**MODULE - 4**

**Assignment 5: Setup the React Project**

**Objective**

Initialize a React project with TypeScript and set up a basic component structure, including

Header, Footer, and Dashboard components.

Sure, here’s a step-by-step approach to setting up a React project with TypeScript using Vite, followed by creating and implementing core components like Header, Footer, and Dashboard.

**Step-by-Step Approach**

**Step 1: Initialize the React Project**

**1. Create a New React Project with TypeScript Support:**

- Use Vite to create a new React project:

npm create vite@latest ielts-speaking-test --template react-ts

cd ielts-speaking-test

npm install

**2. Run the Development Server (Optional for Initial Verification):**

- Start the development server to ensure the setup works correctly:

npm run dev

**Step 2: Set Up Basic Folder Structure**

**1. Organize Project Directory:**

- Ensure your project directory has the following structure:

src/

├── components/

│ ├── Header.tsx

│ ├── Footer.tsx

│ ├── Dashboard.tsx

├── App.tsx

├── index.tsx

**Step 3: Implement Core Components**

**1. Header Component:**

**- Objective:** Displays the platform name and navigation links (e.g., "Home", "Dashboard").

**- Structure:**

- Create a functional component in `Header.tsx`.

- Include the platform name and navigation links.

**2. Footer Component:**

**- Objective:** Shows copyright and developer contact information.

**- Structure:**

- Create a functional component in `Footer.tsx`.

- Include copyright information and contact details.

**3. Dashboard Component:**

- **Objective:** Placeholder content for test overview or instructions.

**- Structure:**

- Create a functional component in `Dashboard.tsx`.

- Include placeholder content (e.g., "Welcome to the Dashboard!").

**Step 4: Integrate Components**

**1. Render Components in `App.tsx`:**

- Use the `App.tsx` file to render `Header`, `Footer`, and `Dashboard` components.

- Ensure proper layout and styling as needed

**Step 5: Run and Test the Application**

**1. Start the Development Server:**

- Start the server using:

npm run dev

- Ensure the components render correctly in the browser.

**2. Verify Components:**

**- Header Component:** Ensure it displays the platform name and navigation links.

- **Footer Component:** Ensure it shows copyright and contact information.

**- Dashboard Component:** Ensure it displays placeholder content.

**Detailed Steps**

**Step 1: Initialize the React Project**

**1. Open your terminal and run the commands to create and set up the project**

npm create vite@latest ielts-speaking-test --template react-ts

cd ielts-speaking-test

npm install

**2. Optionally, start the development server to verify the initial setup:**

npm run dev

Verify that the project runs correctly by opening the provided URL in the browser.

**Step 2: Set Up Basic Folder Structure**

**1. Modify the `src` directory to include the `components` folder and the required component files:**

- Header.tsx

- Footer.tsx

- Dashboard.tsx

- Ensure your directory looks like:

src/

├── components/

│ ├── Header.tsx

│ ├── Footer.tsx

│ ├── Dashboard.tsx

├── App.tsx

├── index.tsx

**Step 3: Implement Core Components**

**1. Header Component (`Header.tsx`):**

- Create a functional component.

- Include HTML elements for the platform name and navigation links (e.g., "Home", "Dashboard").

**2. Footer Component (`Footer.tsx`):**

- Create a functional component.

- Include HTML elements for copyright information and contact details.

**3. Dashboard Component (`Dashboard.tsx`):**

- Create a functional component.

- Include HTML elements for placeholder content (e.g., "Welcome to the Dashboard!").

**Step 4: Integrate Components**

**1. Modify `App.tsx` to Render Components:**

- Import `Header`, `Footer`, and `Dashboard` components.

- Render these components within the `App` component.

- Ensure the layout is appropriate and components are displayed correctly.

**Step 5: Run and Test the Application**

**1. Start the Development Server:**

- Run the server using:

npm run dev

- Check the application in the browser to ensure all components are rendered correctly.

**2. Verify the Output:**

- Ensure that:

- The `Header` component displays the platform name and navigation links.

- The `Footer` component displays the copyright and contact information.

- The `Dashboard` component displays the placeholder content.

**Submission Guidelines**

**1. Prepare the Project Folder for Submission:**

- Ensure all components and files (`Header.tsx`, `Footer.tsx`, `Dashboard.tsx`, `App.tsx`, `index.tsx`) are correctly implemented and included.

- Zip the complete project folder.

**2. Include Screenshots or Screen Recording:**

- Provide evidence (screenshots or screen recording) of the running application showing the correct display of components.

**3. Submit the Project:**

- Provide the zipped project folder or a version control repository link (e.g., GitHub).