**Assignment: Introduction to APIs and REST APIs with Python**

**Objective**

This assignment will introduce you to APIs (Application Programming Interfaces) and REST APIs. You will learn how to interact with APIs using Python’s requests module to fetch, send, and manipulate data.

By the end of this assignment, you should be able to:

* Understand what APIs and REST APIs are.
* Use the requests library to make HTTP requests (GET, POST, PUT, DELETE).
* Parse JSON responses from APIs.
* Work with real-world APIs (e.g., OpenWeatherMap, JSONPlaceholder, or GitHub API).

**Part 1: Understanding APIs and REST APIs**

**Task 1: Research & Short Answers**

1. **What is an API?**
   * Provide a definition and give two real-world examples.

**API (Application Programming Interface)** is a set of rules that allows different software programs to talk to each other and share data or functions.

**Example:**

* Google Maps API shows maps in apps.
* Twitter API lets apps read or post tweets.

1. **What is a REST API?**
   * Explain the key principles (HTTP methods, statelessness, resources).

A **REST API** (Representational State Transfer API) is a type of web API that follows specific rules to enable communication between systems over the internet, typically using **HTTP**.

**Key Principles of REST API:**

1. **HTTP Methods**  
   REST APIs use standard HTTP methods to perform actions:
   * **GET** – Retrieve data (e.g., get a user's profile)
   * **POST** – Create new data (e.g., add a new blog post)
   * **PUT** – Update existing data (e.g., edit a user’s details)
   * **DELETE** – Remove data (e.g., delete a comment)

Example:

* + GET /users/123 – fetch details of user with ID 123
  + DELETE /products/456 – delete product with ID 456

1. **List the common HTTP methods used in REST APIs and their purposes.**
   * (GET, POST, PUT, DELETE, PATCH)

**GET**

* **Purpose:** Retrieve data from the server.
* **Example:** GET /users/1 → Get details of user with ID 1.

**POST**

* **Purpose:** Create a new resource on the server.
* **Example:** POST /users → Create a new user.

**PUT**

* **Purpose:** Update an entire existing resource.
* **Example:** PUT /users/1 → Replace all details of user with ID 1.

**DELETE**

* **Purpose:** Remove a resource from the server.
* **Example:** DELETE /users/1 → Delete user with ID 1.

**PATCH**

* **Purpose:** Update part of an existing resource (partial update).
* **Example:** PATCH /users/1 → Update just the email or name of user 1.

1. **What is JSON? Why is it commonly used in APIs?**

**JSON** (JavaScript Object Notation) is a lightweight, easy-to-read data format used to share data between systems.

**Why it's used in APIs:**

* Simple and readable
* Works with most programming languages
* Fast to transfer
* Ideal for web and mobile apps

**Part 2: Making API Requests with Python**

**Task 3: GET Request (Fetching Data)**

Use the **JSONPlaceholder API** (a free fake API for testing):

* Endpoint: https://jsonplaceholder.typicode.com/posts

**Instructions:**

1. Write a Python script to fetch all posts from the API.
2. Print the response status code.
3. Print the first post in the response (JSON format).

**Expected Output:**

Status Code: 200

First Post: {'userId': 1, 'id': 1, 'title': '...', 'body': '...'}

**Task 4: POST Request (Sending Data)**

**Instructions:**

1. Use the same API to create a new post.
2. Send a JSON payload with:

{

"title": "New Post",

"body": "This is a test post.",

"userId": 1

}

1. Print the response (should include the new post with an ID).

**Expected Output:**

New Post: {'title': 'New Post', 'body': 'This is a test post.', 'userId': 1, 'id': 101}

**Task 5: Error Handling & Authentication**

**Instructions:**

1. Try accessing a non-existent endpoint (e.g., https://jsonplaceholder.typicode.com/nonexistent).
2. Handle the error (check status code, print an error message if request fails).

**Example Code:**

response = requests.get("https://jsonplaceholder.typicode.com/nonexistent")

**Part 4: Challenge Task (Real-World API Usage)**

**Task 6: Fetch Weather Data (OpenWeatherMap API)**

1. Sign up for a free API key at [OpenWeatherMap](https://openweathermap.org/api).
2. Fetch the current weather for a city of your choice.
3. Print temperature, weather description, and humidity.

**Endpoint:**

https://api.openweathermap.org/data/2.5/weather?q={city}&appid={API\_KEY}&units=metric

**Submission:** Submit your Python script.

**Submission Guidelines**

* Submit all Python scripts and the document for Part 1.
* Ensure code is well-commented.