API Testing with Postman

Class 21

19/7/2025

Acknowledgement

The series of the IT & Japanese language course is Supported by AOTS and OEC.

Ministry of Economy, Trade and Industry

Overseas Employment Corporation

What you have Learnt Last Week

We were focused on following points.

- Usage of control and loop flow statement
- Performing Linear Algebra in Numpy
- Why Requirement Analysis is so important in the process?
- Software development Life cycle
- Importance of Security compliance
- Introduction of Bash Scripting
- Introduction of docker and docker compose
- Introduction of Ansible

What you will Learn Today

We will focus on following points.

1. Setting Up Your API Testing Environment
2. Install Postman
3. Step-by-step instructions on building basic API requests
4. Understanding methods (GET, POST, PUT, DELETE), and setting parameters
4. Quiz
5. Q&A Session

Introduction to API Testing

Understand the Core of API Testing

What is an API?

An API (Application Programming Interface) allows different software systems to communicate with each other.
Example: When you log into Facebook, the app uses an API to communicate with the server.

Why is API Testing Important?

Ensures reliable communication between services
Identifies bugs before UI testing
Faster than end-to-end testing

Popular API Testing Tools:

Postman
Insomnia
curl (Command Line)
Swagger (OpenAPI interface)

What Happens in API Testing?

Real-World API Testing Goals

- Validate that the API returns the **correct response**
- Ensure **security** (no unauthorized access)
- Test for **performance** under load
- Handle **edge cases** and **error responses**
- Example: If a user ID doesn't exist, should return 404

Setting Up the API Testing Environment

Get Ready to Test

- Install **Postman** from https://www.postman.com
- Optional: Use **Insomnia** as a lightweight alternative
- Use a **sample API** for learning:
- JSONPlaceholder
- Reqres

Creating Your First Workspace in Postman

Organize Your API Testing

- Open Postman > Click “Workspace” > Create New
- Inside Workspace:
- Create **Collections** to group related APIs
- Save each **GET**, **POST**, **PUT**, or **DELETE** request
- Example:
- Collection: “User Management API”
- Requests: “GET All Users”, “Create User”, “Delete User”

Sample Public API Testing Example

Quick Hands-on Test

- Open Postman
- Method: GET
- URL: https://jsonplaceholder.typicode.com/posts/1
- Hit “Send”
- Check JSON response with userId, id, title, and body

Building Basic API Requests

Learn How to Create API Requests

Request Components:

- **URL:** Endpoint you're interacting with
- **Method:** GET, POST, etc.
- **Headers:** Metadata (e.g., Content-Type: application/json)
- **Body:** Data sent (for POST/PUT)
- **Params:** Key-value pairs for filtering or pagination

Example in Postman:

- URL: https://reqres.in/api/users
- Method: GET
- View: **Response status (200)**, **JSON body**, and **response headers**

HTTP Methods Overview

CRUD with REST APIs

- **GET:** Retrieve data
- **POST:** Create new data
- **PUT:** Update entire resource
- **DELETE:** Remove data

Each method matches a CRUD operation (Create, Read, Update, Delete)

GET Method – Read Data

Fetch Information

Used to fetch one or multiple records

Example:

- URL: https://jsonplaceholder.typicode.com/posts/1
- Result: JSON of one post

No body is sent with GET

Query params can be used: ?userId=1

POST Method – Create Data

Send New Information

Sends data in JSON format
Used to **create** a new user or item

Example:
•URL: https://reqres.in/api/users

Body:

```
{
  "name": “Umair",
  "job": "DevOps"
}
```

Response: 201 Created

1.Go to the **Headers** tab.
2.Add:

1.Add:
- **Key:** x-api-key
- **Value:** reqres-free-v1

PUT Method – Update Data

Replace Existing Resource

Replaces the whole existing record

Example:
•URL: https://reqres.in/api/users/2

Body:

```
{
  "name": "Ali",
  "job": "Engineer"
}
```

Use when you know the ID and want to update fully

DELETE Method – Remove Data

Delete Resource by ID

Used to delete a record

No body required

Example:

URL: https://reqres.in/api/users/2

Method: DELETE

Response: 204 No Content (success but no data returned)

Setting Parameters in API Requests

Control the Request Behavior

- **Query Parameters:** Appended to URL
  - **Example:** GET /posts?page=2

- **Path Parameters:** Variable parts of endpoint
  - **Example:** GET /users/{id} → /users/101

- **Headers:** Provide metadata
  - **Example:** Authorization: Bearer <token>

- **Body Parameters:** Sent in POST/PUT
  - Format: JSON
  - **Example:** { "name": "Ali", "email": "ali@example.com" }

Understanding API Responses

Analyze the Server's Reply

- **Status Codes:**
- 200 OK: Success
- 201 Created: Resource created
- 400 Bad Request: Invalid input
- 404 Not Found: Resource doesn’t exist
- 500 Internal Server Error: Server crashed

- **Response Body (JSON):**
- Example: { "id": 1, "name": "Ali" }

- **Response Headers:**
- Content-Type, Content-Length

- **Response Time:**
- Measure API performance (shown in Postman)

Introduction to API Authorization

Secure Your API Access

Authorization ensures only allowed users access certain endpoints.

Types of Authorization:

- API Key
- Bearer Token
- Basic Auth

Without proper auth, most APIs return **401 Unauthorized**.

Using API Keys

Simple Key-Based Access

API key is a unique identifier passed as a:

- **Query parameter:** ?apikey=XYZ123
- **Header:** x-api-key: XYZ123

Often used in public APIs.

Example with Postman:

- Go to Headers → Add key x-api-key with your key

Using Bearer Tokens

Secure Token-Based Auth

Common in OAuth-based APIs

Send token in header:

Authorization: Bearer eyJhbGciOi...

Tokens usually expire – re-auth required.

Example: Used in Google APIs, GitHub APIs, etc.

Basic Authentication

Username/Password Method

Format: Authorization: Basic base64(username:password)

Postman:

- Go to Authorization tab → Select **Basic Auth**
- Enter username and password

Not recommended unless over HTTPS.

Saving Requests in Postman Collections

Organize and Reuse Your Tests

- Collections = Folders for related API requests
- Click **Save** on a request → Choose/create a collection
- Share, document, and collaborate with team
- Allows **running collection as tests**

Environments in Postman

Reuse Variables like URLs, Tokens

Define environment variables (e.g., {{base_url}}, {{token}})
Create **Environment** → Add variables

Use in requests like:

{{base_url}}/users
Authorization: Bearer {{token}}

Easily switch between staging, dev, and prod environments

Automating with Test Scripts

Add JavaScript Logic to Requests

Run JS scripts after a response

Examples:

```
pm.test("Status is 200", function () {
  pm.response.to.have.status(200);
});
```

Use for status checks, value matching, saving values to variables

Automating Collections with Collection Runner

Batch Run Multiple Requests

Go to Collection → Click **Run**

Select environment, data file (optional), iterations

Useful for:

- Regression tests
- Load testing
- CI/CD pipeline integration

Assignment

Assignment

1. Create your own four APIs GET, POST, PUT and DELETE
2. Test these APIs on the postman
3. Upload documentation on GitHub

Quiz Section

Quiz

Everyone student should click on submit button before time ends otherwise MCQs will not be submitted

[Guidelines of MCQs]

1. There are 20 MCQs
2. Time duration will be 10 minutes
3. This link will be share on 12:25pm (Pakistan time)
4. MCQs will start from 12:30pm (Pakistan time)
5. This is exact time and this will not change
6. Everyone student should click on submit button otherwise MCQs will not be submitted after time will finish
7. Every student should submit Github profile and LinkedIn post link for every class. It include in your performance

Assignment

Assignment should be submit before the next class

[Assignments Requirements]

1. Create a post of today's lecture and post on LinkedIn.
2. Make sure to tag @Plus W @Pak-Japan Centre and instructors LinkedIn profile
3. Upload your code of assignment and lecture on GitHub and share your GitHub profile in respective your region group WhatsApp group
4. If you have any query regarding assignment, please share on your region WhatsApp group.
5. Students who already done assignment, please support other students

Q&A Session

ありがとうございます。
Thank you.
شکریا

For the World with Diverse Individualities