STEP-BY-STEP GUIDE: CREATING A REACT APPLICATION USING JAVASCRIPT
(With Multiple Components)

Prerequisites
----
1. Basic knowledge of JavaScript and HTML.
2. A computer with internet access.

Step 1: Install Node.js and npm
----
1. Open your browser and go to https://nodejs.org
2. Download the LTS (Recommended) version for your operating system (Windows, macOS, or Linux).
3. Run the installer and follow the on-screen steps (Next → Next → Finish).
4. After installation, open a terminal or command prompt and type:
   - node -v
   - npm -v

   You should see version numbers for both commands. If you see them, Node.js and npm are installed correctly.

Step 2: Install a Code Editor (VS Code recommended)
----
1. Go to https://code.visualstudio.com
2. Download and install Visual Studio Code.
3. Open VS Code after installation.

Step 3: Create a New React Project Using Vite

We will use Vite (a fast build tool) to create a React app using JavaScript.

1. Open a terminal or command prompt.
2. Choose or create a folder where you want your project to live. For example:
   - mkdir react-projects
   - cd react-projects
3. Run the following command to create a new Vite project:
   - npm create vite@latest my-react-app
4. You will be asked a few questions in the terminal:
   - Project name: Press Enter to accept "my-react-app" or type a different name.
   - Select a framework: Use arrow keys and choose "React". Press Enter.
   - Select a variant: Choose "JavaScript". Press Enter.
5. After this, a new folder with your project name (for example, my-react-app) will be created.

Step 4: Install Project Dependencies
----
1. Move into the project folder:
   - cd my-react-app
2. Install all required dependencies by running:
   - npm install

This command reads the package.json file and installs React, ReactDOM, and other dependencies needed for your app.

Step 5: Open the Project in VS Code
----

1. With the terminal still inside your project folder (my-react-app), run:
   - code .
2. This will open your React project in Visual Studio Code.

Step 6: Understand the Project Structure

Important files and folders in a Vite + React + JavaScript project:

1. package.json
   - Contains project information and dependencies.
   - You run scripts from here like "npm run dev", "npm run build".

2. index.html
   - The main HTML file. It has a <div id="root"></div> where your React app will be injected.

3. vite.config.js
   - Vite configuration file.

4. src/main.jsx
   - Entry point for the React application.
   - It renders the root React component into the DOM.

   Example main.jsx:

```
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './App.jsx'
import './index.css'

ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
)
```

5. src/App.jsx
   - Main React component file where you build your UI.
6. src/index.css
   - Global CSS styles.

Step 7: Run the React Application
----

1. In the terminal (inside the project folder), run:
   - npm run dev
2. Vite will start a development server and show you a URL like:

```
- Local: http://localhost:5173/
```

3. Open your browser and go to the shown URL.
4. You should see the default Vite + React page.

Step 8: Edit the App.jsx Component

1. In VS Code, open src/App.jsx.
2. Replace its content with the following simple React component:

```
function App() {
  return (
    <div>
      <h1>Hello React with JavaScript!</h1>
      <p>This is my first React app created using Vite.</p>
    </div>
  );
}

export default App;
```

3. Save the file (Ctrl+S or Cmd+S).
4. Your browser will automatically refresh and display the updated content.

Step 9: Add Some Styling Using CSS

1. Open src/index.css.
2. Replace its content with simple styling:

```
body {
  margin: 0;
  font-family: Arial, Helvetica, sans-serif;
  background-color: #f5f5f5;
}

#root {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
}

h1 {
  color: #333;
}

p {
  color: #555;
}
```

3. Save the file and check the browser to see the updated UI.

Step 10: Create a New Component (Greeting)
----

1. Inside the src folder, create a new file called Greeting.jsx.
2. Add the following code:

```
function Greeting(props) {
  return <h2>Welcome, {props.name}!</h2>;
}

export default Greeting;
```

3. Now open src/App.jsx and update it as follows to use the Greeting component:

```
import Greeting from './Greeting.jsx';

function App() {
  return (
    <div>
      <h1>Hello React with JavaScript!</h1>
      <p>This is my first React app created using Vite.</p>
      <Greeting name="Praveen" />
    </div>
  );
}

export default App;
```

4. Save the files. Your browser should now display the greeting message below the heading.

Step 11: Add Another Component and Display Both

Now we will add one more component and display BOTH components inside App.jsx.

1. Inside the src folder, create a new file called Profile.jsx.
2. Add the following code:

```
function Profile() {
  return (
    <div>
      <h2>Trainer Profile</h2>
      <p>Praveen - Java Full Stack & DevOps Trainer</p>
    </div>
  );
}

export default Profile;
```

3. Now open src/App.jsx and update it to use both Greeting and Profile components:

```
import Greeting from './Greeting.jsx';
import Profile from './Profile.jsx';
```

```
function App() {
  return (
    <div>
      <h1>Hello React with JavaScript!</h1>
      <p>This is my first React app created using Vite.</p>
      <Greeting name="Praveen" />
      <Profile />
    </div>
  );
}

export default App;
```

4. Save all files.
5. Your browser will now show:
   - Main heading
   - Paragraph text
   - Greeting component ("Welcome, Praveen!")
   - Profile component ("Trainer Profile" and details)

This demonstrates how to:

- Create multiple components.
- Import them into App.jsx.
- Display both components together in the main layout.

Step 12: Build the App for Production

When your app is ready to be hosted on a server:

1. Stop the dev server (if running) using Ctrl+C in the terminal.
2. Run the build command:
   - npm run build
3. Vite will create an optimized production build inside the dist folder.
4. You can deploy the contents of the dist folder to any static hosting provider (like Netlify, Vercel, GitHub Pages, etc.).

Step 13: Summary

- You installed Node.js and npm.
- You created a new React app using Vite with JavaScript.
- You learned the main files and structure of the project.
- You ran the app in development mode.
- You edited App.jsx and created new components.
- You added basic CSS styling.
- You created multiple components (Greeting and Profile) and displayed both inside App.jsx.
- You built the app for production.

You now have a fully working React application using JavaScript with multiple components.