Smit Vora

Learnings and Conclusion

Module – 6

Topics

27. **Building Web API –** Web API is a programming interface that provides communication between two networks or software applications. It is often used to access data and save back data to the database.

    27.1 Understanding HTTP Verbs - HTTP verbs tell the server what to do with the data identified by the URL. The HTTP method is supplied in the request line and specifies the operation that the client has requested. If you're going to follow the REST architecture and the HTTP protocol, you must choose from the verbs available in that protocol, the primary or most-commonly-used HTTP verbs are POST, GET, PUT, and DELETE.

    Using HTTP Verbs :

    1. **Get(Read) -** This method is used to retrieve a representation of a resource. A GET request is considered safe because as HTTP specifies, these requests are used only to read data and not change it.

       Tasks that you can do with GET requests are:

       - You can cache the requests
       - It can remain in the browser history
       - You can bookmark the requests
       - You can apply length restrictions with GET requests
       - It is used only to retrieve data

       Example - http://www.example.com/customers/12345

    2. **POST(Create) -** In web services, POST requests are used to send data to the API server to create or update a resource. The data sent to the server is stored in the request body of the HTTP request.The simplest example is a contact form on a website. When you fill out the inputs in a form and hit Send, that data is

put in the response body of the request and sent to the server. This may be JSON, XML, or query parameters (there's plenty of other formats, but these are the most common).

Some important points about POST requests:

- Data will be re-submitted
- It cannot be bookmarked
- It cannot be cached
- Parameters are not saved in browser history
- No restrictions. Binary data is also allowed
- Data is not displayed in the URL

Example - POST http://www.example.com/customers

3. **PUT(Update) -** PUT is used to create a resource, not in a general case but when the resource ID is chosen by the client instead of by the server, then only PUT is used. Simply PUT updates data in the repository, replacing any existing data with the supplied data.

   Some good points of PUT requests are:

   - It completes as possible
   - It's as seamless as possible
   - Easy to use via HTML
   - It should integrate well with servers that already support PUT

   Example - http://www.example.com/customers/12345

4. **DELETE(Delete) -** A DELETE request is as simple as its name implies; it just deletes, it is used to delete a resource identified by a URI and on successful deletion, it returns HTTP status 200 (OK) along with a response body.

   Example - DELETE http://www.example.com/customers/12345

27.2 Implementing HTTP Verbs – Implementing HTTP verbs is made easy by platforms like Postman where it helps us to check our API URLs whenever we run the controller program. We can check every API one after other to check their functionalities which are going to be GET,POST,PUT,DELETE in which if it contains an id passing then we have to pass an id.

Key Points in Implementation –

- Build and run the project.
- Then we run the API which is get by default and we can choose options of post, put, delete and also some others needed accordingly to run our different API paths.
- It shows result in the body and then parameters in Params if passed any.
- We have to change the option from GET to POST if we run POST API.

27.3 Understanding the JSON Structure (JavaScript Object Notation)

- JSON is a text-based data exchange format derived from JavaScript that is used in web services and other connected applications.
- JSON is used to send JavaScript object to a server and to also retrieve back the JSON in the form of JavaScript Object from the server.
- JSON is an open-standards document format for human-readable and machine-understandable serialization and deserialization of data.
- Simply, it is used for data-interchange.
- Applications are created using different programming languages and run in very different environments. JSON is suited to this scenario because it is an open standard, it is easy to read and write, and it is more compact than other representations.
- RESTful web services use JSON extensively as the format for the data inside requests and responses. The HTTP header used to indicate that the content of a request or a response is JSON data is :
- **Content-Type: application/json**

JSON Object - In the case of an object, there are properties and their values that are enclosed inside the opening and closing curly braces. These are in

the form of a key/value pair. These key value sets are separated using a colon, “:” and multiple sets are separated using a comma, “,”.

Example - {

```
  "name": "Smit Vora",
  "age": 22
}
```

JSON Array - Like JavaScript, JSON also supports storing the objects in a sequence, series or list; what-ever you would like to put it as. Arrays are not complex structures in JSON document.

Example - ["Value 1", "Value 2"]