**Assignment No. 01**

**Submitted By:** Umar Farooq

**Submitted To:** Prof. Hamza Afzal

**Roll No:** G1F22UBSCS147

**Assignment Title:** To Do List App

**Subject:** Web Programming

**Introduction**

This assignment focuses on developing a backend-only To-Do List application using Node.js and Express.js. The main purpose is to practically implement RESTful API concepts such as creating routes for adding, retrieving, and deleting tasks. This assignment also introduces the use of Postman as an API testing tool. Through this practical, students will understand the process of setting up a Node.js environment, building a lightweight Express server, and testing endpoints effectively. It helps in strengthening backend development skills without using any database by employing a simple array to manage data.

**Steps Used In This Assignment**

**1. Install Node.js and npm**

Node.js was installed on the system to enable JavaScript execution outside the browser. The Node Package Manager (npm) was also installed alongside.

**2. Create Project Folder**

A dedicated folder named backend was created and opened in Visual Studio Code.

**3. Initialize npm**

The terminal command npm init -y was used to generate a default package.json file.

**4. Install Express.js**

Express.js, a minimal Node.js framework, was installed using the command npm install express.

**5. Create Server File**

A file named server.js was created. In it, an Express server was written to handle three main routes:

POST /addTask to add a new task, GET /tasks to view all tasks, and DELETE /task/:id to delete a task by its ID

**6. Run the Server**

The server was started using the command node server.js. It confirmed successful execution by displaying a local URL (localhost:3000).

**7. Install and Open Postman**

Postman was installed to test the backend API by sending HTTP requests.

**8. Test POST Route**

A POST request was sent to /addTask using Postman with JSON data in the body containing the task name. A new task was successfully added and returned in the response.

**9. Test GET Route**

A GET request was made to /tasks to retrieve and display all current tasks stored in the array.

**10. Test DELETE Route**

A DELETE request was sent to /task/:id with a valid task ID to remove a specific task. The response confirmed successful deletion.

**Summary**

This assignment taught how to create a RESTful API using Express.js, Node.js and npm were installed to enable backend JavaScript development. A new project folder and server file were set up in Visual Studio Code. Express.js was installed to simplify route handling. The server was written with three core routes: POST, GET, and DELETE. Tasks were stored in a local array, avoiding the need for a database.

Postman was used to test the API endpoints efficiently. The POST request allowed adding a task with taskName in JSON format. The GET request returned all current tasks in the array. The DELETE request removed a task based on its ID. Each API response was verified and captured with screenshots. The application ran locally on port 3000 for all interactions. This project built a strong foundation for backend API development.