

Практична робота 7

API тестування. Postman

Мета: Навчитися тестувати REST API за допомогою інструменту Postman: створювати запити, аналізувати відповіді, перевіряти статус-коди та писати автотести.

Вимоги до виконання звіту:

- Виконати звіт у текстовому форматі, прикріпивши скріншоти запитів до кожного з завдань
- Скріншоти мають містити тіло запиту, код відповіді та тіло відповіді
- Якщо виконуєте додатково тести до запитів, то додавати лістинг коду та результат виконання тестів до звіту

P.S: За написання тестів до усіх завдань, буде **додатково** нараховано 3 бали, і ще 3 бали буде за виконання 8 та 9 додаткових завдань

Завдання 1: Тестування GET-запитів

Опис:

Підключіться до публічного API, наприклад: <https://jsonplaceholder.typicode.com/posts>.

Інструкції:

1. Виконайте GET-запит до /posts.
2. Перевірте статус-код відповіді (очікується 200 OK).

Додатково:

3. Створіть тест у Postman для перевірки:
 - що відповідь містить щонайменше 10 об'єктів; ○
 - що кожен об'єкт має ключі id, title, body.

Завдання 2: Тестування POST-запиту (створення ресурсу)

Опис:

Протестуйте створення нового запису через POST-запит.

Інструкції:

1. Використайте ендпоінт <https://jsonplaceholder.typicode.com/posts>.
2. Надішліть POST-запит з тілом:

```
{  
  "title": "Test Post",  
  "body": "This is a test",  
}
```

```
"userId": 1  
}
```

3. Перевірте, що відповідь має статус-код 201 Created.

Додатково:

4. Напишіть тести у Postman для перевірки: ○ що відповідь містить створений об'єкт з вашим title; ○ що поле id присвоюється автоматично.

Завдання 3: Тестування PUT/PATCH-запиту (оновлення ресурсу)

Опис:

Виконайте оновлення існуючого посту.

Інструкції:

1. Надішліть PUT або PATCH-запит на /posts/1.
2. Змініть заголовок посту (title) на "Updated Title".
3. Перевірте, що відповідь містить оновлене значення title.

Додатково:

4. Напишіть тести для перевірки: ◦ відповідність статусу (200 OK);
5. оновлення конкретного поля.

Завдання 4: Негативне тестування (обробка помилок)

Опис:

Перевірте реакцію API на некоректні запити.

Інструкції:

1. Виконайте GET-запит до неіснуючого ресурсу, наприклад: /posts/9999.
2. Виконайте POST-запит без тіла або з некоректним JSON.
3. Перевірте статус-коди (наприклад, 404, 400).

Додатково:

4. Напишіть тести, які перевіряють наявність повідомлення про помилку.

Завдання 5: Робота з параметрами запитів

Опис:

Навчіться працювати з query-параметрами та перевіряти фільтрацію даних.

Інструкції:

1. Надішліть GET-запит до <https://jsonplaceholder.typicode.com/comments?postId=1>.
2. Перевірте, що всі коментарі у відповіді стосуються лише посту з postId = 1.

Додатково:

3. Напишіть тест, який перевіряє, що кожен об'єкт у відповіді має postId = 1.

Завдання 6: Тестування колекції з використанням змінних

Опис:

Організуйте API-запити в колекцію з використанням змінних середовища.

Інструкції:

1. Створіть колекцію з 3+ запитів (GET, POST, DELETE).

2. Використовуйте змінну середовища `{{base_url}}` замість прямого URL.
3. Запустіть колекцію через **Collection Runner**.

Завдання 7: Тестування DELETE-запиту

Опис:

Протестуйте видалення об'єкта через API.

Інструкції:

1. Надішліть DELETE-запит до <https://jsonplaceholder.typicode.com/posts/1>.
2. Перевірте, що API повертає статус-код 200 OK або 204 No Content.

Додатково:

3. Напишіть тести, які перевіряють відсутність тіла відповіді при запиті видаленого об'єкта.

Додаткові завдання (окремі бали за виконання):

Завдання 8: Тестування авторизації (Bearer Token)

Опис:

Навчіться додавати токени авторизації у запити.

Інструкції:

1. Виберіть публічне API з авторизацією (або змодельуйте своє).
2. Додайте токен до запиту через вкладку Authorization → Bearer Token.
3. Надішліть запит, переконайтесь у правильній авторизації (200 OK).
4. Напишіть тест, який перевіряє, що токен присутній у заголовках і відповідь не є помилкою.

Завдання 9: Імпортування API за специфікацією OpenAPI (Swagger)

Опис:

Навчіться імпортувати API з OpenAPI (Swagger) документації у Postman та працювати з ним.

Інструкції:

1. Знайдіть або використайте тестову OpenAPI-специфікацію (наприклад: <https://petstore.swagger.io/v2/swagger.json>).
2. У Postman:
 - Перейдіть у **File** → **Import** → **Link**.
 - Вставте посилання на Swagger-специфікацію.
3. Переконайтесь, що створилась колекція запитів.
4. Запустіть 2–3 запити з колекції та протестуйте їх (GET, POST).
5. Напишіть хоча б 1 автотест у вкладці **Tests** для кожного запиту.
6. Змініть значення одного параметра (наприклад, petId) через змінну.

Типи запитів

GET – використовується для отримання з боку серверу певного ресурсу. Якщо ви здійснюєте цей запит, сервер шукає інформацію та відправляє її вам назад.

POST – необхідний для створення певного ресурсу на сервері. Сервер створює в базі даних нову сутність та сповіщує вас, чи був процес створення успішним.

PUT та PATCH – використовуються для оновлення певної інформації на сервері. У такому разі сервер просто змінює інформацію існуючих сутностей у базі даних та повідомляє про успіх виконання операції

DELETE – видаляє вказану сутність із бази чи сигналізує про помилку, якщо такої сутності в базі не було.

Додатки

[hTps://www.postman.com/downloads/](https://www.postman.com/downloads/) - Завантажити Postman

[hTps://learn.ztu.edu.ua/mod/resource/view.php?id=174744](https://learn.ztu.edu.ua/mod/resource/view.php?id=174744) – матеріали до Postman

[hTps://petstoresampleapi.apimahc.dev/v/1.0.5#/hTp/how-to-get-started](https://petstoresampleapi.apimahc.dev/v/1.0.5#/hTp/how-to-get-started) – відкрите АПІ для тестування