MERN STACK TASK 1: Basic React Application Development

□ Task Overview

This task involved creating a **basic ReactJS application** to develop a foundational understanding of React development. It included setting up the project, exploring the file structure, creating components, and understanding the development workflow.

□ Tools and Technologies Used

- **ReactJS**
- **Node.js and npm**
- **VS Code**
- **Command Line / Terminal**

□ 1. React Application Setup

➤ Command Used:

```
npx create-react-app my-react-app
cd my-react-app
```

➤ Initialization:

- Installed all required dependencies.
- Project structure auto-generated with core folders and files.

---

2. File and Folder Structure Explanation

File/Folder	Purpose
`src/`	Contains main code: React components, styles, logic.
`public/`	Holds static files (e.g., `index.html`) – entry point of the app.

File/Folder	Purpose
`node_modules/`	Contains all npm packages. Auto-generated, should not be modified.
`package.json`	Lists all dependencies, scripts, and metadata.
`package-lock.json`	Ensures exact versions of dependencies are installed.
`.gitignore`	Lists files/folders to exclude from Git (e.g., `node_modules/`).
`README.md`	Contains instructions and documentation about the project.

---

3. Basic React Component Created

➤ Component: `Greeting.js`

```
import React, { useState } from 'react';

function Greeting(props) {
  const [name, setName] = useState(props.name || "Guest");

  return (
    <div>
      <h1>Hello, {name}!</h1>
      <button onClick={() => setName("Veeresh")}>Change Name</button>
    </div>
  );
}

export default Greeting;
```

➤ Integration in `App.js`:

```
import React from 'react';
import Greeting from './Greeting';

function App() {
  return (
    <div>
      <Greeting name="React Learner" />
    </div>
  );
}

export default App;
```

---

4. Development Workflow Observed

➤ Running the Development Server:

```
npm start
```

➢ Observations:

- Application runs at `http://localhost:3000`.
- **Hot Reloading** updates the browser view instantly after code changes.
- Debugging and testing made easier with real-time feedback.

---

5. Observations and Learnings

React Core Concepts:

- **JSX**: Used to write HTML-like syntax in JavaScript.
- **Components**: Created reusable `Greeting` component.
- **Props**: Passed dynamic data (`name`) to components.
- **State (`useState`)**: Used to change and manage internal component data.

Challenges Faced:

- Forgetting to import the custom component in `App.js`.
- Initial confusion around file structure and where to place custom components.

Key Learnings:

- React follows a **component-based architecture**.
- Props and State help in making applications **interactive** and **dynamic**.
- Hot reload significantly improves development experience.
- Proper file structure understanding is essential for scaling larger projects.

Submitted by:

Name: Veeresh Hedderi
Task: MERN Stack Task 1 – Basic React Application
Submitted to: Main Flow Services and Technologies Pvt. Ltd.
Contact: veereshhedderi18gmail.com