**FSD Lab 6**

**DEV DESAI PF 29**

**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 (Representational State Transfer) is an architectural style for designing networked applications, with REST APIs serving as the communication mechanism between software systems. RESTful APIs use standard HTTP methods to manipulate resources identified by URIs. They follow a stateless model, where each request contains all the information needed for communication.

1. Main purpose of REST API.

Main Purpose of REST API:

The primary objectives of REST APIs are:

1. Interoperability: Enable seamless communication between diverse applications and platforms.

2. Scalability: Support the distribution of services across servers for horizontal scalability.

3. Simplicity and Uniformity: Follow a straightforward and uniform approach using standard HTTP methods.

4. Resource Manipulation: Focus on manipulating resources identified by URIs using standard CRUD operations.

5. Statelessness: Embrace a stateless communication model, simplifying implementation and enhancing reliability.

In essence, REST APIs facilitate standardized, scalable, and interoperable communication between different software systems over the web.

**FAQ:**

1. What are HTTP Request types?

HTTP (Hypertext Transfer Protocol) supports various request methods indicating the intended action on a resource:

1. GET: Retrieves data from a specified resource without modification.

2. POST: Submits data for processing, often creating a new resource on the server.

3. PUT: Updates a resource or creates a new one if it doesn't exist, replacing the entire resource.

4. PATCH: Applies partial modifications to a resource, updating it with provided changes.

5. DELETE: Requests the removal of a specified resource.

6. HEAD: Retrieves only the headers of a resource for status checks without downloading the content.

7. OPTIONS: Describes communication options for the target resource, allowing the client to determine allowed methods.

8. TRACE: Performs a message loop-back test for diagnostic purposes along the path to the target resource.

These HTTP request methods serve specific purposes, and the choice depends on the desired action and the nature of the operation.

SCREENSHOTS –

A white background with a black border

Description automatically generated with medium confidenceA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

CODES –

A screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screen shot of a computer program

Description automatically generatedA screen shot of a computer

Description automatically generatedA screen shot of a computer program

Description automatically generatedA screen shot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screen shot of a computer code

Description automatically generated

FILE STRUCTURE

A screenshot of a computer

Description automatically generated