

FINÁLNÍ PROJEKT

č.1



Autor: Ing. Monika Cochová
Datum: 20.11.2024

Obsah

ZADÁNÍ.....	3
Přístupové údaje:	3
Poznámky:	3
TESTOVACÍ SCÉNÁŘE A EXEKUCE TESTŮ.....	4
BUG REPORT	14

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 A EXEKUCE TESTŮ

Na základě uvedených testovacích scénářů jsem ověřila funkčnost aplikace.

ID	Název	Kroky	Očekávaný výsledek	Skutečný výsledek	Stav
1	Testování metody GET s validními údaji	<p>1. Pro zjištění existujících id studentů otevřu MySQL Workbench a nechám vypsat všechny studenty z databáze pomocí jejich "id" pomocí příkazu SELECT id FROM student;</p> <p>2. Ze seznamu vyberu například id=352</p> <p>3. Otevřu program Postman, kde pomocí metody GET ověřím, že student s id=352 existuje pomocí příkazu GET http://108.143.193.45:8080/api/v1/students/352</p> <p>4. Potvrdím tlačítkem Send</p> <p>5. Zkontroluji, že získané údaje souhlasí s údaji v databázi. Vrátím se do MySQL Workbench a zadám příkaz SELECT * FROM student WHERE id=352;</p>	Status kód: 200 (OK), body = výpis s odpovídajícími údaji	Status kód: 200 (OK), body = { "id": 352, "firstName": "Jana", "lastName": "NOVAKOVA", "email": "janovak@gmail.com", "age": 19 }	passed
2	Testování metody GET s nevalidními údaji	<p>1. Pro zjištění existujících id studentů otevřu MySQL Workbench a nechám vypsat všechny studenty z databáze pomocí jejich "id" pomocí příkazu SELECT id FROM student;</p> <p>2. Vymyslím číslo id, které není v seznamu, např. 300 a znovu ověřím, že opravdu neexistuje příkazem SELECT * FROM student WHERE id=300;</p> <p>3. Otevřu program Postman, kde pomocí metody GET ověřím, že student s id=300 neexistuje pomocí příkazu GET http://108.143.193.45:8080/api/v1/students/300</p> <p>3. Potvrdím tlačítkem Send</p>	Status kód: 404 (Not Found), body: výpis s odpovídající chybovou hláškou	Status kód: 500 (Internal Server Error), body: { "timestamp": "2024-11-19T20:58:13.606+00:00", "status": 500, "error": "Internal Server Error", "message": "", "path": "/api/v1/students/300" }	failed

3	Testování metody GET s chybným parametrem	1. V programu Postman zadám příkaz GET http://108.143.193.45:8080/api/v1/students/abc	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou	Status kód: 400 (Bad Request), body: { "timestamp": "2024-11-19T20:56:34.439+00:00", "status": 400, "error": "Bad Request", "message": "", "path": "/api/v1/students/abc" }	passed
		2. Potvrdím tlačítkem Send			
4	Testování metody POST s validními údaji	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Monika", "lastName": "Cochova", "email": "mc@gmail.com", "age": 30 }	Status kód: 201 (Created), body = { "id": ..., "firstName": "Monika", "lastName": "COCHOVA", "email": "mc@gmail.com", "age": 30 }. Do databáze byl student přidán.	status kód: 200 (OK), body = { "id": 2342, "firstName": "Monika", "lastName": "COCHOVA", "email": "mc@gmail.com", "age": 30 }. Do databáze byl student přidán.	failed
		2. Potvrdím tlačítkem Send			
		3. Zkontroluji, zda výstup v záložce Body odpovídá zadaným parametrům a zda je správně status kód.			
		4. Zjistím přidělené id			
		5. Zkontroluji ve Workbench, zda byl do databáze skutečně přidán student odpovídající přidělenému id pomocí příkazu SELECT * FROM student WHERE id=2342;			

5	Testování metody POST s chybějícími parametry	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Marek", "lastName": "Kohout", "age": 15 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou	Status kód: 500 (Internal Server Error), body: { "timestamp": "2024-11-19T20:55:46.946+00:00", "status": 500, "error": "Internal Server Error", "message": "", "path": "/api/v1/students/" }	failed
		2. Potvrdím tlačítkem Send			
6	Testování metody POST se špatným formátem dat - first name - čísla ve stringu	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "25964", "lastName": "Chybova", "email": "chkdil@gmail.com", "age": 30 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), body: { "id": 2355, "firstName": "25964", "lastName": "CHYBOVA", "email": "chkdil@gmail.com", "age": 30 }. Student byl přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			
7	Testování metody POST se špatným formátem dat - first name - čísla - type int	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": 25964, "lastName": "Rovná", "email": "2R@gmail.com", "age": 30 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), body: { "id": 2357, "firstName": "25964", "lastName": "ROVNÁ", "email": "2R@gmail.com", "age": 30 }. Student byl přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			

8	Testování metody POST se špatným formátem dat - first name - speciální znaky	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "#?!", "lastName": "Ostrý", "email": "ostry@gmail.com", "age": 30 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), body: { "id": 2358, "firstName": "#?!", "lastName": "OSTRÝ", "email": "ostry@gmail.com", "age": 30 }. Student byl přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			
9	Testování metody POST se špatným formátem dat - first name - string s jinak udělanými velkými a malými písmeny	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "rOmAnA", "lastName": "Přecechtělová", "email": "rp@gmail.com", "age": 14 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), body: { "id": 2359, "firstName": "rOmAnA", "lastName": "PŘECECHTĚLOVÁ", "email": "rp@gmail.com", "age": 14 }. Student byl přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			
10	Testování metody POST se špatným formátem dat - first name - příliš dlouhé neexistující jméno	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Lorem ipsum dolor sit amet, consectetur adipiscing elit.", "lastName": "Sametová", "email": "LS@gmail.com", "age": 19 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), body: { "id": 2360, "firstName": "Lorem ipsum dolor sit amet, consectetur adipiscing elit.", "lastName": "SAMETOVÁ", "email": "LS@gmail.com", "age": 19 }. Student byl přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			

11	Testování metody POST se špatným formátem dat - last name - číslo ve stringu	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Barbora", "lastName": "123456789", "email": "chkdil@gmail.com", "age": 30 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), body: { "id": 2354, "firstName": "Barbora", "lastName": "123456789", "email": "chkdil@gmail.com", "age": 30 }. Student byl přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			
12	Testování metody POST se špatným formátem dat - last name - čísla - type int	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Barbora", "lastName": 123, "email": "b1@gmail.com", "age": 35 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), body: { "id": 2361, "firstName": "Barbora", "lastName": "123", "email": "b1@gmail.com", "age": 35 }. Student byl přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			
13	Testování metody POST se špatným formátem dat - first name - speciální znaky	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Karel", "lastName": "#?!/", "email": "k1@gmail.com", "age": 35 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), body: { "id": 2362, "firstName": "Karel", "lastName": "#?!/", "email": "k1@gmail.com", "age": 35 }. Student byl přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			

14	Testování metody POST se špatným formátem dat - first name - string s jinak udělanými velkými a malými písmeny	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Karel", "lastName": "sTuDnlčKa", "email": "ks@gmail.com", "age": 55 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), body: { "firstName": "Karel", "lastName": "sTuDnlčKa", "email": "ks@gmail.com", "age": 55 }. Student byl přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			
15	Testování metody POST se špatným formátem dat - first name - příliš dlouhé neexistující jméno	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Lolek", "lastName": "Lorem ipsum dolor sit amet, consectetur adipiscing elit.", "email": "LL@gmail.com", "age": 19 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), body: { "id": 2364, "firstName": "Lolek", "lastName": "LOREM IPSUM DOLOR SIT AMET, CONSECTETUR ADIPISCING ELIT.", "email": "LL@gmail.com", "age": 19 }. Student byl přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			
16	Testování metody POST se špatným formátem dat - email - čísla	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Barbora", "lastName": "Okurková", "email": "123456789", "age": 30 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), body: { "id": 2353, "firstName": "Barbora", "lastName": "OKURKOVÁ", "email": "123456789", "age": 30 }. Student byl přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			

17	Testování metody POST se špatným formátem dat - email - speciální znaky	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Eleonora", "lastName": "Drahounová", "email": "#@%^%#\$@#\$@#.com", "age": 30 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), body: { "id": 2365, "firstName": "Eleonora", "lastName": "DRAHOUNOVÁ", "email": "#@%^%#\$@#\$@#.com", "age": 30 }. Student byl přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			
18	Testování metody POST se špatným formátem dat - email - dvojitý @	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Jan", "lastName": "Vaše", "email": "email@example@example.com", "age": 35 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), body: { "id": 2366, "firstName": "Jan", "lastName": "VAŠE", "email": "email@example@example.com", "age": 35 }. Student byl přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			
19	Testování metody POST se špatným formátem dat - email - neexistující přípona	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Patrik", "lastName": "Hrom", "email": "email@example.web", "age": 35 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), body: { "id": 2367, "firstName": "Patrik", "lastName": "HROM", "email": "email@example.web", "age": 35 }. Student byl přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			

20	Testování metody POST se špatným formátem dat - age - string	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Alex", "lastName": "Guláš", "email": "ab@seznam.cz", "age": "abcdef" }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 400 (Bad Request), body: { "timestamp": "2024-11-19T20:48:51.081+00:00", "status": 400, "error": "Bad Request", "message": "", "path": "/api/v1/students/" }. Student není přidán do databáze.	passed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			
21	Testování metody POST s nevalidním vstupem - příliš velký int	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Barbora", "lastName": "Okurková", "email": "kdifj@gmail.com", "age": 123456 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), { "id": 2347, "firstName": "Barbora", "lastName": "OKURKOVÁ", "email": "kdifj@gmail.com", "age": 123456 }. Student je přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			
22	Testování metody POST s nevalidním vstupem - záporný int	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Božidara", "lastName": "Okurková", "email": "kdifj@gmail.com", "age": -125 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), { "id": 2349, "firstName": "Božidara", "lastName": "OKURKOVÁ", "email": "kdifj@gmail.com", "age": -125 }. Student je přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			

23	Testování metody POST s nevalidním vstupem - typ float	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Matěj", "lastName": "Rum", "email": "mr@gmail.com", "age": 55.5 }	Status kód: 400 (Bad Request), body: výpis s odpovídající chybovou hláškou. Student není přidán do databáze.	Status kód: 200 (OK), { "id": 2368, "firstName": "Matěj", "lastName": "RUM", "email": "mr@gmail.com", "age": 55 }. Student je přidán do databáze.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			
24	Testování metody POST - DUPLICITA	1. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON následující údaje: { "firstName": "Olga", "lastName": "Hrabošová", "email": "ohoho@gmail.com", "age": 40 }	Status kód: 409 (Conflict), body: výpis s odpovídající chybovou hláškou. Duplicitní student není přidán.	Status kód: 200 (OK), Došlo k vytvoření obou studentů { "id": 2351, "firstName": "Olga", "lastName": "HRABOŠOVÁ", "email": "ohoho@gmail.com", "age": 40 } { "id": 2352, "firstName": "Olga", "lastName": "HRABOŠOVÁ", "email": "ohoho@gmail.com", "age": 40 }. Duplicitní student je přidán.	failed
		2. Potvrdím tlačítkem Send			
		3. Ověřím ve Workbench, zda byl student přidán do databáze.			
		4. V programu Postman vytvořím nového studenta pomocí metody POST. Do pole "Body" vložím v "raw" ve formátu JSON totožné údaje jako v předchozím případě: { "firstName": "Olga", "lastName": "Hrabošová", "email": "ohoho@gmail.com", "age": 40 }			
		5. Ověřím ve Workbench, zda byl student přidán do databáze.			

25	Testování metody DELETE - existující záznam	1. V programu Postman vymažu studenta, kterého jsem si vytvořila s id=2352 pomocí příkazu DELETE http://108.143.193.45:8080/api/v1/students/2352	Status kód: 200 (OK) s odpovídající odpovědí v body nebo 204 (No Content) bez odpovědi v sekci body. Došlo ke smazání studenta z databáze.	Status kód: 200, body: prázdné. Student byl smazán.	failed
		2. Potvrdím tlačítkem Send			
		3. Zkontroluji, že ke smazání došlo skutečně i v databázi. Vrátím se do MySQL Workbench a zadám příkaz SELECT * FROM student WHERE id=2352;			
26	Testování metody DELETE - neexistující záznam	1. V programu Postman vymažu studenta, kterého jsem už jednou vymazala s id=2352 pomocí příkazu DELETE http://108.143.193.45:8080/api/v1/students/2352	Status kód: 404 (Not Found), body: výpis s odpovídající chybovou hláškou	Status kód: 500 (Internal Server Error), Body: { "timestamp": "2024-11-19T20:46:30.171+00:00", "status": 500, "error": "Internal Server Error", "message": "", "path": "/api/v1/students/2352" }	failed
		2. Potvrdím tlačítkem Send			

BUG REPORT

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

Bug	Pravděpodobnost (probability)	Závažnost (severity)	Test case number	Mitigace
Po pokusu o získání údajů neexistujícího studenta metodou GET vrátí server chybu s kódem 500 (Internal Server Error).	vysoká	vysoká	2	V tomto případě vracet kód 404 (Not Found)
Po vytvoření nového studenta s validními údaji pomocí metody POST vrací server kód 200 (OK).	vysoká	nízká	4	V tomto případě vracet kód 201 (Created)
Při pokusu o vložení nového studenta s chybějícím parametrem vrací server kód 500 (Internal Server Error).	vysoká	vysoká	5	V tomto případě vrátit kód 400 (Bad Request)
Server umožní vložit studenta s nevalidním vstupem u parametru first name - číslo ve stringu. Vrací kód 200 (OK).	vysoká	střední	6	V tomto případě vrátit kód 400 (Bad Request)
Server umožní vložit studenta s nevalidním vstupem u parametru first name - integer. Vrací kód 200 (OK).	vysoká	střední	7	V tomto případě vrátit kód 400 (Bad Request)
Server umožní vložit studenta s nevalidním vstupem u parametru first name - speciální znaky. Vrací kód 200 (OK).	vysoká	střední	8	V tomto případě neukládat studenta do databáze a vrátit kód 400 (Bad Request)
Server umožní vložit studenta s nevalidním vstupem u parametru first name - string písmen s jinak udělanými velkými a malými písmeny. Předpoklad je vložení textového řetězce začínající na velké písmeno na začátku jména a ostatní písmena mají být malá. Vrací kód 200 (OK).	vysoká	střední	9	V tomto případě vrátit kód 400 (Bad Request)

Server umožní vložit studenta s nevalidním vstupem u parametru first name - příliš dlouhé neexistující jméno. Vrací kód 200 (OK).	vysoká	střední	10	V tomto případě vrátit kód 400 (Bad Request)
Server umožní vložit studenta s nevalidním vstupem u parametru last name - číslo ve stringu. Vrací kód 200 (OK).	vysoká	střední	11	V tomto případě vrátit kód 400 (Bad Request)
Server umožní vložit studenta s nevalidním vstupem u parametru last name - integer. Vrací kód 200 (OK).	vysoká	střední	12	V tomto případě vrátit kód 400 (Bad Request)
Server umožní vložit studenta s nevalidním vstupem u parametru last name - speciální znaky. Vrací kód 200 (OK).	vysoká	střední	13	V tomto případě vrátit kód 400 (Bad Request)
Server umožní vložit studenta s nevalidním vstupem u parametru last name - string písmen s jinak udělanými velkými a malými písmeny. Předpoklad je vložení textového řetězce začínající na velké písmeno na začátku jména a ostatní písmena mají být malá. Vrací kód 200 (OK).	vysoká	střední	14	V tomto případě vrátit kód 400 (Bad Request)
Server umožní vložit studenta s nevalidním vstupem u parametru last name - příliš dlouhé neexistující jméno. Vrací kód 200 (OK).	vysoká	střední	15	V tomto případě vrátit kód 400 (Bad Request)
Server umožní vložit studenta s nevalidním vstupem u parametru email - integer. Vrací kód 200 (OK).	vysoká	střední	16	V tomto případě vrátit kód 400 (Bad Request)
Server umožní vložit studenta s nevalidním vstupem u parametru email - speciální znaky. Vrací kód 200 (OK).	vysoká	střední	17	V tomto případě vrátit kód 400 (Bad Request)
Server umožní vložit studenta s nevalidním vstupem u parametru email - dvojité @. Vrací kód 200 (OK).	vysoká	střední	18	V tomto případě vrátit kód 400 (Bad Request)

Server umožní vložit studenta s nevalidním vstupem u parametru email - nevalidní přípona. Vrací kód 200 (OK).	vysoká	střední	19	V tomto případě vrátit kód 400 (Bad Request)
Server umožní vložit studenta s nevalidním vstupem u parametru age - příliš velký int. Vrací kód 200 (OK).	vysoká	střední	21	V tomto případě vrátit kód 400 (Bad Request)
Server umožní vložit studenta s nevalidním vstupem u parametru age - záporný int. Vrací kód 200 (OK).	vysoká	střední	22	V tomto případě vrátit kód 400 (Bad Request)
Server umožní vložit studenta s nevalidním vstupem u parametru age - float. Upraví na int a vrací kód 200 (OK).	vysoká	střední	23	V tomto případě vrátit kód 400 (Bad Request)
Server umožní vložit duplicitní záznam studenta. Vrací kód 200 (OK).	vysoká	střední	24	V tomto případě vrátit kód 409 (Conflict)
Server po vymazání existujícího studenta z databáze metodou DELETE vrací kód 200 (OK) bez odpovídající odpovědi v Body.	vysoká	nízká	25	V tomto případě vrátit kód 200 (OK) s odpovídající odpovědí v "Body", nebo kód 204 (No Content)
Server po příkazu k vymazání neexistujícího studenta z databáze metodou DELETE vrací kód 500 (Internal Server Error).	vysoká	vysoká	26	V tomto případě vrátit kód 404 (Not Found)