DAY 4

- what is API

An **API (Application Programming Interface)** is a set of rules and protocols that allow different software applications to communicate with each other. Think of it as a **messenger** that takes requests from one system, tells another system what you want, and then returns the response back to you.

- **HTTP Methods:**
  - **GET**: Retrieve data.
  - **POST**: Send data to create something new.
  - **PUT/PATCH**: Update existing data.
  - **DELETE**: Remove data.

1. Authentication:

Authentication is the process of **verifying the identity** of a user or system. It ensures that the person or application trying to access a resource is **who they claim to be**.

2. Authorization:

Authorization determines **what actions** or **resources** an authenticated user is allowed to access. It defines **granting permissions** and **access rights**.

- **HTTP Status Codes** are **standardized codes** returned by **web servers** in response to client requests. They indicate whether the request was successful, encountered an error, or requires further action.

Status Code	Message	Meaning
200	OK	The request was successful.
201	Created	The request was successful, and a new resource was created.
204	No Content	The request was successful, but there is no data to return.
301	Moved Permanently	The requested resource has been permanently moved.
302	Found	The resource is temporarily located elsewhere.
400	Bad Request	The server could not understand the request due to bad syntax.
401	Unauthorized	Authentication is needed to access the resource.
403	Forbidden	You don’t have permission to access the resource.

404	Not Found	The requested resource could not be found.
500	Internal Server Error	A generic error occurred on the server.
503	Service Unavailable	The server is not ready to handle the request

`Pytest` is a popular testing framework for Python that simplifies the process of writing and running tests. It is widely used for **unit testing**, **functional testing**, and **integration testing** in Python projects.

Key Features of `pytest`:

1. **Simple Syntax:**
   You can write tests using simple `assert` statements without needing to import special libraries or frameworks.

2.Automatic Test Discovery:
`pytest` automatically finds test files and functions based on naming conventions:

- Files should start with `test_` or end with `_test.py` (e.g., `test_api.py`).
- Functions should start with `test_` (e.g., `test_fetch_data()`).

3.Detailed Failure Reports:
When a test fails, `pytest` provides a detailed report, including the exact line where the failure occurred and the values involved.

4.**Support for Fixtures:**
`pytest` allows you to create reusable test setups using **fixtures**. Fixtures help manage tasks like setting up databases or mocking APIs before tests run.

5.**Plugins and Extensibility:**
`pytest` has a rich ecosystem of plugins for additional functionality like parallel test execution, HTML reporting, and more.

6. Parameterized Testing:
You can run the same test with multiple inputs using parameterization.

Why Use `pytest`?

- **Easy to Learn:** Minimal boilerplate code is required.
- **Flexible:** Suitable for small projects and large, complex applications.
- **Readable Output:** Provides clear feedback on test results.
- **Cross-Platform:** Works on Windows, Mac, and Linux.