčko do políčka firstName

**Závažnost:** Major

### Kroky k reprodukcí chyby:

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data

```
"firstName": "12345",  
"lastName": "Doležal",  
"email": "a_doležal@gmail.com",  
"age": 30
```

3. Zmačknu tlačítko Send

### Očekávaný výsledek:

Vrací se stavový kód 400.

**Skutečný výsledek:** Vrací se stavový kód 200.

### Příloha:

The screenshot shows a REST client interface with a POST request to `http://108.143.193.45:8080/api/v1/students/`. The request body is raw JSON: `{ "firstName": "12345", "lastName": "Doležal", "email": "a_doležal@gmail.com", "age": 30 }`. The response is a 200 OK status with a response time of 328 ms and a body size of 257 B. The response body is also raw JSON: `{ "id": 2379, "firstName": "12345", "lastName": "DOLEŽAL", "email": "a_doležal@gmail.com", "age": 30 }`.

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

{
  "firstName": "12345",
  "lastName": "Doležal",
  "email": "a_doležal@gmail.com",
  "age": 30
}
```

200 OK • 328 ms • 257 B

```
{
  "id": 2379,
  "firstName": "12345",
  "lastName": "DOLEŽAL",
  "email": "a_doležal@gmail.com",
  "age": 30
}
```

## Bug report 12

**Název:** Vložení údajů do databáze, když je zadáno údaj délkou 256 symbolu do políčka firstName.

**Závažnost: Minor**

### Kroky k reprodukcii chyby:

1. Do requestu POST zadám <http://108.143.193.45:8080/api/v1/students/>
2. Do políčka raw zadám testovací data

```
"firstName": "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa  
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa  
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa  
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa",  
  "lastName": "Doležal",  
  "email": "a_dolezal@gmail.com",  
  "age": 30
```

3. Zmačknu tlačítko Send

**Očekávaný výsledek:** Vrací se stavový kód 400.

**Skutečný výsledek:** Vrací se stavový kód 200.

## Příloha:

[illegible]

## Bug report 13

**Název:** Vrací se kód 500 při pokusu smazat data neplatného id studenta

**Závažnost:** Major

### Kroky k reprodukcí chyby:

1. Do requestu DELETE zadám <http://108.143.193.45:8080/api/v1/students/888>
2. Zmačknu tlačítko Send

**Očekávaný výsledek:** Vrací se stavový kód 404.

**Skutečný výsledek:** Vrací se kód 500.

### Příloha:

The screenshot shows a REST client interface. At the top, a DELETE request is configured for the URL `http://108.143.193.45:8080/api/v1/students/888`. The 'Body' tab is selected, and it shows the message 'This request does not have a body'. Below the request configuration, the response is displayed. The status bar indicates a '500 Internal Server Error' with a response time of 128 ms and a size of 287 B. The response body is shown in JSON format, displaying an error object with the following fields: `timestamp`, `status` (500), `error` (Internal Server Error), `message` (empty string), and `path` (`/api/v1/students/888`).

```
{
  "timestamp": "2024-11-23T11:35:08.014+00:00",
  "status": 500,
  "error": "Internal Server Error",
  "message": "",
  "path": "/api/v1/students/888"
}
```