FINÁLNÍ PROJEKT
č.1

Autor: Patrik Perutka
Datum: 24. 10. 2024

OBSAH

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 ověřil funkčnost aplikace.

TESTCASE #001
Název: “Test metody GET zobrazující záznamy všech studentů v databázi”
Prostředí: Programy Postman, MySQL Workbench

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na GET.
3) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
4) Odeslat požadavek pomocí tlačítka Send.
5) Dojet na konec vygenerované databáze a zjistit poslední dostupné ID (nejvyšší hodnota).
6) Otevřít program MySQL Workbench.
7) Přihlásit se k databázi pomocí přihlašovacích údajů dostupných v zadání projektu.
8) Pomocí SQL dotazu *SELECT * FROM student;* nejprve ověřit dostupnost databáze.
9) Pomocí SQL dotazu *SELECT MAX(id) FROM student*; zjistit nejvyšší hodnotu ID v databázi.
10) Porovnat nejvyšší získaná ID pomocí programu Postman a programu MySQL.

Očekávaný výsledek:
Program Postman zobrazí kompletní databázi studentů končící studentem s nejvyšším ID a zobrazí Status Code 200 OK. Program MySQL zobrazí kompletní databázi studentů a pomocí druhého dotazu následně zobrazí záznam s nejvyšším dostupným ID. Toto ID bude identické s nejvyšším ID nalezeným programem Postman, čímž nepřímo ověříme zobrazení kompletní databáze.

TESTCASE #002
Název: “Test metody GET zobrazující záznam studenta se zvoleným platným ID”
Prostředí: Programy Postman, MySQL Workbench

Kroky:

1) Otevřít program MySQL Workbench.
2) Přihlásit se k databázi pomocí přihlašovacích údajů dostupných v zadání projektu.
3) Pomocí SQL dotazu *SELECT MAX(id) FROM student*; zjistit nejvyšší hodnotu ID v databázi.
4) Zvolit si validní ID v dostupném rozmezí databáze.
5) Pomocí SQL dotazu *SELECT * FROM student WHERE id=666* (zvolené ID) zobrazit záznam studenta se zvoleným ID.
6) V případě neexistujícího záznamu se zvoleným ID opakovat kroky 4 a 5.
7) Otevřít program Postman.
8) Nastavit metodu na GET.
9) Zaslat požadavek na REST API - do kolonky URL zadat "http://108.143.193.45:8080/api/v1/students/666"
10) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Program MySQL zobrazí jeden záznam konkrétního studenta se zvoleným ID. Program Postman následně zobrazí identický záznam studenta jako MySQL a zobrazí Status Code 200 OK.

TESTCASE #003
Název: "Test metody GET se zadáním neexistujícího ID"
Prostředí: Programy Postman, MySQL Workbench

Kroky:

1) Otevřít program MySQL Workbench.
2) Přihlásit se k databázi pomocí přihlašovacích údajů dostupných v zadání projektu.
3) Pomocí SQL dotazu *SELECT MAX(id) FROM student*; zjistit nejvyšší hodnotu ID v databázi.
4) Zvolit si neexistující ID přesahující momentální limit databáze.
5) Zadat SQL dotaz *SELECT * FROM student WHERE id=6666;* (zvolené neexistující ID)
6) Otevřít program Postman.
7) Nastavit metodu na GET.
8) Zaslat požadavek na REST API - do kolonky URL zadat "http://108.143.193.45:8080/api/v1/students/6666"
9) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Program MySQL zobrazí nulový záznam - hledaný student neexistuje. Program Postman zobrazí Status Code 404 Not Found s chybovou hláškou, např. ID was not found.

TESTCASE #004
Název: “Test metody GET se zadáním ID ve formě stringu”
Prostředí: Programy Postman, MySQL Workbench

Kroky:

1) Otevřít program MySQL Workbench.
2) Přihlásit se k databázi pomocí přihlašovacích údajů dostupných v zadání projektu.
3) Zvolit si neplatné ID - např. string
4) Zadat SQL dotaz *SELECT * FROM student WHERE id=”Patrik”;* (zvolené neplatné ID, zde ve formě stringu)
5) Otevřít program Postman.
6) Nastavit metodu na GET.
7) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/Patrik”
8) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Program MySQL zobrazí nulový záznam - nečíselné ID neexistuje a nelze vyhledat. Program Postman zobrazí Status Code 400 Bad Request s chybovou hláškou, např. Invalid ID - must be a number.

TESTCASE #005
Název: “Test metody GET se zadáním ID ve formě special characters”
Prostředí: Programy Postman, MySQL Workbench

Kroky:

1) Otevřít program MySQL Workbench.
2) Přihlásit se k databázi pomocí přihlašovacích údajů dostupných v zadání projektu.
3) Zvolit si neplatné ID - např. %”?
4) Zadat SQL dotaz *SELECT * FROM student WHERE id=”*%”?*”;* (zvolené neplatné ID, zde s použitím special characters)
5) Otevřít program Postman.

6) Nastavit metodu na GET.
7) Zaslat požadavek na REST API - do kolonky URL zadat "http://108.143.193.45:8080/api/v1/students/%**?**"
8) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Program MySQL zobrazí nulový záznam - nečíselné ID neexistuje a nelze vyhledat. Program Postman zobrazí Status Code 400 Bad Request s chybovou hláškou, např. Invalid ID - cannot contain special characters.

TESTCASE #006
Název: "Test metody POST s kompletně a správně vyplněnými parametry"
Prostředí: Programy Postman, MySQL Workbench

Kroky:
1) Otevřít program Postman.
2) Nastavit metodu na POST.
3) V možnostech níže zvolit "Body" a následně možnost "raw" a "JSON".
4) Zadat parametry ve formátu: { "first_name: "jmeno", "last_name": "prijmeni", "email": "emailova_adresa", "age": ***int*** }, např. { "first_name: "Vincent", "last_name": "Vega", "email": "zeds@dead.com", "age": ***50*** }
5) Zaslat požadavek na REST API - do kolonky URL zadat "http://108.143.193.45:8080/api/v1/students/"
6) Odeslat požadavek pomocí tlačítka Send.
7) Zjistit ID vygenerované databází po vložení údajů.
8) Otevřít program MySQL Workbench.
9) Přihlásit se k databázi pomocí přihlašovacích údajů dostupných v zadání projektu.
10) Zadat SQL dotaz *"SELECT * FROM student WHERE id=<vygenerované ID>;"*

Očekávaný výsledek:
Postman zobrazí Status Code 200 OK a zobrazí zadané údaje, k nimž přidá automaticky vygenerované ID. MySQL Workbench zobrazí jeden záznam s údaji, které jsme přes metodu POST v předchozím kroku přidali.

TESTCASE #007
Název: "Test metody POST s jedním nevyplněným parametrem"
Prostředí: Program Postman

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na POST.
3) V možnostech níže zvolit “Body” a následně možnost “raw” a “JSON”.
4) Zadat parametry ve formátu: { “first_name: “jmeno”, “last_name”: “prijmeni”, “age”: ***int*** } - vynechat parametr email, např. { “first_name: “Vincent”, “last_name”: “Vega”, “age”: ***50*** }
5) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
6) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman zobrazí Status Code 400 Bad Request s chybovou hláškou, např. missing a required field - email.

TESTCASE #008
Název: “Test metody POST s nesprávně vyplněným parametrem”
Prostředí: Program Postman

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na POST.
3) V možnostech níže zvolit “Body” a následně možnost “raw” a “JSON”.
4) Zadat parametr “age” v nesprávném formátu: { “first_name: “jmeno”, “last_name”: “prijmeni”, “age”: ***str***} např. { “first_name: “Vincent”, “last_name”: “Vega”,“email”: “zeds@dead.com”, “age”: ***padesát*** }
5) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
6) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman zobrazí Status Code 400 Bad Request s chybovou hláškou, např. invalid syntax nebo invalid data type.

TESTCASE #009
Název: “Test metody POST s nevyplněnými parametry”
Prostředí: Program Postman

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na POST.
3) V možnostech níže zvolit “Body” a následně možnost “raw” a “JSON”.
4) Zadat prázdné parametry - pouze {}
5) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
6) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman zobrazí Status Code 400 Bad Request s chybovou hláškou, např. Body cannot be empty.

TESTCASE #010
Název: “Test metody POST se zadáním v databázi neexistujícího parametru”
Prostředí: Program Postman

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na POST.
3) V možnostech níže zvolit “Body” a následně možnost “raw” a “JSON”.
4) Zadat parametry ve formátu: { “first_name: “jmeno”, “last_name”: “prijmeni”, “email”: “emailova_adresa”, “gender”: “pohlavi”, “age”: ***int*** }, např. { “first_name: “Vincent”, “last_name”: “Vega”, “email”: “zeds@dead.com”, “gender”: “male”, “age”: ***50*** }
5) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
6) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman zobrazí Status Code 400 Bad Request s chybovou hláškou, např. Invalid parameter.

TESTCASE #011
Název: “Test metody POST se zadáním neexistujícího věku”
Prostředí: Program Postman

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na POST.
3) V možnostech níže zvolit “Body” a následně možnost “raw” a “JSON”.
4) Zadat parametry ve formátu: { “first_name: “jmeno”, “last_name”: “prijmeni”, “age”: ***negative int*** } - zadat parametr věk jako záporné číslo, např. { “first_name: “Vincent”, “last_name”: “Vega”, “email”: “zeds@dead.com”, “age”: ***-50*** }
5) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
6) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman zobrazí Status Code 400 Bad Request s chybovou hláškou, např. Parameter “age” cannot be a negative number.

TESTCASE #012
Název: “Test metody POST se zadáním nadbytečně či nelogicky dlouhého údaje”
Prostředí: Program Postman

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na POST.
3) V možnostech níže zvolit “Body” a následně možnost “raw” a “JSON”.
4) Zadat parametry ve formátu: { “first_name: “jmeno”, “last_name”: “prijmeni”, “age”: ***int*** } - jako parametr jméno zadat nereálně dlouhý údaj pro ověření maximální délky, např. {“first_name”: “Vincent”, “last_name”: “Vega”, “email”: “zeds@dead.com”, “age”: ***50*** }
5) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
6) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman zobrazí Status Code 413 Request Too Large s chybovou hláškou, např. Parameter “first_name” too long. Maximum length is 50 characters.

TESTCASE #013
Název: “Test metody DELETE se zadáním údajů a jejich následným vymazáním”
Prostředí: Programy Postman, MySQL Workbench

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na POST.
3) V možnostech níže zvolit “Body” a následně možnost “raw” a “JSON”.
4) Zadat parametry ve formátu: { “first_name: “jmeno”, “last_name”: “prijmeni”, “email”: “emailova_adresa”, “age”: ***int*** }, např. { “first_name: “Vincent”, “last_name”: “Vega”, “email”: “zeds@dead.com”, “age”: ***50*** }
5) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
6) Odeslat požadavek pomocí tlačítka Send.
7) Zjistit ID vygenerované databází po vložení údajů.
8) Otevřít program MySQL Workbench.
9) Přihlásit se k databázi pomocí přihlašovacích údajů dostupných v zadání projektu.
10) Zadat SQL dotaz *“SELECT * FROM student WHERE id=<vygenerované ID>;”*
11) Vrátit se zpět do programu Postman.
12) Nastavit metodu na DELETE.
13) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/” , za poslední lomítko doplnit vygenerované ID
14) Odeslat požadavek pomocí tlačítka Send.
15) V programu MySQL Workbench zadat dotaz *“SELECT * FROM student WHERE id=<vygenerované ID>;”*

Očekávaný výsledek:
Po vložení údajů pomocí metody POST vrátí Postman Status Code 200 OK a zobrazí údaje společně s novým vygenerovaným ID. Pomocí tohoto ID si v MySQL ověřím, že záznam je skutečně v databázi. Následně pomocí metody DELETE záznam v Postmanovi vymažu. Postman zobrazí Status Code 200 OK a prázdný záznam. MySQL si ověřím, že záznam byl opravdu smazán a program vrátil tabulku s null.

TESTCASE #014
Název: “Test metody DELETE s neexistujícím ID”
Prostředí: Program Postman

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na DELETE.
3) Zaslat požadavek na REST API - do kolonky URL zadat "http://108.143.193.45:8080/api/v1/students/99999"
4) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman vrátí Status Code 404 Not Found a chybovou hlášku, např. ID not found.

TESTCASE #015
Název: "Test metody DELETE bez zadání ID"
Prostředí: Program Postman
Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na DELETE.
3) Zaslat požadavek na REST API - do kolonky URL zadat "http://108.143.193.45:8080/api/v1/students/"
4) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman vrátí Status Code 405 Not Allowed a chybovou hlášku, např. You are not permitted to execute this action.

EXEKUCE TESTŮ

Testovací scénáře jsem provedl, přikládám výsledky testů.

TEST_EXEC #001
Název: “Test metody GET zobrazující záznamy všech studentů v databázi”
Prostředí: Programy Postman, MySQL Workbench

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na GET.
3) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
4) Odeslat požadavek pomocí tlačítka Send.
5) Dojet na konec vygenerované databáze a zjistit poslední dostupné ID (nejvyšší hodnota).
6) Otevřít program MySQL Workbench.
7) Přihlásit se k databázi pomocí přihlašovacích údajů dostupných v zadání projektu.
8) Pomocí SQL dotazu *SELECT * FROM student;* nejprve ověřit dostupnost databáze.
9) Pomocí SQL dotazu *SELECT MAX(id) FROM student*; zjistit nejvyšší hodnotu ID v databázi.
10) Porovnat nejvyšší získaná ID pomocí programu Postman a programu MySQL.

Očekávaný výsledek:
Program Postman zobrazí kompletní databázi studentů končící studentem s nejvyšším ID a zobrazí Status Code 200 OK. Program MySQL zobrazí kompletní databázi studentů a pomocí druhého dotazu následně zobrazí záznam s nejvyšším dostupným ID. Toto ID bude identické s nejvyšším ID nalezeným programem Postman, čímž nepřímo ověříme zobrazení kompletní databáze.

Skutečný výsledek:
Postman zobrazil Status Code 200 OK a kompletní databázi s nejvyšším ID 1958. MySQL zobrazil kompletní databázi a pomocí druhého dotazu zobrazil MAX ID 1958. Databáze je kompletní.

STATUS: **PASS**

TEST_EXEC #002
Název: “Test metody GET zobrazující záznam studenta se zvoleným platným ID”
Prostředí: Programy Postman, MySQL Workbench

Kroky:

1) Otevřít program MySQL Workbench.
2) Přihlásit se k databázi pomocí přihlašovacích údajů dostupných v zadání projektu.
3) Pomocí SQL dotazu *SELECT MAX(id) FROM student*; zjistit nejvyšší hodnotu ID v databázi.
4) Zvolit si validní ID v dostupném rozmezí databáze.
5) Pomocí SQL dotazu *SELECT * FROM student WHERE id=666* (zvolené ID) zobrazit záznam studenta se zvoleným ID.
6) V případě neexistujícího záznamu se zvoleným ID opakovat kroky 4 a 5.
7) Otevřít program Postman.
8) Nastavit metodu na GET.
9) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/666”
10) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Program MySQL zobrazí jeden záznam konkrétního studenta se zvoleným ID. Program Postman následně zobrazí identický záznam studenta jako MySQL a zobrazí Status Code 200 OK.

Skutečný výsledek:
Program MySQL zobrazil studenta se zvoleným ID s křestním jménem “velkej” a příjmením “TRAK”. Program Postman zobrazil Status Code 200 OK a identický záznam.

STATUS: **PASS**

TEST_EXEC #003

Název: “Test metody GET se zadáním neexistujícího ID”
Prostředí: Programy Postman, MySQL Workbench

Kroky:

1) Otevřít program MySQL Workbench.
2) Přihlásit se k databázi pomocí přihlašovacích údajů dostupných v zadání projektu.
3) Pomocí SQL dotazu *SELECT MAX(id) FROM student*; zjistit nejvyšší hodnotu ID v databázi.
4) Zvolit si neexistující ID přesahující momentální limit databáze.
5) Zadat SQL dotaz *SELECT * FROM student WHERE id=6666;* (zvolené neexistující ID)
6) Otevřít program Postman.
7) Nastavit metodu na GET.
8) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/6666”
9) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Program MySQL zobrazí nulový záznam - hledaný student neexistuje. Program Postman zobrazí Status Code 404 Not Found s chybovou hláškou, např. ID was not found.

Skutečný výsledek:
Program MySQL zobrazil nulový záznam. Program Postman zobrazil Status Code 500 Internal Server Error a pole “message” pro chybovou hlášku nechal prázdné.

STATUS: **FAIL**

TEST_EXEC #004
Název: “Test metody GET se zadáním ID ve formě stringu”
Prostředí: Programy Postman, MySQL Workbench

Kroky:

1) Otevřít program MySQL Workbench.
2) Přihlásit se k databázi pomocí přihlašovacích údajů dostupných v zadání projektu.
3) Zvolit si neplatné ID - např. string
4) Zadat SQL dotaz *SELECT * FROM student WHERE id=”Patrik”;* (zvolené neplatné ID, zde ve formě stringu)

5) Otevřít program Postman.
6) Nastavit metodu na GET.
7) Zaslat požadavek na REST API - do kolonky URL zadat "http://108.143.193.45:8080/api/v1/students/Patrik"
8) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Program MySQL zobrazí chybovou hlášku - nečíselné ID neexistuje a nelze vyhledat. Program Postman zobrazí Status Code 400 Bad Request s chybovou hláškou, např. Invalid ID - must be a number.

Skutečný výsledek:
Program MySQL zobrazil chybovou hlášku syntax error a dotaz neprovedl. Program Postman zobrazil Status Code 400 Bad Request bez chybové hlášky.

STATUS: **FAIL**

TEST_EXEC #005
Název: "Test metody GET se zadáním ID ve formě special characters"
Prostředí: Programy Postman, MySQL Workbench

Kroky:
1) Otevřít program MySQL Workbench.
2) Přihlásit se k databázi pomocí přihlašovacích údajů dostupných v zadání projektu.
3) Zvolit si neplatné ID - např. %"?
4) Zadat SQL dotaz *SELECT * FROM student WHERE id="%"?";* (zvolené neplatné ID, zde s použitím special characters)
5) Otevřít program Postman.
6) Nastavit metodu na GET.
7) Zaslat požadavek na REST API - do kolonky URL zadat "http://108.143.193.45:8080/api/v1/students/%"?"
8) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Program MySQL zobrazí nulový záznam - nečíselné ID neexistuje a nelze vyhledat. Program Postman zobrazí Status Code 400 Bad Request s chybovou hláškou, např. Invalid ID - cannot contain special characters.

Skutečný výsledek:
Program MySQL zobrazil chybovou hlášku syntax error a dotaz neprovedl. Program Postman zobrazil Status Code 400 Bad Request a část kódu v jazyce CSS (?). Webový prohlížeč Chrome zobrazil Status Code 400 Bad Request správně.

STATUS: **FAIL**

TEST_EXEC #006
Název: “Test metody POST s kompletně a správně vyplněnými parametry”
Prostředí: Programy Postman, MySQL Workbench

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na POST.
3) V možnostech níže zvolit “Body” a následně možnost “raw” a “JSON”.
4) Zadat parametry ve formátu: { “first_name: “jmeno”, “last_name”: “prijmeni”, “email”: “emailova_adresa”, “age”: ***int*** }, např. { “first_name: “Vincent”, “last_name”: “Vega”, “email”: “zeds@dead.com”, “age”: ***50*** }
5) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
6) Odeslat požadavek pomocí tlačítka Send.
7) Zjistit ID vygenerované databází po vložení údajů.
8) Otevřít program MySQL Workbench.
9) Přihlásit se k databázi pomocí přihlašovacích údajů dostupných v zadání projektu.
10) Zadat SQL dotaz *“SELECT * FROM student WHERE id=<vygenerované ID>;”*

Očekávaný výsledek:
Postman zobrazí Status Code 200 OK a zobrazí zadané údaje, k nimž přidá automaticky vygenerované ID. MySQL Workbench zobrazí jeden záznam s údaji, které jsme přes metodu POST v předchozím kroku přidali.

Skutečný výsledek:
Postman zobrazil Status Code 200 OK a zobrazil zadané údaje s doplněným ID 1961 (údaj příjmení byl přepsán do velkých písmen, pravděpodobně záměrně?). MySQL zobrazil kompletní záznam se správnými údaji.

STATUS: **PASS**

TEST_EXEC #007
Název: “Test metody POST s jedním nevyplněným parametrem”
Prostředí: Program Postman

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na POST.
3) V možnostech níže zvolit “Body” a následně možnost “raw” a “JSON”.
4) Zadat parametry ve formátu: { “first_name: “jmeno”, “last_name”: “prijmeni”, “age”: ***int*** } - vynechat parametr email, např. { “first_name: “Vincent”, “last_name”: “Vega”, “age”: ***50*** }
5) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
6) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman zobrazí Status Code 400 Bad Request s chybovou hláškou, např. missing a required field - email.

Skutečný výsledek:
Postman zobrazil Status Code 500 Internal Server Error bez chybové hlášky.

STATUS: **FAIL**

TEST_EXEC #008
Název: “Test metody POST s nesprávně vyplněným parametrem”
Prostředí: Program Postman

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na POST.
3) V možnostech níže zvolit “Body” a následně možnost “raw” a “JSON”.
4) Zadat parametr “age” v nesprávném formátu: { “first_name: “jmeno”, “last_name”: “prijmeni”, “age”: ***str***} např. { “first_name: “Vincent”, “last_name”: “Vega”,“email”: “zeds@dead.com”, “age”: ***padesát*** }
5) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
6) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman zobrazí Status Code 400 Bad Request s chybovou hláškou, např. invalid syntax nebo invalid data type.

Skutečný výsledek:
Postman zobrazil Status Code 400 Bad Request bez chybové hlášky.

STATUS: **FAIL**

TEST_EXEC #009
Název: “Test metody POST s nevyplněnými parametry”
Prostředí: Program Postman

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na POST.
3) V možnostech níže zvolit “Body” a následně možnost “raw” a “JSON”.
4) Zadat prázdné parametry - pouze {}
5) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
6) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman zobrazí Status Code 400 Bad Request s chybovou hláškou, např. Body cannot be empty.

Skutečný výsledek:
Postman zobrazil Status Code 500 Internal Server error bez chybové hlášky.

STATUS: **FAIL**

TEST_EXEC #010
Název: “Test metody POST se zadáním v databázi neexistujícího parametru”
Prostředí: Program Postman

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na POST.

3) V možnostech níže zvolit “Body” a následně možnost “raw” a “JSON”.
4) Zadat parametry ve formátu: { “first_name: “jmeno”, “last_name”: “prijmeni”, “email”: “emailova_adresa”, “gender”: “pohlavi”, “age”: ***int*** }, např. { “first_name: “Vincent”, “last_name”: “Vega”, “email”: “zeds@dead.com”, “gender”: “male”, “age”: ***50*** }
5) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
6) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman zobrazí Status Code 400 Bad Request s chybovou hláškou, např. Invalid parameter.

Skutečný výsledek:
Postman zobrazil Status Code 200 OK a vytvořil nový záznam, námi vymyšlený parametr zcela ignoroval a do databáze ho nezanesl.

STATUS: **FAIL**

TEST_EXEC #011
Název: “Test metody POST se zadáním neexistujícího věku”
Prostředí: Program Postman

Kroky:
1) Otevřít program Postman.
2) Nastavit metodu na POST.
3) V možnostech níže zvolit “Body” a následně možnost “raw” a “JSON”.
4) Zadat parametry ve formátu: { “first_name: “jmeno”, “last_name”: “prijmeni”, “age”: ***negative int*** } - zadat parametr věk jako záporné číslo, např. { “first_name: “Vincent”, “last_name”: “Vega”, “email”: “zeds@dead.com”, “age”: ***-50*** }
5) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
6) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman zobrazí Status Code 400 Bad Request s chybovou hláškou, např. Parameter “age” cannot be a negative number.

Skutečný výsledek:
Postman zobrazil Status Code 200 OK a vytvořil nový záznam s neplatným, záporným věkem.

STATUS: **FAIL**

TEST_EXEC #012
Název: “Test metody POST se zadáním nadbytečně či nelogicky dlouhého údaje”
Prostředí: Program Postman

Kroky:
1) Otevřít program Postman.
2) Nastavit metodu na POST.
3) V možnostech níže zvolit “Body” a následně možnost “raw” a “JSON”.
4) Zadat parametry ve formátu: { “first_name: “jmeno”, “last_name”: “prijmeni”, “age”: ***int*** } - jako parametr jméno zadat nereálně dlouhý údaj pro ověření maximální délky, např. {“first_name”: “Vincent”, “last_name”: “Vega”, “email”: “zeds@dead.com”, “age”: ***50*** }
5) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
6) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman zobrazí Status Code 413 Request Too Large s chybovou hláškou, např. Parameter “first_name” too long. Maximum length is 50 characters.

Skutečný výsledek:
Postman zobrazil Status Code 200 OK a vytvořil záznam.

STATUS: **FAIL**

TEST_EXEC #013
Název: “Test metody DELETE se zadáním údajů a jejich následným vymazáním”
Prostředí: Programy Postman, MySQL Workbench

Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na POST.
3) V možnostech níže zvolit “Body” a následně možnost “raw” a “JSON”.
4) Zadat parametry ve formátu: { “first_name: “jmeno”, “last_name”: “prijmeni”, “email”: “emailova_adresa”, “age”: ***int*** }, např. { “first_name: “Vincent”, “last_name”: “Vega”, “email”: “zeds@dead.com”, “age”: ***50*** }
5) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
6) Odeslat požadavek pomocí tlačítka Send.
7) Zjistit ID vygenerované databází po vložení údajů.
8) Otevřít program MySQL Workbench.
9) Přihlásit se k databázi pomocí přihlašovacích údajů dostupných v zadání projektu.
10) Zadat SQL dotaz *“SELECT * FROM student WHERE id=<vygenerované ID>;”*
11) Vrátit se zpět do programu Postman.
12) Nastavit metodu na DELETE.
13) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/” , za poslední lomítko doplnit vygenerované ID
14) Odeslat požadavek pomocí tlačítka Send.
15) V programu MySQL Workbench zadat dotaz *“SELECT * FROM student WHERE id=<vygenerované ID>;”*

Očekávaný výsledek:
Po vložení údajů pomocí metody POST vrátí Postman Status Code 200 OK a zobrazí údaje společně s novým vygenerovaným ID. Pomocí tohoto ID si v MySQL ověřím, že záznam je skutečně v databázi. Následně pomocí metody DELETE záznam v Postmanovi vymažu. Postman zobrazí Status Code 200 OK a prázdný záznam. MySQL si ověřím, že záznam byl opravdu smazán a program vrátil tabulku s null.

Skutečný výsledek:
Po vložení údajů Postman zobrazil Status Code 200 OK a vygeneroval nové ID 1967. Pomocí tohoto ID jsem ověřil existenci záznamu v databázi přes MySQL. Záznam jsem přes Postman a metodu DELETE vymazal. Postman zobrazil prázdné datové pole a Status Code 200 OK. Po zopakování dotazu v MySQL s ID 1967 program vrátil prázdnou (null) tabulku.

STATUS: **PASS**

TEST_EXEC #014
Název: “Test metody DELETE s neexistujícím ID”
Prostředí: Program Postman
Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na DELETE.
3) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/99999”
4) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman vrátí Status Code 404 Not Found a chybovou hlášku, např. ID not found.

Skutečný výsledek:
Postman vrátil Status Code 500 Internal Server Error bez chybové hlášky.

STATUS: **FAIL**

TEST_EXEC #015
Název: “Test metody DELETE bez zadání ID”
Prostředí: Program Postman
Kroky:

1) Otevřít program Postman.
2) Nastavit metodu na DELETE.
3) Zaslat požadavek na REST API - do kolonky URL zadat “http://108.143.193.45:8080/api/v1/students/”
4) Odeslat požadavek pomocí tlačítka Send.

Očekávaný výsledek:
Postman vrátí Status Code 405 Not Allowed a chybovou hlášku, např. You are not permitted to execute this action.

Skutečný výsledek:
Postman vrátil Status Code 405 Not Allowed bez chybové hlášky.

STATUS: **FAIL**

BUG REPORT

Na základě provedených scénářů jsem objevil uvedené chyby aplikace.

bug	probability	severity	test case number	mitigace
wrong status code and missing error message	medium	moderate	#003	add an error message, change status code
missing error message	medium	minor	#004	add an error message
odd returned data in Postman, probably a missing error message	low	minor	#005	?
wrong status code and missing error message	medium	moderate	#007	add an error message, change status code
missing error message	medium	minor	#008	add an error message
wrong status code and missing error message	medium	moderate	#009	add an error message, change status code
ignores extra parameters not supported by database, creates new record anyway	low	major	#010	add a fix that catches entries with parameters unsupported by database
lets user enter negative number for parameter “age”	medium	critical	#011	set a realistic range for “age” parameter and block user from entering numbers that are negative or too large
accepts unrealistically long	medium	critical	#012	set a limit for input length and if exceeded,

inputs				display an error message and do not enter the data into the database
wrong status code and missing error message	medium	moderate	#014	add an error message, change status code
missing error message	medium	minor	#015	add an error message