ReactJs Programming

Section 1

- **React JSX and styling, Conditional Rendering**
- **React Events Handing**
- **React State**
- **React Props, TypeChecking with PropType**
- **Communication Between Parent and Child Component**
- **React Lifecycle(Mounting)**
- **React Forms(Controlled and UnControlled Component), React Bootstrap**
- **React Routing, React Higher Order Component**
- **HTTP Server Communication (Fetch API)**
- **React Hook**
- **React Redux**
- **Redux toolkit**
- **React LocalStorage**
- **React Axios and JSON Parsing, REST API**
- **Firebase Phone Authentication**
- **File Uploading**
- **JWT Token**
- **React-Bootstrap**

1. Create a simple React component that renders a button and applies custom styling using inline CSS. Add an onClick event handler to the button that displays an alert message when clicked.
2. Build a weather application that conditionally renders different components based on the weather condition (e.g., sunny, rainy, cloudy). Use conditional rendering to display different icons and messages based on the weather data.
3. Implement a counter component that increments or decrements a value using event handling. Add buttons for incrementing and decrementing the value, and display the current value on the screen.
4. Develop a todo list application with React state management to add, delete and update tasks. Use React forms to capture user input for adding new tasks and implement functionality to delete or mark tasks as Completed.
5. Create a parent component that passes data and functions as props to child components to manage a shopping cart. Implement functionality to add items to the cart, update quantities, and remove items.
6. Build a movie search application that communicates with an API server using the Fetch API and displays the results.

1. Build a blog application with React Router for navigating between different blog posts. Create multiple routes for displaying individual blog posts and a home page that lists all the blog posts.
2. Develop a contact form component with both controlled and uncontrolled inputs. Implements validation for required fields and display error messages. Use React Bootstrap for styling the form.
3. Implement a file upload feature that allows users to select and upload files. Use a library like React Dropzone to handle gile uploads and display progress indicators.
4. Create a login form component with JWT token authentication. Use Axios to make HTTP requests to an API for user authentication. Store the JWT token in local storage and include it in subsequent API requests for authorization.
5. Build a payment integration feature using Stripe or RazorPay API. Implement a payment form with validation and process the payment using he selected payment gateway.
6. Create a real-tie chat application using React and Firebase Firestore for storing and retrieving chat messages.