ADVANCED WEB APPLICATION DEVELOPMENT
WEEK 9 ASSIGNMENT

1. **List out the CRUD operations using REST API for developing web applications**

The four main HTTP methods (GET, PUT, POST, and DELETE) can be mapped to CRUD operations as follows:

- GET retrieves the representation of the resource at a specified URI. GET should have no side effects on the server.
- PUT updates a resource at a specified URI. PUT can also be used to create a new resource at a specified URI, if the server allows clients to specify new URIs. For this tutorial, the API will not support creation through PUT.
- POST creates a new resource. The server assigns the URI for the new object and returns this URI as part of the response message.
- DELETE deletes a resource at a specified URI.

Note: The PUT method replaces the entire product entity. That is, the client is expected to send a complete representation of the updated product. If you want to support partial updates, the PATCH method is preferred. This tutorial does not implement PATCH.

Creating An REST API with Express and Node.js

- First, create a folder where you want to store your files and folders. Let's name test-api, you can name it whatever you want.
- Now go to that directory and run npm init -y in the command line. It will initialize an empty directory to use for node.js and also create a file called package.json. package.json file contains much information about your project, like the name of the app, description about your project, author name, dependencies, and devDependecies. You can change it whenever you want.
- Above in scripts, we create a command run and we are going to use it with npm. When you run npm start then it will be executed as node server.js, where server.js is our main file for the app, you can name it whatever you want. Let's create a server.js file. But first, we have to install express, for that run given command: npm install express –save It will install express and save it to package.json.
- One thing you have noticed that one another folder is created named node_modules. In the node_modules folder, all the libraries and packages will be saved. Now let's create the server.js file and type given code. We have imported express and then initialized an app by calling express function, and then using the listen to the method, we have created a server on port 5000.
- When you run npm run command, it will run the server on port 5000. When you open the https://localhost:5000, then it will show you nothing, because we didn't create any route yet. By route, I mean a path like https://github.com/rajeshberwal Before creating any route, first, understand some HTTP methods that we use in web development. GET: get request is used to retrieve the data from the web, e.g., when you opened our article, you make a get request. POST: post request is used to send data to the server, e.g., user information like passwords, usernames, or any personal information.
- PUT: used to update the data on the server
- DELETE: used to delete a piece of information from server So, let's create a get request to the homepage.

- Now, run npm run command again and open http://localhost:5000 in your browser. Then you will see Hello world! on the homepage. In the above code / indicates our main homepage route, and in a method, we have to pass two things first, one is the route, and the other one is a middleware function.
- A middleware function is used to create functionality for that route. Now let's create a user route. Whenever someone calls our user route, they will get the user's information. We are going to store the rest of the routes in the routes folder. You can save them wherever you want. But storing paths in a routes folder is a standard way. To create folder routes in the current directory and create another file called users.js.
- You can name it whatever you want. In the user's files, we are going to create some dummy users. And now, import this route to our main server.js file. Here in users.js file, we are importing router from express. And then, we created two routes, the first one is /users, and the second one is /users/{user_id}.
- When someone calls users route, then they will get all the user's data, and when they specify the user by using id user/1 they will get data related to that user. And then we have exported that router. And we have imported that route in our main server.js file. In the server.js file, we have used it using the use method.
- It accepts two arguments, the first one is a route on that we want to show data, and the second one is a module. When we run the command npm start then we can see all the user's information on http://localhost:5000/users route, and when we specify an id http://localhost:5000/users/1,
- it will return data only for that user, and if the id is not available, then it will show "User not found." On http://localhost:5000/users If id available then: http://localhost:5000/users/0 If id is not possible: You have created your first API using Node.js and express.
- You can use this knowledge to develop your applications. You can use other HTTP methods to increase its functionality