PUMO TECHNOVATION
INDIAS ADVANCED TECH CAMPUS

Duration : 2hrs

NODE & EXPRESS ASSESSMENT

Marks : 50

Section 1:

Marks : 15

1. What are the different types of middleware in Express.js?
2. What is status code in Express.js? Explain any 5 of them.
3. What is NPM? How do you initialize a Node.js project?

Section 2:

Marks : 15

4. How do you create a simple Express.js server?
5. How do you handle routing in Express.js?
6. How do you handle errors in Express.js?

Section 3:

Marks : 20

7. TASK

Requirements:

Create a simple **Node.js Express API** for user authentication using **JWT** and **bcrypt**.

Register API (POST /register)

Accepts name, email, and password.

Hashes the password using bcrypt.

Stores users in an in-memory array (no database).

Responds with a success message.

Login API (POST /login)

Accepts email and password.

Validates user credentials.

Returns a signed JWT token if login is successful.

Use jsonwebtoken and .env for secret key.

Protected Route (GET /profile)

Accessible only with a valid JWT in the Authorization header.

Returns a welcome message with the user's email.

Environment Config (.env)

Store PORT and JWT_SECRET securely using the dotenv package.

8. Build a basic MERN Stack application where the **React frontend** communicates with the **Node.js + Express backend**, and data is stored in **MongoDB**.

Requirements

Backend (Node.js + Express + MongoDB):

Create an Express server.

Connect it to MongoDB using **Mongoose**.

Create a simple REST API to:

POST /api/users – Add a user with name and email

GET /api/users – Fetch all users

Use cors and express.json() middleware.

Store users in a MongoDB collection (users).

Frontend (React):

Create a React app using create-react-app or Vite.

Create a form to submit name and email.

On form submit:

Send POST request to backend (/api/users)

Show success message or error

Display a list of users below the form:

Use GET /api/users to fetch and render all users

Connectivity Requirements:

Use **Axios** or fetch() in React to connect with Express backend.

Ensure the backend runs on a different port (e.g., 5000), and React on another (e.g., 3000).

Handle **CORS** properly on the backend.