Name:K.Priya

Hall ticket number:2303a51194

Batch-04

Date:-28-01-2025

1. Setting up a new React project using Create React App or Vite

Using Create React App (CRA)

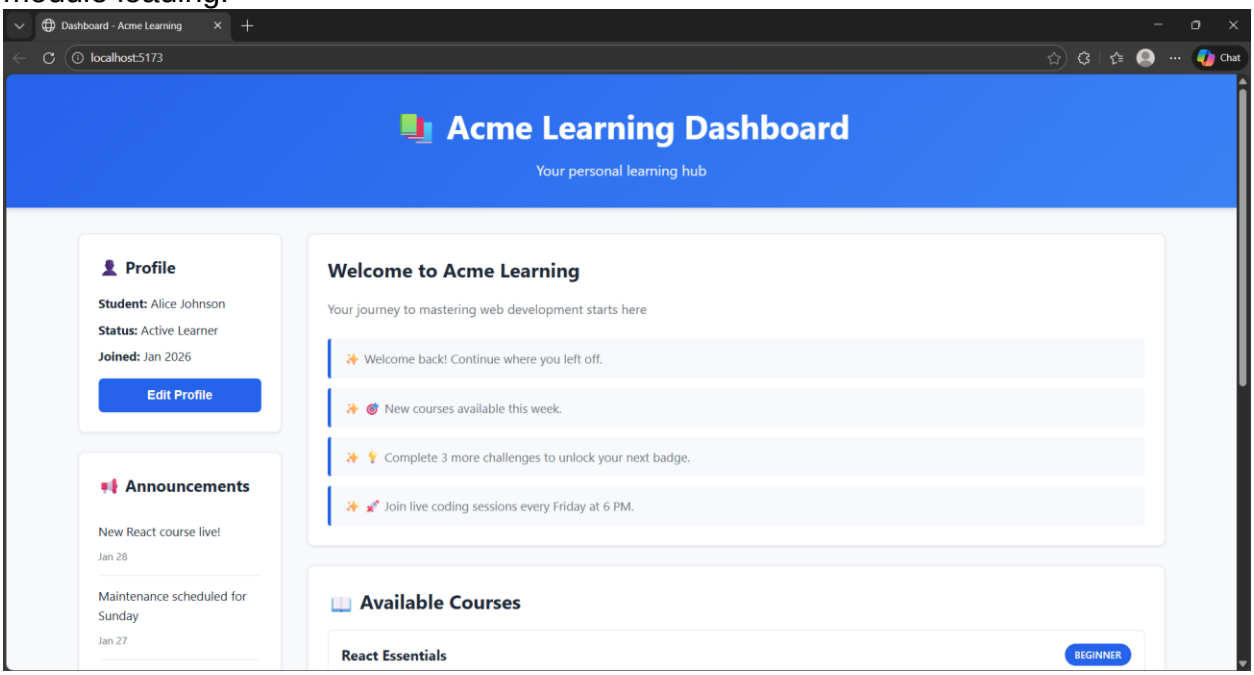
Create React App provides a preconfigured environment suitable for beginners and small projects.

Steps: 1. Ensure Node.js and npm are installed. 2. Run the following command: `bash npx create-react-app my-dashboard` 3. Navigate into the project folder: `bash cd my-dashboard` 4. Start the development server: `bash npm start` The app runs on `http://localhost:3000` with hot reloading enabled.

Using Vite

Vite is a modern build tool known for faster startup and optimized builds.

Steps: 1. Ensure Node.js is installed. 2. Run: `bash npm create vite@latest my-dashboard` 3. Select the React template. 4. Navigate to the project folder and install dependencies: `bash cd my-dashboard npm install` 5. Start the development server: `bash npm run dev` Vite serves the app with faster cold starts and efficient module loading.



---

2. Role of package.json in a React Project

The `package.json` file is the configuration and dependency management file for a React project.

It defines: - Project metadata (name, version, description) - Dependencies and devDependencies required by the application - Scripts for running common tasks such as development, build, test, and lint - Engine and tool configuration constraints

It allows consistent setup across different environments and simplifies dependency installation.

---

3. Creating a Functional Component in React

A functional component is a JavaScript function that returns JSX.

Example:

```
function Welcome() {
  return <h1>Welcome to the Dashboard</h1>;
}

export default Welcome;
```

Functional components are simple, reusable, and support hooks for managing state and lifecycle behavior.

---

4. Rendering Components Inside the Main App Component

Components are rendered by importing them into the `App` component and using them as JSX elements.

Example:

```
import Welcome from './Welcome';

function App() {
  return (
    <div>
      <Welcome />
    </div>
  );
}

export default App;
```

The `App` component acts as the root container for composing the UI.

---

5. Benefits of Breaking UI into Reusable Components

Breaking the UI into small reusable components provides several advantages:

- **Reusability:** Components can be reused across different parts of the application.
- **Maintainability:** Changes are isolated to specific components, reducing risk.
- **Readability:** Smaller components are easier to understand and debug.
- **Scalability:** Applications can grow without becoming tightly coupled or complex.
- **Testability:** Components can be tested independently.

This approach follows React's component-based architecture and improves overall development efficiency.