FINÁLNÍ PROJEKT   
 č.1



Autor: Radim Černil

Discord name: nabis

Datum: 16.8.2024

OBSAH

[**ZADÁNÍ 3**](#_30j0zll)

[**TESTOVACÍ SCÉNÁŘE 4**](#_2et92p0)

[**EXEKUCE TESTŮ 5**](#_tyjcwt)

[**BUG REPORT 5**](#_3dy6vkm)

# 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 | database: qa\_demo Host: aws.connect.psdb.cloud Username: *\*\*\*\*\*\**  Password: \*\*\*\*\*\*\* |
| --- | --- |
| 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.*

### **1. Testování metody GET**

#### **Testovací scénář 1: Získání dat o existujícím studentovi**

* **Cíl:** Ověřit, zda metoda GET vrací správná data pro existujícího studenta.
* **URL:** http://108.143.193.45:8080/api/v1/students/319
* **Metoda:** GET
* **Vstupní data:**
  + id existujícího studenta (319)
* **Očekávaný výstup:**
  + Status kód: 200 OK
  + Tělo odpovědi: JSON objekt obsahující data studenta  
    {  
     "id": 319,  
     "firstName": "Test",  
     ”lastName” : “TESTSDF”,  
     "email": "[jjohn34@yourmail.com](mailto:jjohn34@yourmail.com)",  
     "age": 34  
     }
* **Popis:** Ověřte, že vrácená data odpovídají záznamu ve vaší databázi. Zkontrolujte, že všechny pole jsou přítomné a odpovídají očekávaným hodnotám.

#### **Testovací scénář 2: Získání dat o neexistujícím studentovi**

* **Cíl:** Ověřit, jak API reaguje na požadavek o neexistujícím studentovi.
* **URL:** http://108.143.193.45:8080/api/v1/students/199
* **Metoda:** GET
* **Vstupní data:**
  + id neexistujícího studenta (199)
* **Očekávaný výstup:**
  + Status kód: 404 Not Found
  + Tělo odpovědi: JSON objekt s chybovou zprávou { "error": "Student not found" }
* **Popis:** Ověřte, že API vrací správný status kód a chybovou zprávu, pokud se pokusíte získat data o neexistujícím studentovi.

### **2. Testování metody POST**

#### **Testovací scénář 3: Vytvoření nového studenta**

* **Cíl:** Ověřit, zda metoda POST správně vytvoří nového studenta.
* **URL:** http://108.143.193.45:8080/api/v1/students/
* **Metoda:** POST
* **Vstupní data:**
  + JSON objekt obsahující údaje nového studenta  
     {  
     "id": 1,  
     "firstName": "Radim",  
     ”lastName” : “Černil”,  
     "email": "r.cernil@google.cz",  
     "age": 33  
     }
* **Očekávaný výstup:**
  + Status kód: 201 Created
  + Tělo odpovědi: JSON objekt s potvrzením vytvoření
  + {  
     "id": 1477,  
     "firstName": "Radim",  
     ”lastName” : “Černil”,  
     "email": "r.cernil@google.cz",  
     "age": 33  
     }
  + **Popis:** Ověřte, že nový student byl vytvořen a že vrácený JSON obsahuje správně nastavené údaje včetně unikátního id.

#### **Testovací scénář 4: Vytvoření studenta s neúplnými údaji**

* **Cíl:** Ověřit, jak API reaguje na pokus o vytvoření studenta s neúplnými údaji.
* **URL:** http://108.143.193.45:8080/api/v1/students/
* **Metoda:** POST
* **Vstupní data:**
  + JSON objekt s neúplnými údaji
  + {  
     "id": ,  
     "firstName": "Radim",  
     ”lastName” : “Černil”,  
     "email": "r.cernil@google.cz",  
     "age": 33  
     }
  + **Očekávaný výstup:**
  + Status kód: 400 Bad Request
  + Tělo odpovědi: JSON objekt s chybovou zprávou (např. { "error": "Invalid data" })
* **Popis:** Ověřte, že API správně zpracovává chyby v datech a vrací vhodný status kód a chybovou zprávu.

#### **Testovací scénář 5: Vytvoření studenta s neplatným emailem**

* **Cíl:** Ověřit, jak API reaguje na pokus o vytvoření studenta s neplatným emailem.
* **URL:** http://108.143.193.45:8080/api/v1/students/
* **Metoda:** POST
* **Vstupní data:**
  + JSON objekt s neúplnými údaji
  + {  
     "id": 1,  
     "firstName": "Radim",  
     ”lastName” : “Černil”,  
     "email": "r.cernilgoogle.cz",  
     "age": 33  
     }

**Očekávaný výstup:**

* + Status kód: 400 Bad Request
  + Tělo odpovědi: JSON objekt s chybovou zprávou (např.

{ "error": "Invalid data"

"message": "Email address is invalid"

})

* **Popis:** Ověřte, že API správně zpracovává chyby v datech a vrací vhodný status kód a chybovou zprávu.

#### **Testovací scénář 6: Vytvoření studenta s neplatným jménem**

* **Cíl:** Ověřit, jak API reaguje na pokus o vytvoření studenta s neplatným jménem.
* **URL:** http://108.143.193.45:8080/api/v1/students/
* **Metoda:** POST
* **Vstupní data:**
  + JSON objekt s neúplnými údaji
  + {  
     "id": 1,  
     "firstName": "/\*-+",  
     ”lastName” : “Černil”,  
     "email": "r.cernilgoogle.cz",  
     "age": 33  
     }
  + **Očekávaný výstup:**
  + Status kód: 400 Bad Request
  + Tělo odpovědi: JSON objekt s chybovou zprávou (např.

{ "error": "Invalid data"

"message": "Name is invalid"

})

* **Popis:** Ověřte, že API správně zpracovává chyby v datech a vrací vhodný status kód a chybovou zprávu.

#### **Testovací scénář 7: Vytvoření studenta s neplatným příjmením**

* **Cíl:** Ověřit, jak API reaguje na pokus o vytvoření studenta s neplatným příjmením.
* **URL:** http://108.143.193.45:8080/api/v1/students/
* **Metoda:** POST
* **Vstupní data:**
  + JSON objekt s neúplnými údaji
  + {  
     "id": 1,  
     "firstName": "Radim”,  
     ”lastName” : “@#$%^”,  
     "email": "r.cernil@google.cz",  
     "age": 33  
     }
  + **Očekávaný výstup:**
  + Status kód: 400 Bad Request
  + Tělo odpovědi: JSON objekt s chybovou zprávou (např.

{ "error": "Invalid data"

"message": "Surname is invalid"

})

* **Popis:** Ověřte, že API správně zpracovává chyby v datech a vrací vhodný status kód a chybovou zprávu.

### **3. Testování metody DELETE**

#### **Testovací scénář 8: Smazání existujícího studenta**

* **Cíl:** Ověřit, zda metoda DELETE správně smaže existujícího studenta.
* **URL:** http://108.143.193.45:8080/api/v1/students/1478
* **Metoda:** DELETE
* **Vstupní data:**
  + id existujícího studenta (1478)
* **Očekávaný výstup:**
  + Status kód: 200 OK nebo 204 No Content
  + Tělo odpovědi: Žádné nebo potvrzovací zpráva
* **Popis:** Ověřte, že student byl úspěšně smazán.

#### **Testovací scénář 9: Smazání neexistujícího studenta**

* **Cíl:** Ověřit, jak API reaguje na pokus o smazání neexistujícího studenta.
* **URL:** http://108.143.193.45:8080/api/v1/students/1593
* **Metoda:** DELETE
* **Vstupní data:**
  + id neexistujícího studenta (1593)
* **Očekávaný výstup:**
  + Status kód: 404 Not Found
  + Tělo odpovědi: JSON objekt s chybovou zprávou (např. { "error": "Student not found" })
* **Popis:** Ověřte, že API správně zpracovává pokus o smazání neexistujícího studenta a vrací vhodný status kód a chybovou zprávu.

# EXEKUCE TESTŮ

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

### **1. Testovací případy GET**

### ***1: Získání dat o existujícím studentovi***

***Cíl:****Ověřit, zda metoda GET vrací správná data pro existujícího studenta.*

***URL:****http://108.143.193.45:8080/api/v1/students/319*

***Metoda:****GET*

***Vstupní data:****ID existujícího studenta: 319*

***Kroky:***

1. *Otevřete nástroj pro testování API* (např. Postman).
2. *Nastavte metodu HTTP na GET.*
3. *Zadejte URL: http://108.143.193.45:8080/api/v1/students/319*
4. *Neposílejte žádné tělo požadavku, protože je to GET požadavek.*
5. *Odešlete požadavek.*

***Očekávaný výstup:***

***Status kód:*** *200 OK*

***Tělo odpovědi:*** *JSON objekt obsahující data studenta:  
  
{*

*"id": 319,*

*"firstName": "Test",*

*"lastName": "TESTSDF",*

*"email": "jjohn34@yourmail.com",*

*"age": 34*

*}*

### **Status:** Pass

### **2: Získání dat o neexistujícím studentovi**

**Cíl:**Ověřit, jak API reaguje na požadavek o neexistujícím studentovi.

**URL:**http://108.143.193.45:8080/api/v1/students/199

**Metoda:**GET

**Vstupní data:**ID neexistujícího studenta: 199

**Kroky:**

1. Otevřete nástroj pro testování API (např. Postman).
2. Nastavte metodu HTTP na GET.
3. Zadejte URL: http://108.143.193.45:8080/api/v1/students/199
4. Neposílejte žádné tělo požadavku, protože je to GET požadavek.
5. Odešlete požadavek.

**Očekávaný výstup:**

* **Status kód:** 404 Not Found

**Tělo odpovědi:** JSON objekt s chybovou zprávou:  
json  
Zkopírovat kód  
{

"error": "Student not found"

}

### **Status:** Fail

### **2. Testovací případy POST**

### **3: Vytvoření nového studenta**

**Cíl:**Ověřit, zda metoda POST správně vytvoří nového studenta.

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

**Metoda:**POST

**Vstupní data:**JSON objekt obsahující údaje nového studenta:

{

"id": 1,

"firstName": "Radim",

"lastName": "Černil",

"email": "r.cernil@google.cz",

"age": 33

}

**Kroky:**

1. Otevřete nástroj pro testování API (např. Postman).
2. Nastavte metodu HTTP na POST.
3. Zadejte URL: http://108.143.193.45:8080/api/v1/students/
4. Vložte do těla požadavku JSON objekt s údaji nového studenta, jak je uvedeno výše.
5. Odešlete požadavek.

**Očekávaný výstup:**

* **Status kód:** 201 Created

**Tělo odpovědi:** JSON objekt s potvrzením vytvoření studenta:  
{

"id": 1477,

"firstName": "Radim",

"lastName": "Černil",

"email": "r.cernil@google.cz",

"age": 33

}

### **Status:** Pass

### **4: Vytvoření studenta s neúplnými údaji**

**Cíl:**Ověřit, jak API reaguje na pokus o vytvoření studenta s neúplnými nebo neplatnými údaji.

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

**Metoda:**POST

**Vstupní data:**JSON objekt s neúplnými údaji (neúplné pole id):

{

"id": ,

"firstName": "Radim",

"lastName": "Černil",

"email": "r.cernil@google.cz",

"age": 33

}

**Kroky:**

1. Otevřete nástroj pro testování API (např. Postman).
2. Nastavte metodu HTTP na POST.
3. Zadejte URL: http://108.143.193.45:8080/api/v1/students/
4. Vložte do těla požadavku JSON objekt s neúplnými údaji, jak je uvedeno výše.
5. Odešlete požadavek.

**Očekávaný výstup:**

* **Status kód:** 400 Bad Request

**Tělo odpovědi:** JSON objekt s chybovou zprávou, např.:  
  
{

"error": "Bad Request"

}

### **Status:** Pass

### **5: Vytvoření studenta s neplatným emailem**

**Cíl:**Ověřit, jak API reaguje na pokus o vytvoření studenta s neplatnými údaji, konkrétně s neplatnou e-mailovou adresou.

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

**Metoda:**POST

**Vstupní data:**JSON objekt s neplatnou e-mailovou adresou (chybí znak @):

{

"id": 1,

"firstName": "Radim",

"lastName": "Černil",

"email": "r.cernilgoogle.cz",

"age": 33

}

**Kroky:**

1. Otevřete nástroj pro testování API (např. Postman).
2. Nastavte metodu HTTP na POST.
3. Zadejte URL: http://108.143.193.45:8080/api/v1/students/
4. Vložte do těla požadavku JSON objekt s neplatnými údaji, jak je uvedeno výše.
5. Odešlete požadavek.

**Očekávaný výstup:**

* **Status kód:** 400 Bad Request

**Tělo odpovědi:** JSON objekt s chybovou zprávou, např.:  
  
{

"error": "Invalid data",

"message": "Email address is invalid"

}

### **Status:** Fail

### **6: Vytvoření studenta s neplatným jménem**

**Cíl:**Ověřit, jak API reaguje na pokus o vytvoření studenta s neplatným jménem.

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

**Metoda:**POST

**Vstupní data:**JSON objekt s neplatným jménem (např. obsahující speciální znaky):

json

Zkopírovat kód

{

"id": 1,

"firstName": "/\*-+",

"lastName": "Černil",

"email": "r.cernilgoogle.cz",

"age": 33

}

**Kroky:**

1. Otevřete nástroj pro testování API (např. Postman)..
2. Nastavte metodu HTTP na POST.
3. Zadejte URL: http://108.143.193.45:8080/api/v1/students/
4. Vložte do těla požadavku JSON objekt s neplatnými údaji, jak je uvedeno výše.
5. Odešlete požadavek.

**Očekávaný výstup:**

* **Status kód:** 400 Bad Request

**Tělo odpovědi:**  
{

"error": "Invalid data",

"message": "Name is invalid"

}

### **Status:** Fail

### **7: Vytvoření studenta s neplatným příjmením**

**Cíl:**Ověřit, jak API reaguje na pokus o vytvoření studenta s neplatným příjmením.

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

**Metoda:**POST

**Vstupní data:**JSON objekt s neplatným příjmením obsahujícím speciální znaky:

{

"id": 1,

"firstName": "Radim",

"lastName": "@#$%^",

"email": "r.cernil@google.cz",

"age": 33

}

**Kroky k reprodukci:**

1. Otevřete nástroj pro testování API (např. Postman).
2. Nastavte metodu HTTP na POST.
3. Zadejte URL: http://108.143.193.45:8080/api/v1/students/
4. Vložte do těla požadavku JSON objekt s neplatnými údaji, jak je uvedeno výše.
5. Odešlete požadavek.

**Očekávaný výstup:**

* **Status kód:** 400 Bad Request

**Tělo odpovědi:**  
{

"error": "Invalid data",

"message": "Surname is invalid"

}

### **Status:** Fail

### **3. Testovací případy DELETE**

### **8: Smazání existujícího studenta**

**Cíl:**Ověřit, zda metoda DELETE správně smaže existujícího studenta.

**URL:**http://108.143.193.45:8080/api/v1/students/1593

**Metoda:**DELETE

**Vstupní data:**ID existujícího studenta: 1593

**Kroky k reprodukci:**

1. Otevřete nástroj pro testování API (např. Postman).
2. Nastavte metodu HTTP na DELETE.
3. Zadejte URL: http://108.143.193.45:8080/api/v1/students/1593
4. Odešlete požadavek.

**Očekávaný výstup:**

* **Status kód:** 200 OK nebo 204 No Content
* **Tělo odpovědi:**
  + **Pro 200 OK:** Může obsahovat potvrzovací zprávu, např. { "message": "Student deleted successfully" }

### **Status:** Pass

### **9: Smazání neexistujícího studenta**

**Cíl:**Ověřit, jak API reaguje na pokus o smazání neexistujícího studenta.

**URL:**http://108.143.193.45:8080/api/v1/students/1593

**Metoda:**DELETE

**Vstupní data:**ID neexistujícího studenta: 1593

**Kroky k reprodukci:**

1. Otevřete nástroj pro testování API (např. Postman).
2. Nastavte metodu HTTP na DELETE.
3. Zadejte URL: http://108.143.193.45:8080/api/v1/students/1593
4. Odešlete požadavek.

**Očekávaný výstup:**

* **Status kód:** 404 Not Found

**Tělo odpovědi:**  
{

"error": "Student not found"

}

### **Status:** Fail

# BUG REPORT

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

### **ID:** BR-001

**Název:** API vrací chybu 500 Internal Server Error při požadavku na neexistujícího studenta

**Popis:**API vrací chybu 500 Internal Server Error, když se uživatel pokusí získat data o neexistujícím studentovi pomocí metody GET. Očekává se, že API vrátí status kód 404 Not Found a jasnou chybovou zprávu, jako je "error": "Student not found". Namísto toho dojde k interní chybě serveru, což naznačuje, že API není správně ošetřeno pro tento případ.

**Kroky k reprodukci:**

1. Otevřete nástroj pro testování API (např. Postman).
2. Nastavte metodu HTTP na GET.
3. Zadejte následující URL: http://108.143.193.45:8080/api/v1/students/199 (kde ID 199 neexistuje).
4. Odešlete požadavek.

**Očekávaný výsledek:**

* Status kód: 404 Not Found

Tělo odpovědi:  
{

"error": "Student not found"

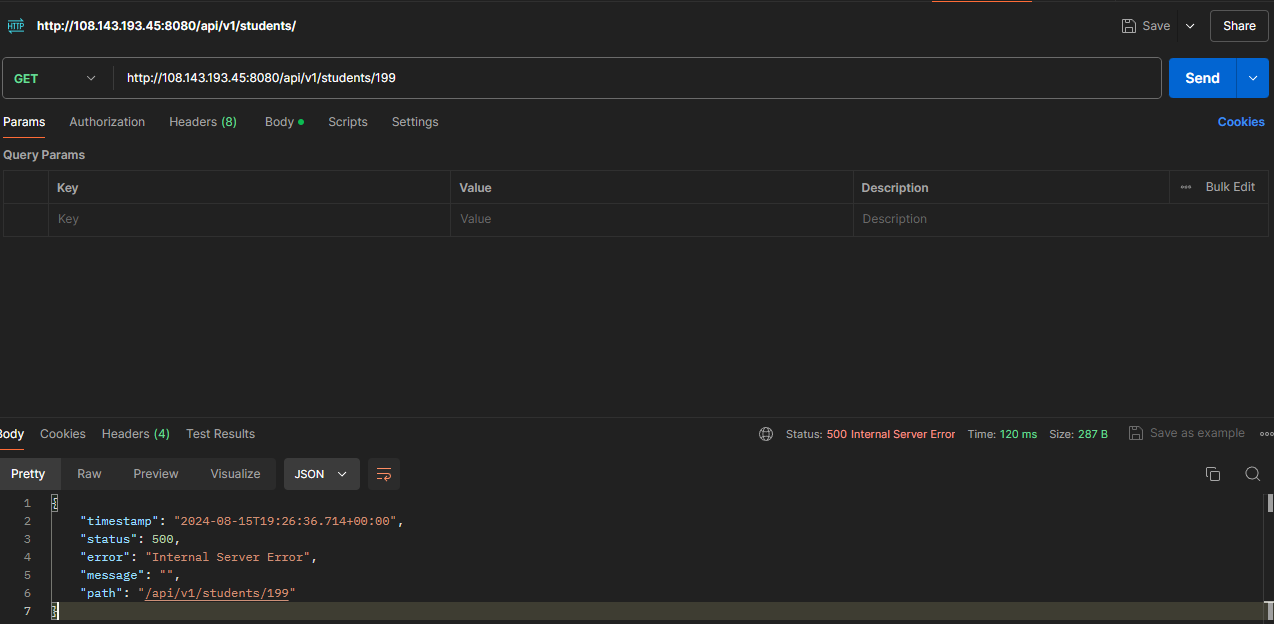
}

**Skutečný výsledek:**

* Status kód: 500 Internal Server Error
* Tělo odpovědi obsahuje zprávu o interní chybě serveru.

**Přílohy:**

* Screenshot chybové zprávy z nástroje pro testování API.



**Navrhované řešení:**

* Zajistit, aby při neexistenci studenta API vracelo status kód 404 Not Found a odpovídající chybovou zprávu.

### **ID:** BR-002

**Název:**API umožňuje vytvoření studenta s neplatnou e-mailovou adresou (chybějící znak @)

**Popis:**API umožňuje vytvoření nového studenta i v případě, že e-mailová adresa neobsahuje znak @. Očekává se, že API by mělo správně validovat e-mailové adresy a odmítnout neplatné e-maily. Místo toho, záznam s neplatnou e-mailovou adresou (např. r.cernilgoogle.cz) je úspěšně uložen do databáze, což ukazuje na nedostatečnou validaci údajů na serverové straně.

**Kroky k reprodukci:**

1. Otevřete nástroj pro testování API (např. Postman).
2. Nastavte metodu HTTP na POST.
3. Zadejte URL: http://108.143.193.45:8080/api/v1/students/
4. Vložte do těla požadavku následující JSON objekt s neplatnou e-mailovou adresou:  
     
   {

"id": 1,

"firstName": "Radim",

"lastName": "Černil",

"email": "r.cernilgoogle.cz",

"age": 33

}

1. Odešlete požadavek.

**Očekávaný výsledek:**

* **Status kód:** 400 Bad Request

**Tělo odpovědi:**  
{

"error": "Invalid data",

"message": "Email address is invalid"

}

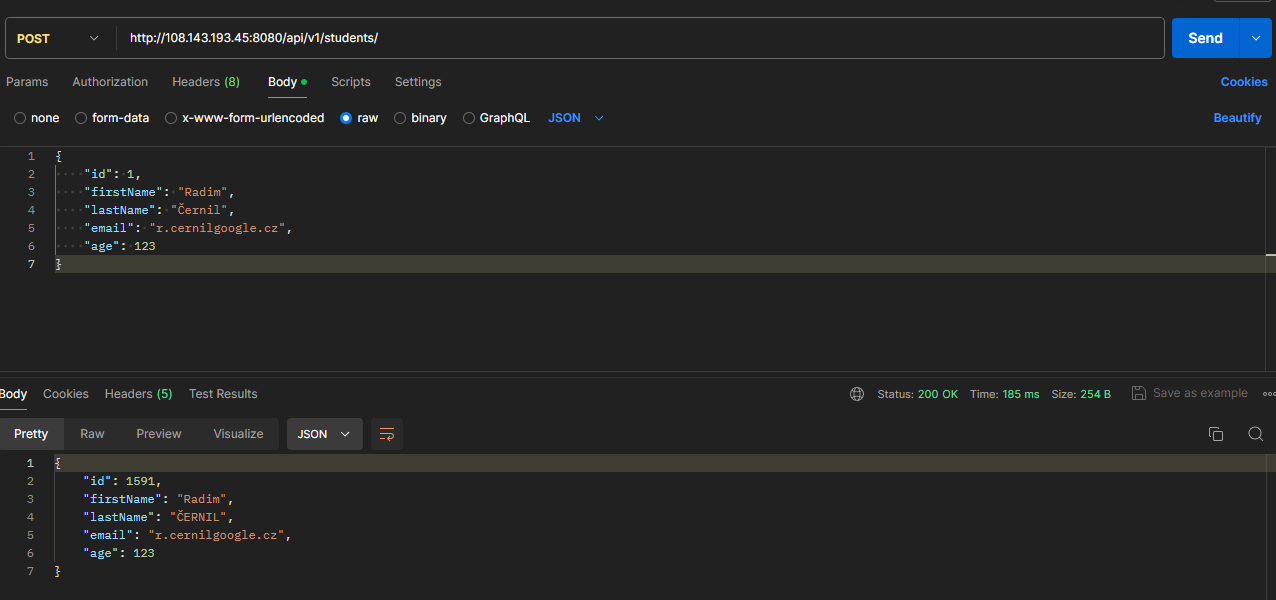
* Nový student by neměl být vytvořen a záznam by neměl být uložen do databáze.

**Skutečný výsledek:**

* **Status kód:** 201 Created
* **Tělo odpovědi:** Úspěšné vytvoření záznamu, i když e-mailová adresa je neplatná.
* Nový student s neplatnou e-mailovou adresou je uložen do databáze.

**Přílohy:**

* Screenshot odpovědi z nástroje pro testování API, který ukazuje úspěšné vytvoření záznamu i s neplatným e-mailem.

****

**Navrhované řešení:**

* Implementovat validaci e-mailových adres, která zajistí, že e-mail obsahuje znak @ a splňuje základní formát.
* Ujistit se, že API vrací status kód 400 Bad Request s odpovídající chybovou zprávou, pokud e-mailová adresa není platná.

### **ID:** BR-003

**Název:**API umožňuje vytvoření studenta s neplatným jménem obsahujícím speciální znaky

**Popis:**API umožňuje vytvoření nového studenta i v případě, že jméno obsahuje speciální znaky, jako jsou /\*-+. Očekává se, že API by mělo správně validovat jména a odmítnout neplatné znaky. Místo toho, záznam s neplatným jménem je úspěšně uložen do databáze, což ukazuje na nedostatečnou validaci údajů na serverové straně.

**Kroky k reprodukci:**

1. Otevřete nástroj pro testování API (např. Postman).
2. Nastavte metodu HTTP na POST.
3. Zadejte URL: http://108.143.193.45:8080/api/v1/students/
4. Vložte do těla požadavku následující JSON objekt s neplatným jménem:  
     
   {

"id": 1,

"firstName": "/\*-+",

"lastName": "Černil",

"email": "r.cernilgoogle.cz",

"age": 33

}

1. Odešlete požadavek.

**Očekávaný výsledek:**

* **Status kód:** 400 Bad Request

**Tělo odpovědi:**  
{

"error": "Invalid data",

"message": "Name is invalid"

}

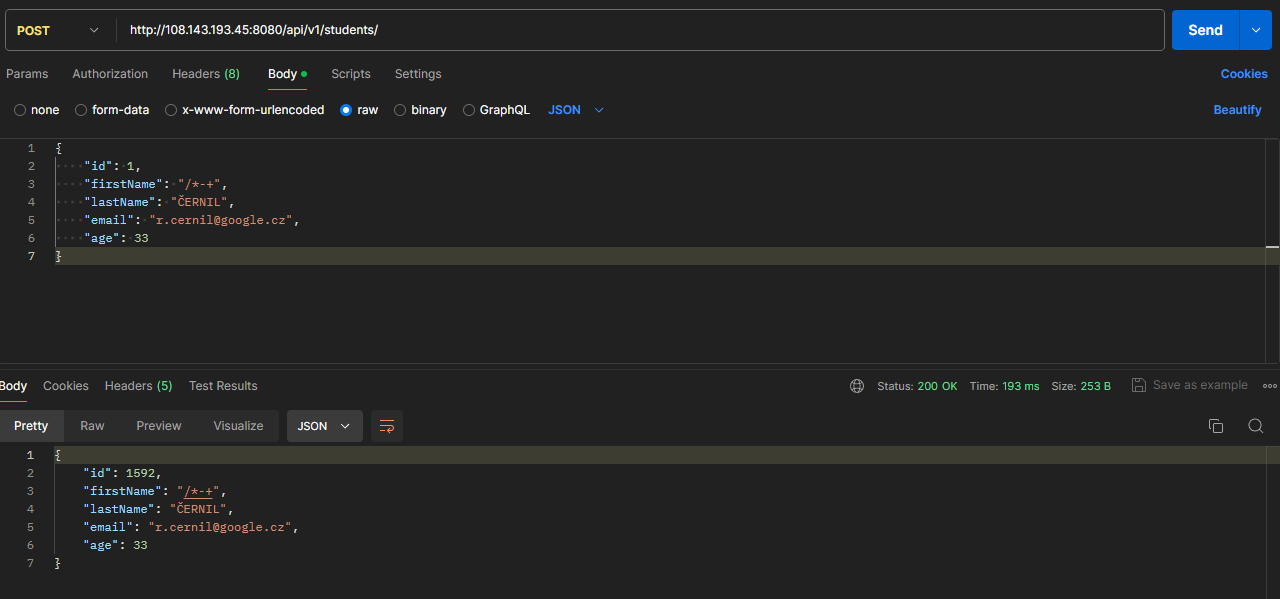
* Nový student by neměl být vytvořen a záznam by neměl být uložen do databáze.

**Skutečný výsledek:**

* **Status kód:** 201 Created
* **Tělo odpovědi:** Úspěšné vytvoření záznamu, i když jméno obsahuje neplatné znaky.
* Nový student s neplatným jménem je uložen do databáze.

**Přílohy:**

* Screenshot odpovědi z nástroje pro testování API, který ukazuje úspěšné vytvoření záznamu i s neplatným jménem.



**Navrhované řešení:**

* Implementovat validaci jmen, která zajistí, že jména neobsahují speciální znaky, které nejsou povoleny.
* Ujistit se, že API vrací status kód 400 Bad Request s odpovídající chybovou zprávou, pokud jméno obsahuje neplatné znaky.

.

### **ID:** BR-004

**Název:**API umožňuje vytvoření studenta s neplatným příjmením obsahujícím speciální znaky

**Popis:**API umožňuje vytvoření nového studenta i v případě, že příjmení obsahuje speciální znaky, jako jsou @#$%^. Očekává se, že API by mělo správně validovat příjmení a odmítnout neplatné znaky. Místo toho, záznam s neplatným příjmením je úspěšně uložen do databáze, což ukazuje na nedostatečnou validaci údajů na serverové straně.

**Kroky k reprodukci:**

1. Otevřete nástroj pro testování API (např. Postman).
2. Nastavte metodu HTTP na POST.
3. Zadejte URL: <http://108.143.193.45:8080/api/v1/students/>
4. Vložte do těla požadavku následující JSON objekt s neplatným příjmením:  
   {

"id": 1,

"firstName": "Radim",

"lastName": "@#$%^",

"email": "r.cernil@google.cz",

"age": 33

}

1. Odešlete požadavek.

**Očekávaný výsledek:**

* **Status kód:** 400 Bad Request

**Tělo odpovědi:**  
{

"error": "Invalid data",

"message": "Surname is invalid"

}

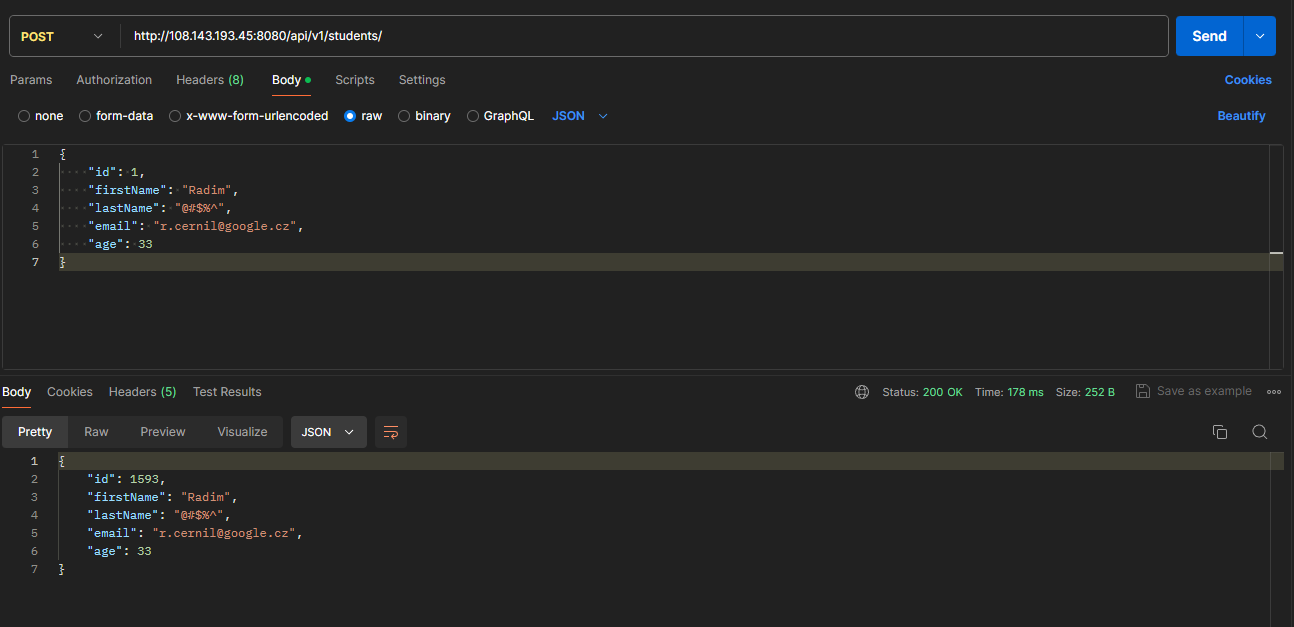
* Nový student by neměl být vytvořen a záznam by neměl být uložen do databáze.

**Skutečný výsledek:**

* **Status kód:** 200 OK
* **Tělo odpovědi:** Úspěšné vytvoření záznamu, i když příjmení obsahuje neplatné znaky.
* Nový student s neplatným příjmením je uložen do databáze.

**Přílohy:**

* Screenshot odpovědi z nástroje pro testování API, který ukazuje úspěšné vytvoření záznamu i s neplatným příjmením.

****

**Navrhované řešení:**

* Implementovat validaci příjmení, která zajistí, že příjmení neobsahuje speciální znaky, které nejsou povoleny.
* Ujistit se, že API vrací status kód 400 Bad Request s odpovídající chybovou zprávou, pokud příjmení obsahuje neplatné znaky.

.

### **ID:** BR-005

**Název:**API vrací chybu 500 Internal Server Error při pokusu o smazání neexistujícího studenta

**Popis:**API vrací chybu 500 Internal Server Error při pokusu o smazání neexistujícího studenta s ID 1593. Očekává se, že API vrátí status kód 404 Not Found a jasnou chybovou zprávu, například { "error": "Student not found" }. Místo toho dojde k interní chybě serveru, což naznačuje, že API není správně ošetřeno pro tento případ a dochází k nečekaným chybám na serverové straně.

**Kroky k reprodukci:**

1. Otevřete nástroj pro testování API (např. Postman).
2. Nastavte metodu HTTP na DELETE.
3. Zadejte URL: http://108.143.193.45:8080/api/v1/students/1593 (kde ID 1593 neexistuje).
4. Odešlete požadavek.

**Očekávaný výsledek:**

* **Status kód:** 404 Not Found

**Tělo odpovědi:**json  
Zkopírovat kód  
{

"error": "Student not found"

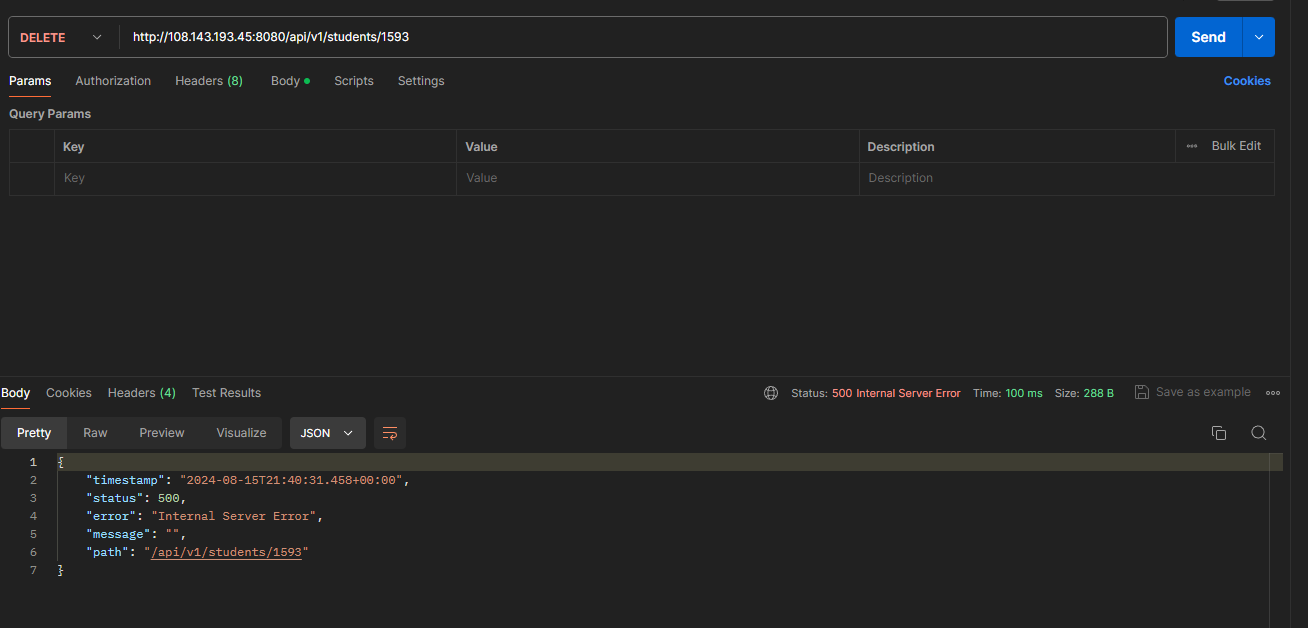
}

**Skutečný výsledek:**

* **Status kód:** 500 Internal Server Error
* **Tělo odpovědi:** Obsahuje zprávu o interní chybě serveru.

**Přílohy:**

* Screenshot chybové zprávy z nástroje pro testování API, který ukazuje interní chybu serveru při pokusu o smazání neexistujícího studenta.

****

**Navrhované řešení:**

* Ujistit se, že API vrací status kód 404 Not Found s odpovídající chybovou zprávou, pokud student s daným ID neexistuje.
* Zajistit, aby API správně identifikovalo neexistující záznamy a vracelo odpovídající zpětnou vazbu bez vyvolání interní chyby serveru.