Additional Lab programs

21. Build a React application using React Router with the following pages:
• **Dashboard** – shows a list of tasks.
• **Task Details** – dynamic route showing task info using /task/:taskId.
• **Settings** – contains user preferences.
Add a navigation menu and show a "Page Not Found" screen for invalid routes

22. Design a registration form containing:

1. Username
2. Email
3. Mobile Number
4. Password
5. Confirm Password
6. Branch Selection (dropdown)
7. Upload Profile Picture
8. Register and Reset buttons

Write a JavaScript program that validates:
i. Username must be at least 6 characters.
ii. Email must follow correct format.
iii. Mobile number must contain exactly 10 digits.
iv. Password must include letters and numbers.
v. Confirm Password must match.
vi. Error messages should appear next to each field.
vii. After successful validation, display the user details in a formatted card using DOM elements.

23. Build a webpage that handles events: **onclick, onmouseover, onkeyup, onfocus.**

24. Develop an **Express.js** API for Teacher Records that performs CRUD operations:
i. Add routes to create, fetch all, fetch by ID, update, and delete teacher details.
ii. Ensure JSON data is parsed using middleware and invalid requests return error messages.
iii. Include fields: name, department, experience, and email.
iv. Return meaningful success messages for each action.

25.Create a Course Management API using Node.js and MongoDB:
• Add routes to create, view, update, and delete course details.
• Each course contains: courseName, faculty, credits, and studentsEnrolled.
• Implement a separate route /courses/totalStudents to calculate total enrollment across all courses.
• Use Mongoose validation for required fields.

26. Develop a Node.js program using the **MongoDB native driver** to accomplish the following:

1. Read a list of employee records from a file named **employees.json**.
2. Connect to a MongoDB database **CompanyDB** and insert the records into a collection called **employees**.

3. Update the record of the employee with **empId = 101** by changing the value of the **designation** field.
4. Retrieve all employee records and write them into a file named **updated_employees.json**.
5. Display a message confirming that the export process has been successfully completed.