**FSD Laboratory 06**

**Aim:** Develop a set of REST API using Express and Node.

**Objectives:**

1. To define HTTP GET and POST operations.
2. To understand and make use of ‘REST’, ‘a REST endpoint’, ‘API Integration’, and ‘API Invocation’
3. To understand the use of a REST Client to make POST and GET requests to an API.

**Theory:**

1. What is REST API?

**REST API** (Representational State Transfer Application Programming Interface) is a web service architectural style that uses HTTP requests to perform CRUD (Create, Read, Update, Delete) operations on resources. It follows a set of principles and constraints to ensure stateless interactions and a clear separation between client and server. REST APIs are commonly used to enable communication between a client (such as a web application or mobile app) and a server that hosts data or services.

Key characteristics of REST APIs include:

* **Statelessness**: Each request from the client to the server must contain all the information needed to understand and process the request. The server does not store client context between requests.
* **Resource-Based**: Resources are identified by URLs (Uniform Resource Locators), and actions on those resources are performed using standard HTTP methods.
* **Use of Standard HTTP Methods**: REST APIs utilize standard HTTP methods for operations (GET, POST, PUT, DELETE).
* **Representation**: Resources can be represented in various formats, such as JSON, XML, or HTML, allowing clients to consume them in a way that suits their needs.

1. Main purpose of REST API.

The main purposes of a REST API include:

1. **Interoperability**: REST APIs allow different systems, platforms, and programming languages to communicate with each other. This is crucial for integrating various services and applications.
2. **Decoupling Client and Server**: By using a stateless architecture and standard HTTP methods, REST APIs enable developers to change the client or server independently without affecting the other. This promotes flexibility and scalability in application development.
3. **Simplicity**: REST APIs are built on standard HTTP protocols, making them easier to use and understand. This simplicity contributes to faster development and deployment.
4. **Scalability**: REST APIs can handle a large number of requests, making them suitable for applications with varying loads. Their stateless nature allows for efficient resource management and scaling.
5. **Standardization**: REST APIs provide a standardized way of interacting with web services, reducing complexity and improving maintainability.

**FAQ:**

1. What are HTTP Request types?

Ans: HTTP request types (also known as HTTP methods) define the action to be performed on a specific resource. The most common HTTP request types are:

1. **GET**:
   * **Purpose**: Retrieve data from the server.
   * **Characteristics**:
     + Safe and idempotent (does not change the state of the resource).
     + Can include query parameters in the URL.
     + Data is typically returned in the response body.
   * **Example**: Retrieving user information from /users/123.
2. **POST**:
   * **Purpose**: Send data to the server to create a new resource.
   * **Characteristics**:
     + Not idempotent (multiple requests may create multiple resources).
     + Data is sent in the request body.
   * **Example**: Creating a new user by sending data to /users.
3. **PUT**:
   * **Purpose**: Update an existing resource or create a new resource if it does not exist.
   * **Characteristics**:
     + Idempotent (subsequent requests with the same data do not change the outcome).
     + Data is sent in the request body.
   * **Example**: Updating user information by sending data to /users/123.
4. **DELETE**:
   * **Purpose**: Remove a specified resource from the server.
   * **Characteristics**:
     + Idempotent (deleting the same resource multiple times has the same effect).
   * **Example**: Deleting a user by sending a request to /users/123.
5. **PATCH**:
   * **Purpose**: Apply partial modifications to a resource.
   * **Characteristics**:
     + Not necessarily idempotent.
     + Data sent in the request body specifies the changes to be made.
   * **Example**: Updating just the email of a user by sending a request to /users/123.

**Sample Problem Statements:**

**Creating and adding new book records in the book database using REST API.**

Help Link:

https://stackabuse.com/building-a-rest-api-with-node-and-express/

**Output: Screenshots of the output to be attached.**

**A screenshot of a computer

Description automatically generated**

Get:

A screenshot of a computer

Description automatically generated

A white background with black and white clouds

Description automatically generated

Post:

A screenshot of a computer

Description automatically generated

**Github link:** [**https://github.com/Aayush01055/FSD-Assignments.git**](https://github.com/Aayush01055/FSD-Assignments.git)