# React Task 1: Observations and Learnings

## 1. File and Folder Structure

**Key Folders:**

**- src Folder:** This is the core of the application, where all components, styles, and utility files are located.

- **public Folder:** Contains static assets such as `index.html`, which acts as the entry point for rendering the React app.

- **node\_modules Folder:** Stores all the installed libraries and dependencies for the project to function properly.

**Key Files:**

- `**package.json`:** Keeps track of project dependencies, scripts, and other project metadata.

- `package-lock.json`: Ensures consistent dependency versions are installed across environments.

- **`.gitignore`:** Lists files and folders to exclude from version control.

- `**README.md`:** Includes setup instructions and a brief overview of the project.

## 2. Challenges Faced

**1. Installing Dependencies:**

- Issue: Faced an error during `npm install`.

- Resolution: Cleared the npm cache using `npm cache clean --force` and reinstalled the dependencies.

**2. Hot Reloading Issue:**

- Issue: Changes made to the code were not reflecting in the browser.

- Resolution: Restarted the development server using `npm start`, which resolved the issue.

**3. Component Integration:**

- Issue: Initially struggled with importing and integrating a custom component into the application.

- Resolution: Referred to the React documentation to understand component imports and successfully added it to `App.js`.

## 3. Insights Gained

**1. React Fundamentals:**

- Gained a better understanding of how to create and use React components.

- Learned about JSX syntax and how to manage state and props within a component.

**2. Development Workflow:**

- Understood how to run the development server using `npm start`.

- Observed how hot reloading updates the browser view immediately after code changes.

**3. Project Organization:**

- Recognized the importance of having a well-structured file system in React.

- Learned the role of `index.html` in rendering the application.

## 4. Workflow Summary

**1. Project Setup:**

- Used `npx create-react-app my-react-app` to initialize the project.

- Confirmed successful installation of dependencies in the `node\_modules` folder.

**2. Explored File Structure:**

- Analyzed the purpose of each folder and file in the project directory.

**3. Created a React Component:**

- Built a simple component (`MyComponent.js`) that utilized JSX, state, and props.

- Integrated the component into `App.js` to display it in the application.

**4. Ran and Tested the Application:**

- Started the development server using `npm start`.

- Accessed the application in the browser at `http://localhost:3000`.

- Tested the hot reloading feature by observing live updates as changes were made to the code.

## 5. Final Reflections

This task was a valuable introduction to ReactJS. It provided hands-on experience in setting up and working with a React application. Creating components and observing how they interact helped deepen my understanding of React’s component-based architecture. These skills will be beneficial for future projects and more advanced tasks in React development.