

# FINÁLNÍ PROJEKT

## č.1



# ENGETO

Autor: Ing. Daniel Jarnot  
Datum: 14.4.2024  
discord: amik4806

## OBSAH

ZADÁNÍ.....	3
Přístupové údaje:.....	3
Poznámky:.....	3
TESTOVACÍ SCÉNÁŘE.....	4
EXEKUCE TESTŮ.....	13
BUG REPORT.....	18

# ZADÁNÍ

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

## Přístupové údaje:

Databáze	database: qa_demo Host: aws.connect.psdb.cloud Username: 68q0llcbuyijdt5mzq2 Password: pscale_pw_zxfenAXfSYx9loe9PCS1snuGKr Fzw2J84BMIVZI96o
REST-API	http://108.143.193.45:8080/api/v1/students/

## Poznámky:

Nezapomeňte, že v IT se data musí někde uložit a poté získat. Proto ověřte, že data jsou správně uložena a získávána z databáze.

Nezapomeňte do testovacích scénářů uvést testovací data, očekávaný výsledek včetně těla odpovědi a stavových kódů.

# TESTOVACÍ SCÉNÁŘE

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

Test case	1.0
Datum	14.4.2024
Abstract	Použití metody GET u existujícího studenta v databázi „student“
Verze	v01
Zdroj	Databáze „student“
Aplikace	Postman, MySQL Workbench
Kroky	<ol style="list-style-type: none"><li>1. Otevři aplikaci Postman a MySQL Workbench</li><li>2. V MySQL workbench najdi existujícího studenta s id 210.</li><li>3. V aplikaci Postman vyber metodu GET a vlož adresu: <code>http://108.143.193.45:8080/api/v1/students/210</code></li><li>4. Klikni na tlačítko „Send“</li><li>5. V aplikaci Postman ověř, že údaje studenta s id 210 se shodují s údaji v databázi MySQL Workbench.</li></ol>
Sktutečný výsledek	<pre>{   "id": 210,   "firstName": "Anabela",   "lastName": "LARSON",   "email": "ana.larson@xxmail.com",   "age": 25 }</pre> <p>Status: 200 OK</p>
Očekávaný výsledek	<pre>{   "id": 210,   "firstName": "Anabela",   "lastName": "LARSON",   "email": "ana.larson@xxmail.com",   "age": 25 }</pre> <p>Status: 200 OK</p>

Test case	2.0
Datum	14.4.2024
Abstract	Použití metody GET u neexistujícího studenta v databázi „student“
Verze	v01
Zdroj	Databáze „student“
Aplikace	Postman, MySQL Workbench
Kroky	<ol style="list-style-type: none"> <li>1. Otevři aplikaci Postman a MySQL Workbench</li> <li>2. V MySQL workbench zjisti, které číslo id není použito. V tomto případě id 177.</li> <li>3. V aplikaci Postman vyber metodu GET a vlož adresu: <code>http://108.143.193.45:8080/api/v1/students/177</code></li> <li>4. Klikni na tlačítko „Send“</li> <li>5. V aplikaci Postman ověř, že nelze zobrazit údaje studenta pro id 177.</li> </ol>
Skutečný výsledek	<pre>{   "timestamp": "2024-04-14T10:10:02.827+00:00",   "status": 500,   "error": "Internal Server Error",   "message": "",   "path": "/api/v1/students/177" }</pre> <p>Status: 500 Internal Server Error</p>
Očekávaný výsledek	<pre>{   "timestamp": "2024-04-14T10:10:02.827+00:00",   "status": 500,   "error": "Internal Server Error",   "message": "",   "path": "/api/v1/students/177" }</pre> <p>Status: 500 Internal Server Error</p>

Test case	3.0
Datum	14.4.2024
Abstract	Vytvoření nového studenta pomocí metody POST.
Verze	v01
Zdroj	Databáze „student“
Aplikace	Postman, MySQL Workbench
Kroky	<ol style="list-style-type: none"> <li>1. Otevři aplikaci Postman a MySQL Workbench</li> <li>2. V aplikaci Postman vyber metodu POST a vlož adresu: <a href="http://108.143.193.45:8080/api/v1/students/">http://108.143.193.45:8080/api/v1/students/</a></li> <li>3. V aplikaci Postman do záložky Body vlož následující text: <pre>{ "firstName": "Daniel", "lastName": "Jarnot", "age": 33, "email": "dj90@gmail.com" }</pre> </li> <li>4. Klikni na tlačítko „Send“</li> <li>5. V aplikaci Postman zkontroluj, zda Status se rovná 200 OK.</li> <li>6. V aplikaci MySQL Workbench ověř, že se vytvořil nový záznam, dle zadaných údajů.</li> </ol>
Sktutečný výsledek	<pre>{ "id": 554, "firstName": "Daniel", "lastName": "JARNOT", "email": "dj90@gmail.com", "age": 33 }</pre> <p>Status: 200 OK</p>
Očekávaný výsledek	<pre>{ "id": 554, "firstName": "Daniel", "lastName": "JARNOT", "email": "dj90@gmail.com", "age": 33 }</pre> <p>Status: 200 OK</p>

Test case	4.0
Datum	14.4.2024
Abstract	Vytvoření duplicitního studenta pomocí metody POST.
Verze	v01
Zdroj	Databáze „student“
Aplikace	Postman, MySQL Workbench
Kroky	<ol style="list-style-type: none"> <li>1. Otevři aplikaci Postman a MySQL Workbench</li> <li>2. V aplikaci MySQL Workbench si vyber již existující záznam.</li> <li>3. V aplikaci Postman vyber metodu POST a vlož adresu: <a href="http://108.143.193.45:8080/api/v1/students/">http://108.143.193.45:8080/api/v1/students/</a></li> <li>3.V aplikaci Postman do záložky Body vlož následující text. Tzn. Již existující údaje: <pre>{ "firstName": "Daniel", "lastName": "Jarnot", "age":33, "email": "dj90@gmail.com" }</pre> </li> <li>4. Klikni na tlačítko „Send“</li> <li>5. V aplikaci Postman zkontroluj, zda Status se rovná 400 OK.</li> <li>6. V aplikaci MySQL Workbench ověř, že se nevytvořil duplicitní záznam.</li> </ol>
Sktutečný výsledek	<pre>{ "id": 556, "firstName": "Daniel", "lastName": "JARNOT", "email": "dj90@gmail.com", "age": 33 }</pre> <p>Status: 200 OK</p>
Očekávaný výsledek	Status: 400 Bad Request

Test case	5.0
Datum	14.4.2024
Abstract	Psaní velkých a malých písmen
Verze	v01
Zdroj	Databáze „student“
Aplikace	Postman, MySQL Workbench
Kroky	<ol style="list-style-type: none"> <li>1. Otevři aplikaci Postman a MySQL Workbench</li> <li>2. V aplikaci Postman vyber metodu POST a vlož adresu: <a href="http://108.143.193.45:8080/api/v1/students/">http://108.143.193.45:8080/api/v1/students/</a></li> <li>3. V aplikaci Postman do záložky Body vlož následující text.   <pre>{   "firstName": "Michal",   "lastName": "RYBÍZEK",   "age": 18,   "email": "mryb22@seznam.cz", }</pre> </li> <li>4. Klikni na tlačítko „Send“</li> <li>5. V aplikaci Postman zkontroluj, zda Status 200 OK.</li> <li>6. V aplikaci MySQL Workbench ověř, že se vytvořil nový údaj.</li> </ol>
Sktutečný výsledek	<pre>{   „id“: 557   "firstName": "Michal",   "lastName": "RYBÍZEK",   "email": "<a href="mailto:mryb22@seznam.cz">mryb22@seznam.cz</a>",   "age": 18, }</pre>
Očekávaný výsledek	<pre>{   „id“: 557   "firstName": "Michal",   "lastName": "Rybízek",   "email": "<a href="mailto:mryb22@seznam.cz">mryb22@seznam.cz</a>",   "age": 18, }</pre>



Test case	6.0
Datum	14.4.2024
Abstract	Psaní čísel a písmen do pole „age“
Verze	v01
Zdroj	Databáze „student“
Aplikace	Postman, MySQL Workbench
Kroky	<ol style="list-style-type: none"> <li>1. Otevři aplikaci Postman a MySQL Workbench</li> <li>2. V aplikaci Postman vyber metodu POST a vlož adresu: <a href="http://108.143.193.45:8080/api/v1/students/">http://108.143.193.45:8080/api/v1/students/</a></li> <li>3. V aplikaci Postman do záložky Body vlož následující text. <pre>{   "firstName": "Klaudie",   "lastName": "Suchánková",   "age": 41a,   "email": "ksuch@seznam.cz", }</pre> </li> <li>3. Klikni na tlačítko „Send“</li> <li>4. V aplikaci Postman zkontroluj, zda Status se rovná 400 Bad Request.</li> <li>5. V aplikaci MySQL Workbench ověř, že se nevytvořil nový údaj.</li> </ol>
Sktutečný výsledek	<pre>{   "timestamp": "2024-04-14T15:39:11.881+00:00",   "status": 400,   "error": "Bad Request",   "message": "",   "path": "/api/v1/students/" }</pre> Status: 400 Bad Request
Očekávaný výsledek	<pre>{   "timestamp": "2024-04-14T15:39:11.881+00:00",   "status": 400,   "error": "Bad Request",   "message": "",   "path": "/api/v1/students/" }</pre> Status: 400 Bad Request

Test case	7.0
Datum	14.4.2024
Abstract	Adresa emailu bez znaku: @
Verze	v01
Zdroj	Databáze „student“
Aplikace	Postman, MySQL Workbench
Kroky	<ol style="list-style-type: none"> <li>1. Otevři aplikaci Postman a MySQL Workbench</li> <li>2. V aplikaci Postman vyber metodu POST a vlož adresu: <a href="http://108.143.193.45:8080/api/v1/students/">http://108.143.193.45:8080/api/v1/students/</a></li> <li>3. V aplikaci Postman do záložky Body vlož následující text. <pre>{   "firstName": "Martin",   "lastName": "Jelito",   "age": 49,   "email": "mejelseznam.cz", }</pre> </li> <li>3. Klikni na tlačítko „Send“</li> <li>4. V aplikaci Postman zkontroluj, zda Status se rovná 400 Bad Request.</li> <li>5. V aplikaci MySQL Workbench ověř, že se nevytvořil nový údaj.</li> </ol>
Sktutečný výsledek	<pre>{   „id: 568,   "firstName": "Martin",   "lastName": "Jelito",   "email": "mejelseznam.cz",   "age": 49, }</pre>
Očekávaný výsledek	<pre>{   "timestamp": "2024-04-14T15:39:11.881+00:00",   "status": 400,   "error": "Bad Request",   "message": "",   "path": "/api/v1/students/" }</pre> <p>Status: 400 Bad Request</p>

Test case	8.0
Datum	14.4.2024
Abstract	Psaní čísel a speciálních znaků do pole „firstName“ a „lastName“
Verze	v01
Zdroj	Databáze „student“
Aplikace	Postman, MySQL Workbench
Kroky	<p>1. Otevři aplikaci Postman a MySQL Workbench</p> <p>2. V aplikaci Postman vyber metodu POST a vlož adresu:  <a href="http://108.143.193.45:8080/api/v1/students/">http://108.143.193.45:8080/api/v1/students/</a></p> <p>3. V aplikaci Postman do záložky Body vlož následující text.</p> <pre>{   "firstName": "kamil9@",   "lastName": "7Jelito@7",   "age": 89,   "email": "kamjel7@seznam.cz", }</pre> <p>3. Klikni na tlačítko „Send“</p> <p>4. V aplikaci Postman zkontroluj, zda Status se rovná 400 Bad Request.</p> <p>5. V aplikaci MySQL Workbench ověř, že se nevytvořil nový údaj.</p>
Sktutečný výsledek	<pre>{   „id: 563,   "firstName": "kamil9@",   "lastName": "7Jelito@7",   "email": "kamjel7@seznam.cz",   "age": 89, }</pre> <p>Status: 200 OK</p>
Očekávaný výsledek	<pre>{   "timestamp": "2024-04-14T15:39:11.881+00:00",   "status": 400,   "error": "Bad Request",   "message": "",   "path": "/api/v1/students/" }</pre> <p>Status: 400 Bad Request</p>

Test case	9.0
Datum	14.4.2024
Abstract	Ostranění záznamu metodou DELETE
Verze	v01
Zdroj	Databáze „student“
Aplikace	Postman, MySQL Workbench
Kroky	<ol style="list-style-type: none"> <li>1. Otevři aplikaci Postman a MySQL Workbench</li> <li>2. V aplikaci MySQL Workbench najdi studenta s číslem id:491</li> <li>3. V aplikaci Postman vyber metodu DELETE a vlož adresu: <a href="http://108.143.193.45:8080/api/v1/students/491">http://108.143.193.45:8080/api/v1/students/491</a></li> <li>3. Klikni na tlačítko „Send“</li> <li>4. V aplikaci MySQL Workbench ověř, že se odstranil student s id: 491.</li> <li>4. V aplikaci Postman zkontroluj, zda Status se rovná 200 OK.</li> </ol>
Sktutečný výsledek	Status: 200 OK
Očekávaný výsledek	Status: 200 OK

Test case	10.0
Datum	14.4.2024
Abstract	Ostranění záznamu, který neexistuje
Verze	v01
Zdroj	Databáze „student“
Aplikace	Postman, MySQL Workbench
Kroky	<ol style="list-style-type: none"> <li>1. Otevři aplikaci Postman a MySQL Workbench</li> <li>2. V aplikaci MySQL Workbench v databázi „student“ si vyber číslo id, které nebylo použito.</li> <li>3. V aplikaci Postman vyber metodu DELETE a vlož adresu: <a href="http://108.143.193.45:8080/api/v1/students/491">http://108.143.193.45:8080/api/v1/students/491</a></li> <li>3. Klikni na tlačítko „Send“</li> <li>4. V aplikaci Postman zkontroluj, zda Status se rovná 500 Internal Error</li> </ol>
Sktutečný výsledek	<pre>{   "timestamp": "2024-04-14T16:06:37.559+00:00",   "status": 500,   "error": "Internal Server Error",   "message": "",   "path": "/api/v1/students/491" }</pre> Status: 500 Internal Error
Očekávaný výsledek	<pre>{   "timestamp": "2024-04-14T16:06:37.559+00:00",   "status": 500,   "error": "Internal Server Error",   "message": "",   "path": "/api/v1/students/491" }</pre> Status: 500 Internal Error

# EXEKUCE TESTŮ

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

Test case 1.0: Použití metody GET u existujícího studenta v databázi „student“

http://108.143.193.45:8080/api/v1/students/197

GEThttp://108.143.193.45:8080/api/v1/students/210Send

ParamsAuthorizationHeaders (6)BodyPre-request ScriptTestsSettingsCookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

BodyCookiesHeaders (5)Test ResultsStatus: 200 OKTime: 159 msSize: 257 BSave as example

PrettyRawPreviewVisualizeJSON

```
1 {
2   "id": 210,
3   "firstName": "Anabela",
4   "lastName": "LARSON",
5   "email": "ana.larson@xxmail.com",
6   "age": 25
7 }
```

209	34	John	JOSHUA
210	25	ana.larson@xxmail.com	Anabela LARSON
211	34	jjohn34@yourmail.com	JOSHUA
213	34	John	JOSHUA

Test case 2.0: Použití metody GET u neexistujícího studenta v databázi „student“

HTTP client interface showing a GET request to `http://108.143.193.45:8080/api/v1/students/177`. The response status is **500 Internal Server Error**. The response body (JSON) is:

```
{
  "timestamp": "2024-04-14T10:10:02.827+00:00",
  "status": 500,
  "error": "Internal Server Error",
  "message": "",
  "path": "/api/v1/students/177"
}
```

171	34	jjohn34@yourmail.com	John	234
175	34	ayrton@senna.com		SENN
176	34	jjohn34@yourmail.com	john	JOSHUA
180	0	jwick@jwick.com	John	WICK
181	0	jwick@jwick.com	JOHN	WICK

Test case 3.0: Vytvoření nového studenta pomocí metody POST.

HTTP client interface showing a POST request to `http://108.143.193.45:8080/api/v1/students/`. The request body (JSON) is:

```
{
  "firstName": "Daniel",
  "lastName": "Jarnot",
  "age": 33,
  "email": "dj90@gmail.com"
}
```

The response status is **200 OK**. The response body (JSON) is:

```
{
  "id": 554,
  "firstName": "Daniel",
  "lastName": "JARNOT",
  "email": "dj90@gmail.com",
  "age": 33
}
```

553	56	alicestark.com	Alice	STARK
554	33	dj90@gmail.com	Daniel	JARNOT
NULL	NULL	NULL	NULL	NULL

## Test case 4.0: Vytvoření duplicitního studenta pomocí metody POST.

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 a JSON object: `{ "firstName": "Daniel", "lastName": "Jarnot", "age": 33, "email": "dj90@gmail.com" }`. The response status is 200 OK, with a time of 216 ms and a size of 249 B. The response body is a JSON object: `{ "id": 556, "firstName": "Daniel", "lastName": "JARNOT", "email": "dj90@gmail.com", "age": 33 }`.

554	33	dj90@gmail.com	Daniel	JARNOT
555	19	t23434@yourmail.com	Jan	NOVAK
556	33	dj90@gmail.com	Daniel	JARNOT
HULL	HULL	HULL	HULL	HULL

## Test case 5.0: Psaní velkých a malých písmen

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 a JSON object: `{ "firstName": "Michal", "lastName": "Rybízek", "age": 18, "email": "mryb22@seznam.cz" }`. The response status is 200 OK, with a time of 188 ms and a size of 253 B. The response body is a JSON object: `{ "id": 557, "firstName": "Michal", "lastName": "RYBÍZEK", "email": "mryb22@seznam.cz", "age": 18 }`.

555	19	t23434@yourmail.com	Jan	NOVAK
556	33	dj90@gmail.com	Daniel	JARNOT
557	18	mryb22@seznam.cz	Michal	RYBÍZEK

## Test case 6.0: Psaní čísel a písmen do pole „age“

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 a JSON object: `{ "firstName": "Klaudie", "lastName": "Suchánková", "age": "41a", "email": "ksuch@seznam.cz" }`. The response status is `400 Bad Request` with a message: `{ "timestamp": "2024-04-14T15:39:11.881+00:00", "status": 400, "error": "Bad Request", "message": "", "path": "/api/v1/students/" }`.

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

{
  "firstName": "Klaudie",
  "lastName": "Suchánková",
  "age": "41a",
  "email": "ksuch@seznam.cz"
}
```

```
{
  "timestamp": "2024-04-14T15:39:11.881+00:00",
  "status": 400,
  "error": "Bad Request",
  "message": "",
  "path": "/api/v1/students/"
}
```

## Test case 7.0: Adresa emailu bez znaku: @

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 a JSON object: `{ "firstName": "Martin", "lastName": "Jelito", "age": 49, "email": "mejelseznam.cz" }`. The response status is `200 OK` with a message: `{ "id": 560, "firstName": "Martin", "lastName": "JELITO", "email": "mejelseznam.cz", "age": 49 }`.

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

{
  "firstName": "Martin",
  "lastName": "Jelito",
  "age": 49,
  "email": "mejelseznam.cz"
}
```

```
{
  "id": 560,
  "firstName": "Martin",
  "lastName": "JELITO",
  "email": "mejelseznam.cz",
  "age": 49
}
```



## Test case 8.0: Psaní čísel a speciálních znaků do pole „firstName“ a „lastName“

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

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

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

```

1 {
2   "firstName": "kamil9@",
3   "lastName": "7Jelito@7",
4   "age": 89,
5   "email": "kamjel7@seznam.cz"
6 }

```

Body Cookies Headers (5) Test Results Status: 200 OK Time: 157 ms Size: 256 B Save as example ...

Pretty Raw Preview Visualize JSON ▼ ↺

```

1 {
2   "id": 563,
3   "firstName": "kamil9@",
4   "lastName": "7JELITO@7",
5   "email": "kamjel7@seznam.cz",
6   "age": 89
7 }

```

559	18	mryp22@seznam.cz	Michal	RYBIZEK55
560	49	mejelseznam.cz	Martin	JELITO
563	89	kamjel7@seznam.cz	kamil9@	7JELITO@7
* NULL	NULL	NULL	NULL	NULL

## Test case 9.0: Ostranění záznamu metodou DELETE

DELETE ▼ http://108.143.193.45:8080/api/v1/students/491 Send ▼

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

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers Test Results Status: 200 OK Time: 130 ms Size: 123 B Save as example ...

Pretty Raw Preview Visualize Text ▼ ↺

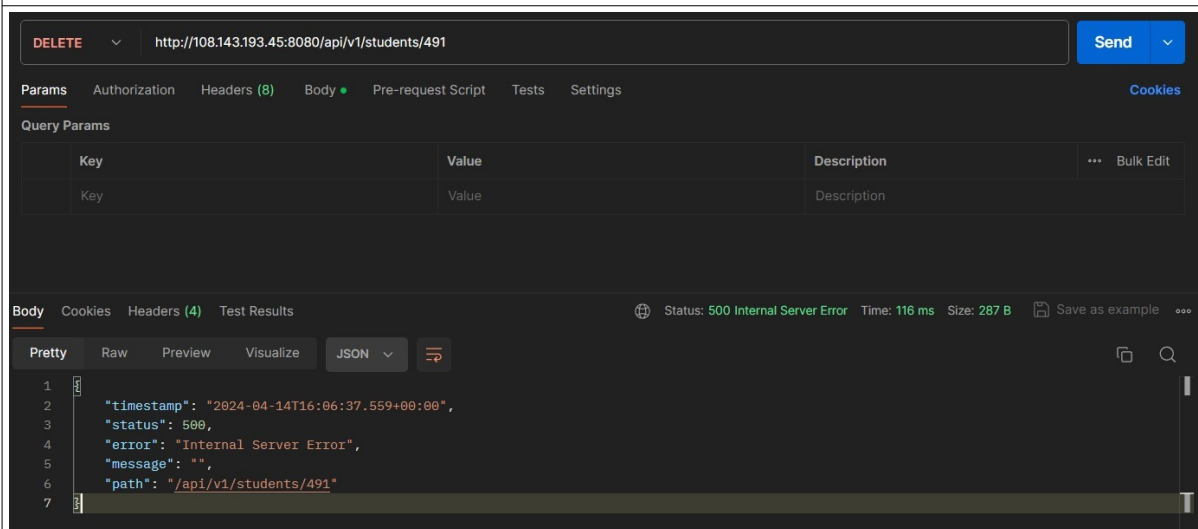
```

1

```

489	56	dfgdg4@yourmail.com	Petr	PAVEL
490	100	nemam	Bilbo	PYTLÍK
492	27	ekocourek@yourmail.com	Eva	KOCOUREK
493	31	t23264@yourmail.com	Vlado	NOVAK

## Test case 10.0: Ostranění záznamu, který neexistuje



## BUG REPORT

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

Bug report	1.0
Test case	4.0
Datum	14.4.2024
Abstract	API umožní vytvořit duplicitní záznam s jiným id číslem.
Očekávaný výsledek	Mělo by být zakázáno vytváření záznamů se stejnou emailovou adresou.

Bug report	2.0
Test case	5.0
Datum	14.4.2024
Abstract	Pole „lastName“ – uloží se do databáze vždy velkými písmeny.
Očekávaný výsledek	Pokud do pole „lastName“ napíšu jméno malými písmeny, tak by se do databáze mělo uložit ve stejném formátu.

Bug report	3.0
Test case	7.0
Datum	14.4.2024
Abstract	Lze psát emailovou adresu bez znaku @.
Očekávaný výsledek	Pokud v emailové adrese bude chybět zavináč, pak by se měl spustit Status: 400 Bad Request

Bug report	4.0
Test case	8.0
Datum	14.4.2024
Abstract	Lze psát čísla a speciální znaky do pole „firstName“ a „lastName“
Očekávaný výsledek	Čísla a speciální znaky by měly být zakázány v poli „firstName“ a „lastName“. Měl by se spustit Status: 400 Bad Request