APIs and REST APIs in Python

■ What is an API?

An API (Application Programming Interface) is like a messenger between two software systems. It allows one application to talk to another, share data, and request services. You can think of it as a waiter in a restaurant: you (the client) place an order, the waiter (API) takes it to the kitchen (server), and then brings your food (data) back.
■ Example: When you log in to a website using Google, the site uses Google's API to fetch your details securely.

■ Where are APIs used?

APIs are used almost everywhere in technology today:
- Web Applications – Frontend gets data from backend using APIs.
- Mobile Apps – Weather apps fetch real-time data from weather APIs.
- Social Media – Instagram APIs let apps show your posts, likes, and comments.
- Payments – Apps like PayPal, Razorpay use APIs for secure transactions.
- Machine Learning – APIs help deploy ML models so apps can use predictions.

■ Why use APIs?

APIs are important because they make development easier and more powerful:
- Reusability – Developers don't need to build everything from scratch.
- Communication – Different apps and systems can talk to each other.
- Security – APIs only share the data you allow, keeping systems safe.
- Scalability – Large systems can connect different services smoothly.

■ Types of APIs

There are different categories of APIs depending on their usage:
- Open APIs (Public) – Anyone can use them (e.g., Google Maps API).
- Internal APIs (Private) – Only used inside a company or organization.
- Partner APIs – Shared only with trusted business partners.
- Composite APIs – Combine results from multiple APIs into one.

By Protocols:
- REST API – Most common, uses HTTP requests.
- SOAP API – Uses XML, older and heavier.
- GraphQL – Flexible and allows custom queries.
- gRPC – Very fast, used for large-scale systems.

■ REST APIs (Representational State Transfer)

REST is the most popular way of designing APIs. It works over the web using HTTP methods. Each method has a specific purpose:
- GET → Used to retrieve data (like reading).
- POST → Used to send new data (like creating).
- PUT → Used to update existing data.
- DELETE → Used to remove data.

■ Example: Typing this URL in your browser – https://api.github.com/users/octocat – will return data about the GitHub user 'octocat'.

■ REST API in Python

Python makes it very easy to work with REST APIs using the 'requests' library.
Example 1 (GET Request – fetch user info from GitHub):

```
import requests url = 'https://api.github.com/users/octocat' response =
requests.get(url) print(response.json())
```

Example 2 (POST Request – send new data):

```
import requests url = 'https://jsonplaceholder.typicode.com/posts' data = {'title':
'My First Post', 'body': 'Hello API world!', 'userId': 1} response =
requests.post(url, json=data) print(response.json())
```

■ Summary

To recap what we learned about APIs and REST APIs:

- API = A bridge that allows apps to communicate.
- They are used in web apps, mobile apps, payments, and ML.
- APIs save time, improve security, and connect systems.
- REST API = Most common, uses HTTP methods like GET, POST, PUT, DELETE.
- Python's 'requests' library is widely used to interact with APIs.