

FINÁLNÍ PROJEKT

č.1



ENGETO

Autor: Lenka Michalčáková
Datum: 4. 11. 2024

OBSAH

ZADÁNÍ	3
TESTOVACÍ SCÉNÁŘE	4
EXEKUCE TESTŮ	5
BUG REPORT	5

ZADÁNÍ

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

Přístupové údaje:

Databáze	Default scheme: qa_demo Host: aws.connect.psdb.cloud Port: 3306
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 ověřila funkčnost aplikace.

1. Metoda GET zobrazí data o existujícím studentovi

Testing data: `http://108.143.193.45:8080/api/v1/students/365`

Steps to reproduce:

1. Ve Workbench si pomocí `SELECT * FROM student;` zobrazím všechna data z tabulky.
2. Vyberu si libovolného studenta, např. `id = 365`, `age = 19`, `email = alfred@mail.com`, `first_name = alfred`, `last_name = HITCHCOCK`
3. V aplikaci Postman zavolám metodou GET údaje o studentovi s `id = 365`

Expected result:

Aplikace Postman mi vrátí odpověď se stavovým kódem 200 OK a v těle odpovědi budou uvedena všechna pravdivá data, která jsou o studentovi uvedena v databázi.

2. Metoda GET zobrazí chybový stavový kód pro žádost o data o neexistujícím studentovi

Testing data: `http://108.143.193.45:8080/api/v1/students/99999`

`http://108.143.193.45:8080/api/v1/students/0000000`

`http://108.143.193.45:8080/api/v1/students/14086579`

`http://108.143.193.45:8080/api/v1/students/2378545098765435678987654345678987654345678987654345678876545678987654`

`http://108.143.193.45:8080/api/v1/students/-2345678`

`http://108.143.193.45:8080/api/v1/students/1111`

Steps to reproduce:

1. V Postman zadávám pomocí metody GET různé číselné požadavky pro získání dat o neexistujících studentech viz. Testing data.

Expected result:

Aplikace Postman mi vrátí odpověď se stavovým kódem 404 Not Found nebo 400 Bad Request a v těle odpovědi budou prázdná data.

3. Metoda GET příkaz `http://108.143.193.45:8080/api/v1/students/` vrátí 200 OK a záznamy všech studentů v DTB

Testing data: `http://108.143.193.45:8080/api/v1/students/`

Steps to reproduce:

1. V Postman zadám pomocí metody GET požadavek pro získání dat o všech studentech uložených v DTB - `http://108.143.193.45:8080/api/v1/students/`

Expected result:

Aplikace Postman mi vrátí odpověď se stavovým kódem 200 OK a v těle odpovědi budou data všech studentů z DTB.

4. Při zadání chybných vstupů vrátí GET chybovou hlášku ve stavovém kódu

Testing data: `http://108.143.193.45:8080/api/v1/abc`

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

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

`http://108.143.193.45:8080/api/v1/students/fnhtukioýturtreawqaSDCFVGBHJUKI
ÁÝTŽRTERESAFVGBHJZTRETFSDCDVBGHNBMZKUÝŽŘRZES`

Steps to reproduce:

1. V Postman zadám pomocí metody GET požadavek pro získání dat o studentovi s id = abc, id = petrzel, id = fgrhtgefgehbtrwbfvdf id = fnhtukioýturtreawqaSDCFVGBHJUKIÁÝTŽRTERESAFVGBHJZTRETFSDCDVBGHNBMZKUÝŽŘRZES

Expected result:

Aplikace Postman mi vrátí odpověď se stavovým kódem 400 nebo 404 nikoliv 500.

5. Metoda GET vrátí vždy všechny položky z dat o studentovi (id, first name, last name, email, age)

Testing data: <http://108.143.193.45:8080/api/v1/students/>
<http://108.143.193.45:8080/api/v1/students/354>
<http://108.143.193.45:8080/api/v1/students/368>
<http://108.143.193.45:8080/api/v1/students/513>

Steps to reproduce:

1. V Postman zadám pomocí metody GET požadavek pro získání dat o všech studentech a následně o konkrétních studentech s různými id viz Testing data,

Expected result:

V těle odpovědi budou data o studentech vždy obsahovat položky id, first name, last name, email a age.

6. Metoda POST vytvoří v databázi studenta se zadanými parametry

Testing data: {

```
"firstName": "jmeno",  
"lastName": "primeni",  
"email": "emailova_adresa",  
"age": 25  
}  
  
{  
  "firstName": "lumira",  
  "lastName": "svetlova",  
  "email": "lumira@svetlova.cz",  
  "age": 15  
}
```

Steps to reproduce:

1. V Postman odešlu pomocí metody POST data o studentovi (viz testovací data).
2. Ve Workbench zkontroluju pomocí `SELECT * FROM student WHERE last_name = 'primeni'`; zda v databázi existuje student se zadanými parametry.

Expected result:

Postman mi vrátí 201 Created a v těle odpovědi budou zadané parametry nově vytvořeného studenta. Při kontrole ve Workbench najdu vytvořeného studenta se zadanými parametry.

7. Metoda POST nevytvoří studenta se špatně zadanými parametry a vrátí chybu ve stavovém kódu

Testing data:

```
{
  "firstName": 345369,
  "lastName": "ýřčřřžýá",
  "email": 56,
  "age": "krásná"
}
```

Steps to reproduce:

1. V Postman odešlu pomocí metody POST data o studentovi (viz testovací data).
2. Ve Workbench zkontroluju pomocí SELECT * FROM student WHERE last_name = 'primeni'; zda v databázi existuje student se zadanými parametry.

Expected result:

Postman mi vrátí chybový stavový kód a nového studenta nevytvoří.

8. Při zadání číselných hodnot do položek vyžadující písemné vstupy (metoda POST) se objeví chybový stavový kód a student se nevytvoří

Testing data:

```
{
  "firstName":78,
  "lastName": 56,
  "email": 67,
  "age": 25
}
```

Steps to reproduce:

1. V Postman odešlu pomocí metody POST data o studentovi (viz testovací data).

2. Ve Workbench ověřím, že se student opravdu nevytvořil.

Expected result:

Postman mi vrátí chybový stavový kód a nového studenta nevytvoří.

9. Při ponechání prázdných polí na vstupu se vrátí chybný stavový kód a student se nevytvoří

Testing data:

```
{  
  "firstName": "",  
  "lastName": "primeni",  
  "email": "emailova_adresa",  
  "age": 25  
}
```

Steps to reproduce:

1. V Postman odešlu pomocí metody POST data o studentovi (viz testovací data).
2. V Postman ověřím, že odpověď obsahuje chybový kód. Ve Workbench ověřím, že se student opravdu nevytvořil.

Expected result:

Postman mi vrátí chybový stavový kód a nového studenta nevytvoří.

10. Při zadání špatného formátu e-mailu se vrátí chyba a student se nevytvoří

Testing data:

```
{  
  "firstName": "frantiska",  
  "lastName": "voprsalkova",  
  "email": "frantiska@voprsalkova",  
  "age": 25  
}
```

Steps to reproduce:

1. V Postman odešlu pomocí metody POST data o studentovi (viz testovací data).
2. Ve Workbench ověřím, že se student opravdu nevytvořil.

Expected result:

Postman mi vrátí chybový stavový kód a nového studenta nevytvoří.

11. Při zadání neplatného věku (záporná nebo příliš vysoká hodnota) se vrátí chyba a student se nevytvoří

Testing data:

"age": -5

"age": 250

Steps to reproduce:

1. V Postman odešlu pomocí metody POST data o studentovi (viz testovací data).
2. Ve Workbench ověřím, že se student opravdu nevytvořil.

Expected result:

Postman mi vrátí chybový stavový kód a nového studenta nevytvoří.

12. Při zadání duplicitního e-mailu se mi vrátí chyba a student se nevytvoří

Testing data:

frantiska@voprsalkova

Steps to reproduce:

1. V Postman odešlu pomocí metody POST data o studentovi (viz testovací data).
2. Ve Workbench ověřím, že se student opravdu nevytvořil.

Expected result:

Postman mi vrátí chybový stavový kód a nového studenta nevytvoří.

13. Při opakovaném zadání identických údajů se vrátí chyba a noví studenti se přestanou vytvářet.

Testing data:

```
{
  "firstName": "jmeno",
  "lastName": "primeni",
  "email": "frantiska@voprsalkova",
  "age": 250
}
```

Steps to reproduce:

1. V Postman odešlu opakovaně pomocí metody POST data o studentovi (viz testovací data).
2. Ve Workbench ověřím, že se student opravdu nevytvořil.

Expected result:

Postman mi podruhé vrátí chybový stavový kód a nového studenta nevytvoří.

14. Aplikace dovolí vytvoření studenta s duplicitním jménem, příjmením a věkem

Testing data:

```
{
  "firstName": "violet",
  "lastName": "sorrengail",
  "email": "vi@gmail.com",
  "age": 25
}

{
  "firstName": "violet",
  "lastName": "sorrengail",
  "email": "andarna@gmail.com",
  "age": 25
}

{
  "firstName": "violet",
  "lastName": "sorrengail",
  "email": "tairn@gmail.com",
  "age": 25
}
```

Steps to reproduce:

3. V Postman odešlu opakovaně pomocí metody POST data o studentech (viz testovací data).
4. Ve Workbench ověřím, že se studenti opravdu vytvořili.

Expected result:

Server vrátí 200 OK/201 Created a studenti se vytvoří.

15. Metoda DELETE vymaže studenta s daným ID z databáze

Testing data:

<http://108.143.193.45:8080/api/v1/students/2109>

<http://108.143.193.45:8080/api/v1/students/2113>

Steps to reproduce:

1. Ve Workbench SELECT * FROM student WHERE email="frantiska@voprsalkova" - zjistím si ID a
2. V Postman postupně odesílám požadavky na smazání.
3. Ověřím ve Workbench, že se student skutečně smazal.

Expected result:

Server vrátí 200 OK a data o studentovi se vymažou.

16. Při žádosti o vymazání neexistujícího studenta mi server vrátí 404

Testing data:

<http://108.143.193.45:8080/api/v1/students/5678>

<http://108.143.193.45:8080/api/v1/students/-456>

<http://108.143.193.45:8080/api/v1/students/10000>

Steps to reproduce:

1. V Postman postupně odesílám požadavky na smazání studentů s neexistujícím ID.

Expected result:

Server vrátí 404 Not Found.

17. Při žádosti o vymazání již jednou smazaného studenta mi server vrátí 404

Testing data:

<http://108.143.193.45:8080/api/v1/students/2109>

<http://108.143.193.45:8080/api/v1/students/2113>

Steps to reproduce:

1. V Postman postupně odesílám požadavky na smazání již smazaných studentů z předchozího testování.

Expected result:

Server vrátí 404 Not Found.

EXEKUCE TESTŮ

Testovací scénáře jsem provedla, přikládám výsledky testů.

1. Metoda GET zobrazí data o existujícím studentovi

Testing data:

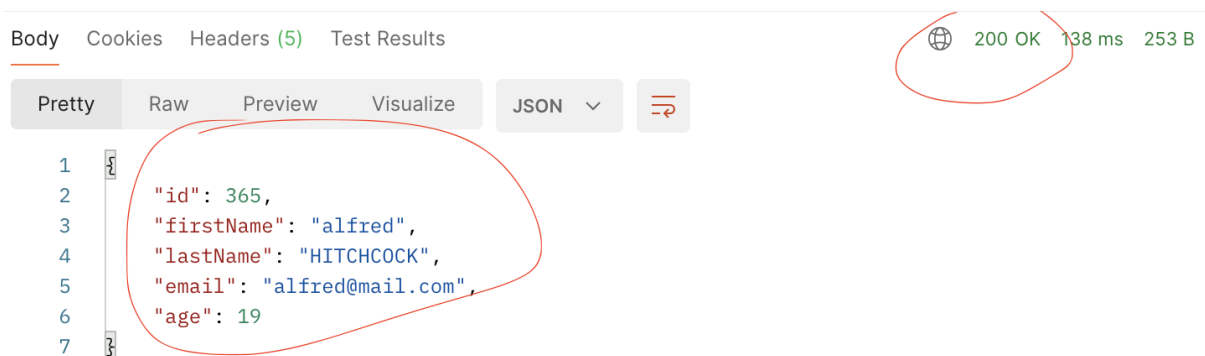
Steps to reproduce:

4. Ve Workbench si pomocí `SELECT * FROM student`; zobrazím všechna data z tabulky.
5. Vyberu si libovolného studenta, např. id = 365, age = 19, email = alfred@mail.com, first_name = alfred, last_name = HITCHCOCK
6. V aplikaci Postman zavolám metodou GET údaje o studentovi s id = 365

Expected result:

Aplikace Postman mi vrátí odpověď se stavovým kódem 200 OK a v těle odpovědi budou uvedena všechna pravdivá data, která jsou o studentovi uvedena v databázi.

Actual result:



Test status: **Passed**

2. Metoda GET zobrazí stavový kód 400 nebo 404 pro žádost o data o neexistujícím studentovi

Testing data: http://108.143.193.45:8080/api/v1/students/99999

http://108.143.193.45:8080/api/v1/students/0000000

http://108.143.193.45:8080/api/v1/students/14086579

http://108.143.193.45:8080/api/v1/students/2378545098765435678987654345678987654345678987654345678876545678987654

http://108.143.193.45:8080/api/v1/students/-2345678

http://108.143.193.45:8080/api/v1/students/1111

Steps to reproduce:

1. V Postman zadávám pomocí metody GET různé číselné požadavky pro získání dat o neexistujících studentech viz. Testing data.

Expected result:

Aplikace Postman mi pro každý číselný požadavek vrátí odpověď se stavovým kódem **404 Not Found** nebo **400 Bad Request** a v těle odpovědi budou prázdná data.

Actual result:

http://108.143.193.45:8080/api/v1/students/99999 500 Internal Server Error

http://108.143.193.45:8080/api/v1/students/0000000 500 Internal Server Error

http://108.143.193.45:8080/api/v1/students/14086579 500 Internal Server Error

http://108.143.193.45:8080/api/v1/students/2378545098765435678987654345678987654345678987654345678876545678987654 400 Bad Request

http://108.143.193.45:8080/api/v1/students/-2345678 500 Internal Server Error

http://108.143.193.45:8080/api/v1/students/1111 500 Internal Server Error

Test status: Failed

3. Metoda GET příkaz `http://108.143.193.45:8080/api/v1/students/` vrátí 200 OK a záznamy všech studentů v DTB

Testing data: `http://108.143.193.45:8080/api/v1/students/`

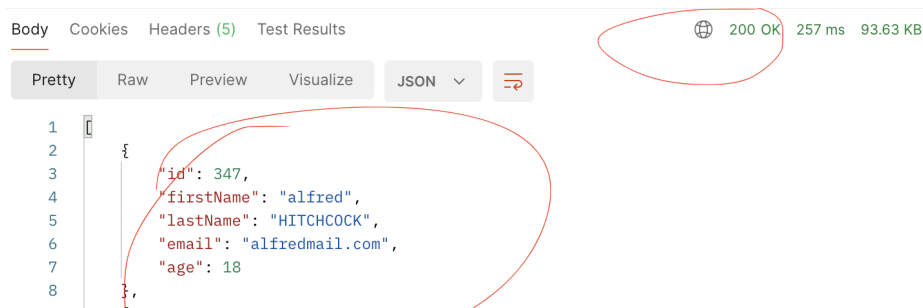
Steps to reproduce:

1. V Postman zadám pomocí metody GET požadavek pro získání dat o všech studentech uložených v DTB - `http://108.143.193.45:8080/api/v1/students/`

Expected result:

Aplikace Postman mi vrátí odpověď se stavovým kódem **200 OK** a v těle odpovědi budou data všech studentů z DTB.

Actual result:



Test status: Passed

4. Při zadání chybných vstupů vrátí GET chybovou hlášku 400 nebo 404 ve stavovém kódu

Testing data: `http://108.143.193.45:8080/api/v1/abc`

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

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

`http://108.143.193.45:8080/api/v1/students/fnhtukioýturztreawqasDCFVGBHJUKI
ÁÝTŽRTERESAFVGBHJZTRETFSDCDVBGHNMZKUÝŽŘRZES`

Steps to reproduce:

2. V Postman zadám pomocí metody GET požadavek pro získání dat o studentovi s id viz Testing data.

Expected result:

Aplikace Postman mi vrátí odpověď se stavovým kódem **400** nebo **404** nikoliv **500**.

Actual result:

http://108.143.193.45:8080/api/v1/abc 404 Not Found

http://108.143.193.45:8080/api/v1/students/petrzel 404 Not Found

http://108.143.193.45:8080/api/v1/students/fgrhtgefgehbtrwbfdvdf 400 Bad

Request

http://108.143.193.45:8080/api/v1/students/fnhtukioýturztreawqaSDCFVGBHJUKI
ÁÝTŽRTERESAFVGBHJZTRETFSDCDVBGHNMZKUÝŽŘRZES 400 Bad Request

Test status: Passed

5. Metoda GET vrátí vždy všechny položky z dat o studentovi (id, first name, last name, email, age)

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

http://108.143.193.45:8080/api/v1/students/354

http://108.143.193.45:8080/api/v1/students/368

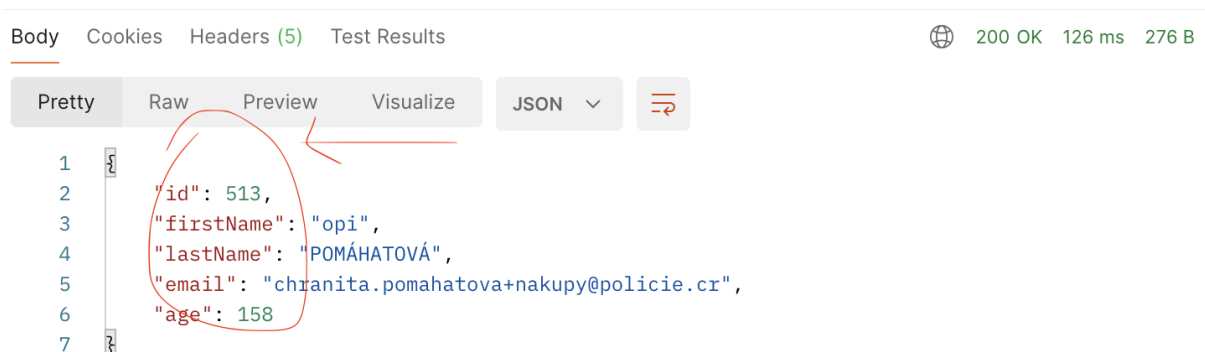
http://108.143.193.45:8080/api/v1/students/513

Steps to reproduce:

2. V Postman zadám pomocí metody GET požadavek pro získání dat o všech studentech a následně o konkrétních studentech s různými id viz Testing data,

Expected result:

V těle odpovědi budou data o studentech vždy obsahovat položky id, first name, last name, email a age.

Actual result:

Test status: Passed

6. Metoda POST vytvoří v databázi studenta se zadanými parametry

Testing data: {

```
"firstName": "jmeno",
"lastName": "primeni",
"email": "emailova_adresa",
"age": 25
}
{
"firstName": "lumira",
"lastName": "svetlova",
"email": "lumira@svetlova.cz",
"age": 15
}
{
"firstName": "LUMIRA",
"lastName": "SVETLOVA",
"email": "LUMIRA@SVETLOVA.CZ",
"age": 15
}
```

Steps to reproduce:

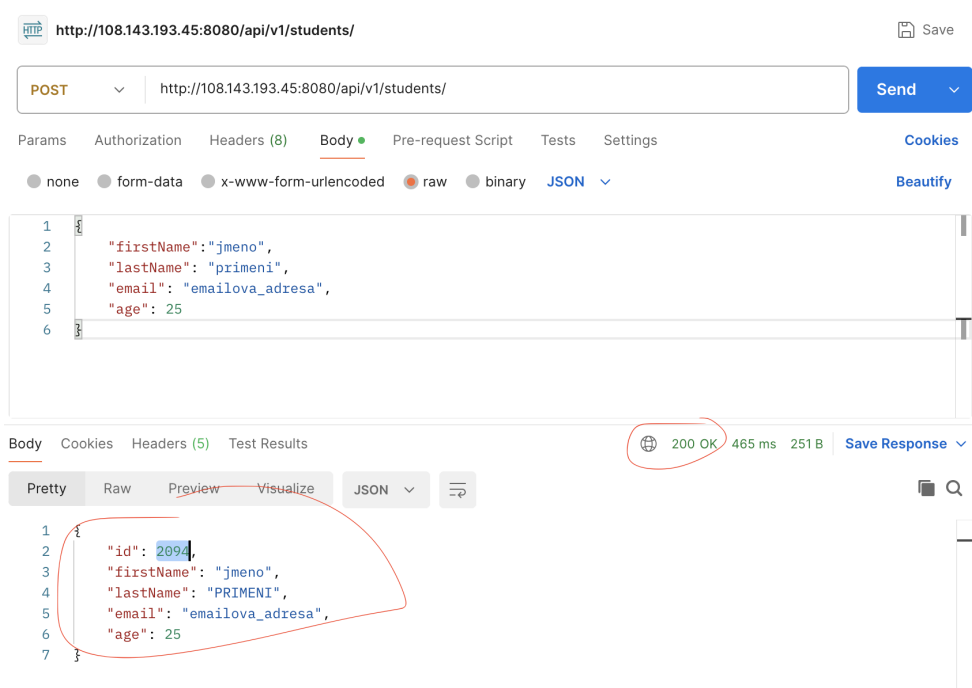
1. V Postman odešlu pomocí metody POST data o studentovi (viz testovací data).
2. Ve Workbench zkontroluju pomocí `SELECT * FROM student WHERE last_name = 'primeni'`; zda v databázi existuje student se zadanými parametry.
3. Postman mi vrátil 200 OK a ID nového studenta. Ve Workbench zkontroluji data o novém studentovi podle nového ID. Použiji příkaz `SELECT * FROM student WHERE id = 2094`;

Expected result:

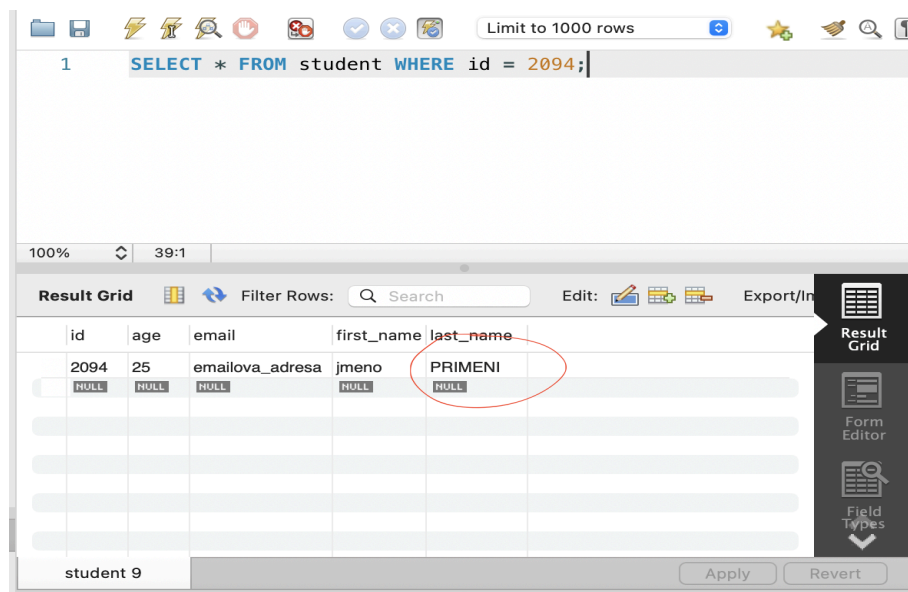
Postman mi vrátí 201 Created a v těle odpovědi budou zadané parametry nově vytvořeného studenta. Při kontrole ve Workbench najdu vytvořeného studenta se zadanými parametry.

Actual result:

Aplikace Postman vrátí 200 OK a data o nově vytvořeném studentovi:

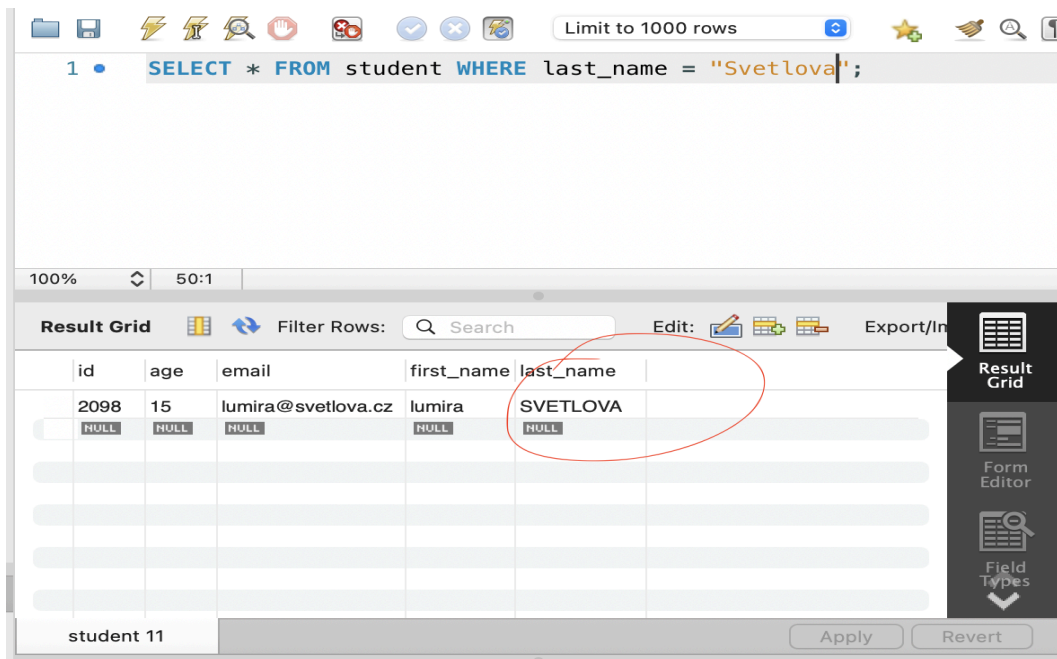


Ve Workbench se vytvoří nový student se zadanými parametry:

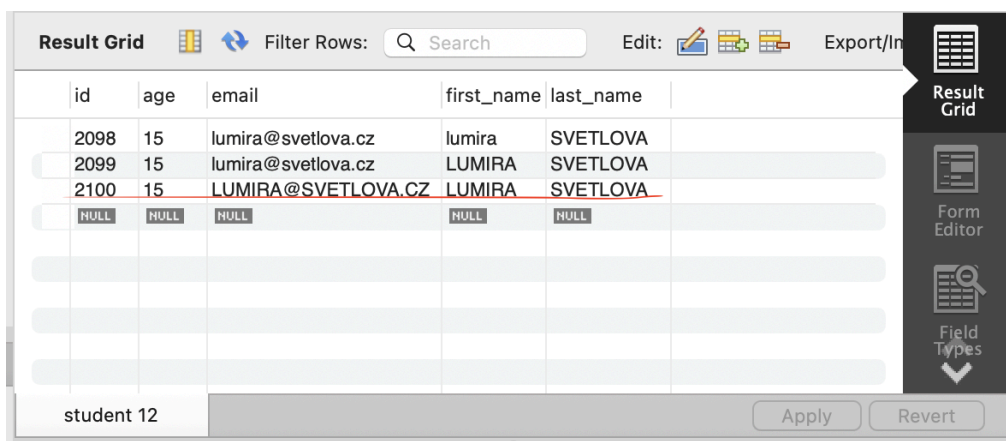


V zadání je položka last_name = "primeni" definována s malými písmeny, nicméně po odeslání metody POST se položka last_name vytvoří v kapitálkách - PRIMENI.

Při zadání nových náhodných dat (viz Testing data) získám stejný (pozitivní) výsledek se stejnou formátovou chybou:



Pokud zadám data na vstupu v kapitálkách, na výstupu se uloží také v kapitálkách.



7. Metoda POST nevytvoří studenta se špatně zadanými parametry a vrátí chybu ve stavovém kódu

Testing data:

```
{
  "firstName": 345369,
  "lastName": "ýřčřřžžýá",
  "email": 56,
  "age": "krásná"
}
```

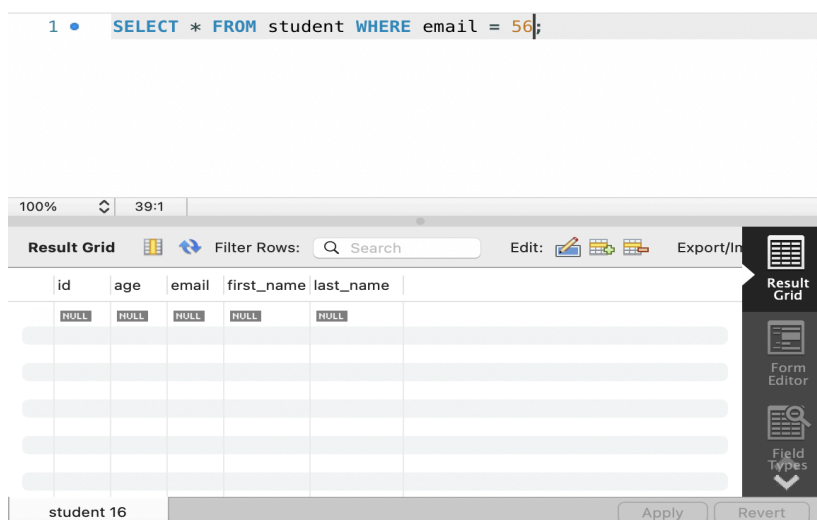
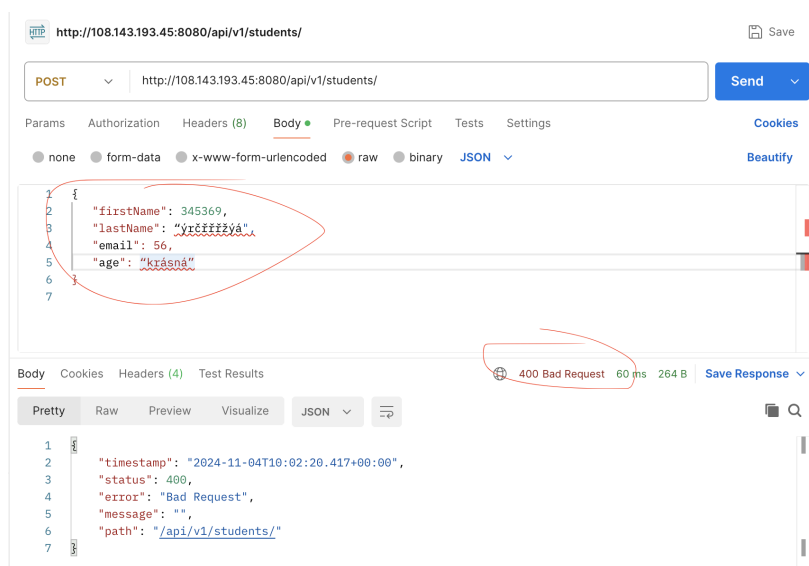
Steps to reproduce:

3. V Postman odešlu pomocí metody POST data o studentovi (viz testovací data).
4. Ve Workbench zkontroluju pomocí SELECT * FROM student WHERE last_name = 'primeni'; zda v databázi existuje student se zadanými parametry.

Expected result:

Postman mi vrátí chybový stavový kód a nového studenta nevytvoří.

Actual result: 400 Bad Request. Ověřeno ve Workbench - student se skutečně nevytvoří.



Test status: Passed

8. Při zadání číselných hodnot do položek vyžadující písemné vstupy (metoda POST) se objeví chybový stavový kód a student se nevytvoří

Testing data:

```
{
  "firstName":78,
  "lastName": 56,
  "email": 67,
  "age": 25
}
```

Steps to reproduce:

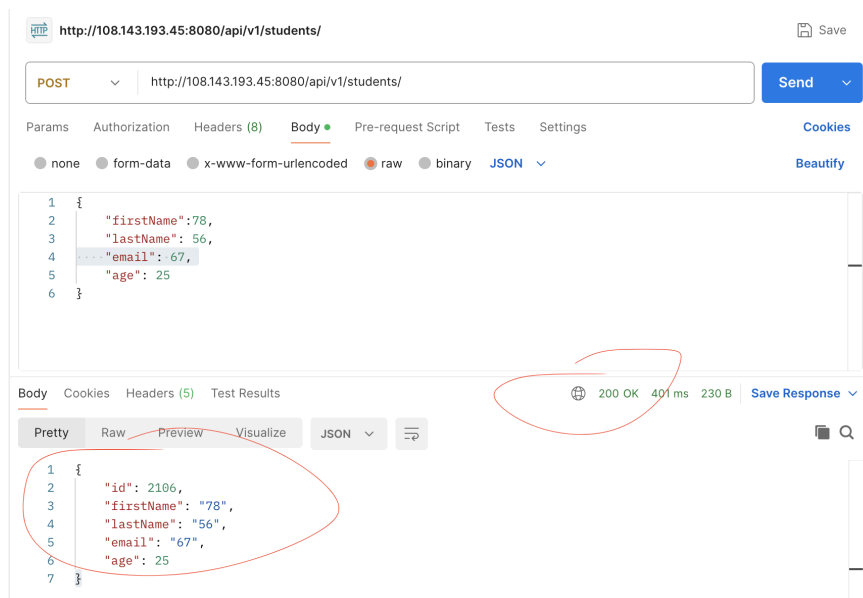
1. V Postman odešlu pomocí metody POST data o studentovi (viz testovací data).
2. Ve Workbench ověřím, že se student opravdu nevytvořil.

Expected result:

Postman mi vrátí chybový stavový kód a nového studenta nevytvoří.

Actual result:

Postman vrátí 200 OK a student se vytvoří.



Result Grid

Filter Rows:

Edit:

Export/Import

id	age	email	first_name	last_name
2106	25	67	78	56
NULL	NULL	NULL	NULL	NULL

student 15

Apply

Revert

Result Grid

Form Editor

Field Types

Test status: Failed

9. Při ponechání prázdných polí na vstupu se vrátí chybný stavový kód a student se nevytvoří

Testing data:

```
{
  "firstName": "",
  "lastName": "primeni",
  "email": "emailova_adresa",
  "age": 25
}
```

Steps to reproduce:

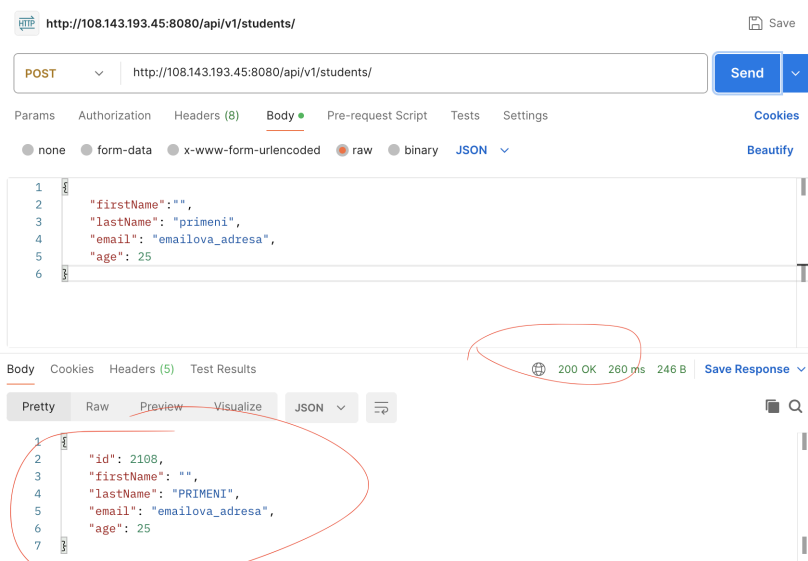
1. V Postman odešlu pomocí metody POST data o studentovi (viz testovací data).
2. V Postman ověřím, že odpověď obsahuje chybový kód. Ve Workbench ověřím, že se student opravdu nevytvořil.

Expected result:

Postman mi vrátí chybový stavový kód a nového studenta nevytvoří.

Actual result:

Postman mi 200 OK a nového studenta vytvoří, viz dále:



id	age	email	first_name	last_name
2108	25	emailova_adresa	PRIMENI	
NULL	NULL	NULL	NULL	NULL

Test status: **Failed**

10. Při zadání špatného formátu e-mailu se vrátí chyba a student se nevytvoří

Testing data:

```
{  "firstName": "frantiska",  "lastName": "voprsalkova",  "email": "frantiska@voprsalkova",  "age": 25}
```

Steps to reproduce:

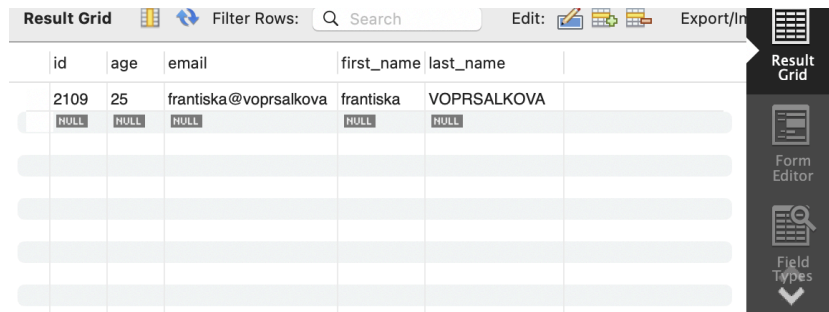
3. V Postman odešlu pomocí metody POST data o studentovi (viz testovací data).
4. Ve Workbench ověřím, že se student opravdu nevytvořil.

Expected result:

Postman mi vrátí chybový stavový kód a nového studenta nevytvoří.

Actual result:

Student se vytvoří i se špatným formátem e-mailu.



id	age	email	first_name	last_name
2109	25	frantiska@voprsalkova	frantiska	VOPRSALKOVA
NULL	NULL	NULL	NULL	NULL

Test status: Failed

11. zadání neplatného věku (záporná nebo příliš vysoká hodnota) se vrátí chyba a student se nevytvoří

Testing data:

"age": -5

"age": 250

Steps to reproduce:

1. V Postman odešlu pomocí metody POST data o studentovi (viz testovací data).
2. Ve Workbench ověřím, že se student opravdu nevytvořil.

Expected result:

Postman mi vrátí chybový stavový kód a nového studenta nevytvoří.

Expected result:

V obou testovaných případech se vrátilo 200 OK a student vytvořil.

Test status: Failed

12. Při zadání duplicitního e-mailu se mi vrátí chyba a student se nevytvoří

Testing data:

frantiska@voprsalkova

Steps to reproduce:

1. V Postman odešlu pomocí metody POST data o studentovi (viz testovací data).
2. Ve Workbench ověřím, že se student opravdu nevytvořil.

Expected result:

Postman mi vrátí chybový stavový kód a nového studenta nevytvoří.

Actual result:

Vrátí se 200 OK a student se vytvoří i s duplicitním e-mailem.

Test status: **Failed**

13. Při opakovaném zadání identických údajů se vrátí chyba a noví studenti se přestanou vytvářet.

Testing data:

```
{  
  "firstName": "jmeno",  
  "lastName": "primeni",  
  "email": "frantiska@voprsalkova",  
  "age": 250  
}
```

Steps to reproduce:

5. V Postman odešlu opakovaně pomocí metody POST data o studentovi (viz testovací data).
6. Ve Workbench ověřím, že se student opravdu nevytvořil.

Expected result:

Postman mi podruhé vrátí chybový stavový kód a nového studenta nevytvoří.

Actual result:

Aplikace opakovaně vytváří nové a nové duplicitní studenty s novým ID.

Test status: **Failed**

14. Aplikace dovolí vytvoření studenta s duplicitním jménem, příjmením a věkem

Testing data:

```
{
  "firstName": "violet",
  "lastName": "sorrengail",
  "email": "vi@gmail.com",
  "age": 25
}

{
  "firstName": "violet",
  "lastName": "sorrengail",
  "email": "andarna@gmail.com",
  "age": 25
}

{
  "firstName": "violet",
  "lastName": "sorrengail",
  "email": "tairn@gmail.com",
  "age": 25
}
```

Steps to reproduce:

1. V Postman odešlu opakovaně pomocí metody POST data o studentech (viz testovací data).
2. Ve Workbench ověřím, že se studenti opravdu vytvořili.

Expected result:

Server vrátí 200 OK/201 Created a studenti se vytvoří.

Actual result:

Server vrátí 200 OK a studenti se vytvoří.

Result Grid						
		Filter Rows:		Search		
		Edit:		Export/In		
	id	age	email	first_name	last_name	
	2120	25	vi@gmail.com	violet	SORRENGAIL	
	2121	25	andarna@gmail.com	violet	SORRENGAIL	
	2122	25	tairn@gmail.com	violet	SORRENGAIL	
	NULL	NULL	NULL	NULL	NULL	

Test status: **Passed**

15. Metoda DELETE vymaže studenta s daným ID z databáze

Testing data:

http://108.143.193.45:8080/api/v1/students/2109

http://108.143.193.45:8080/api/v1/students/2113

Steps to reproduce:

4. Ve Workbench SELECT * FROM student WHERE email="frantiska@voprsalkova" - zjistím si ID a
5. V Postman postupně odesílám požadavky na smazání.
6. Ověřím ve Workbench, že se student skutečně smazal.

Expected result:

Server vrátí 200 OK a data o studentovi se vymažou.

Actual result:

Server vrátí 200 OK a data o studentovi se skutečně vymažou.

Test status: **Passed**

16. Při žádosti o vymazání neexistujícího studenta mi server vrátí 404

Testing data:

http://108.143.193.45:8080/api/v1/students/5678

http://108.143.193.45:8080/api/v1/students/-456

http://108.143.193.45:8080/api/v1/students/10000

Steps to reproduce:

2. V Postman postupně odesílám požadavky na smazání studentů s neexistujícím ID.

Expected result:

Server vrátí 404 Not Found.

Actual result:

Server vrátí 500 Internal Server Error

Test status: Failed

17. Při žádosti o vymazání již jednou smazaného studenta mi server vrátí 404

Testing data:

http://108.143.193.45:8080/api/v1/students/2109

http://108.143.193.45:8080/api/v1/students/2113

Steps to reproduce:

1. V Postman postupně odesílám požadavky na smazání již smazaných studentů z předchozího testování.

Expected result:

Server vrátí 404 Not Found.

Actual result:

Server vrátí 500 Internal Server Error.

Test status: Failed

BUG REPORT

Na základě provedených scénářů jsem objevila uvedené chyby aplikace.

bug	probability	severity	test case number	mitigace
Při žádosti o data o neexistujícím studentovi (zadání náhodných čísel) vrací GET stavový kód 500 Internal Server Error	High	High	2	Server by měl vrátit 404 Not Found.
Při vkládání dat metodou POST se změni formát dat vložených do položky "last_name" z malých písmen na velká.	High	Medium	6	Při zadání malých písmen do položky "last_name" se příjmení uloží do databáze v malých písmenách a naopak.
(Metoda POST) Při vložení číselných vstupů do položek jméno, příjmení a email se vrátí 200 OK a student se vytvoří.	High	High	8	Aplikace nepovolí zadávání číselných údajů do položek vyžadujících písmenný vstup.
(Metoda POST) Při odeslání dat a vytvoření nového studenta se mi v Postmanovi vrátí kód 200 OK namísto 201 Created.	High	Low	6	Při úspěšném vytvoření studenta vrátí server odpověď 201 Created.
(Metoda POST) Při zadání prázdných polí se nenahlásí chyba a student se vytvoří i s neúplnými údaji.	High	High	9	Aplikace nepovolí zadání ponechání prázdných polí a vrátí chybu.

Aplikace vytvoří studenta i tehdy, pokud zadám špatný formát e-mailu.	High	High	10	Aplikace nepovolí zadání neplatného formátu emailu a upozorní na něj chybovou hláškou.
Pokud zadám neplatná věk (záporný a příliš vysoký), tak se vrátí 200 OK a student se vytvoří.	High	High	11	Aplikace nepovolí zadání neplatného věku a upozorní na něj chybovou hláškou.
Při zadání duplicitního e-mailu se vrátí 200 OK a student se vytvoří.	High	High	12	Aplikace nepovolí vytvoření nového záznamu s již existujícím emailem.
Aplikace dovoluje opakované zadávání identických údajů a ty se propisují a ukládají do databáze.	High	High	13	Při opakovaném zadání údajů, které již v dtb existují vrátí aplikace chybu a studenta nevytvoří.
Při žádosti o vymazání studenta s neexistujícím ID vrátí server stavový kód 500.	High	High	16	Při žádosti o vymazání studenta s neexistujícím ID server vrátí 404 Not Found.
Při žádosti o vymazání již jednou smazaného studenta mi server vrátí 500 Internal Server Error.	High	High	17	Při žádosti o vymazání již jednou smazaného studenta mi server vrátí 404 Not Found.