

Internship Report – Frontend Dev

Week 6: React.js Basics

Name: Zainab

Father Name: Assad Qayyum

Date: 28th July, 2025

Internship Domain: Front-end Intern

Task: Intro to React, create-react-app setup, JSX, Components (Functional), Props

Task Overview: (Day1)

In today's task, I was introduced to React.js and learned the basics of setting up a React project using create-react-app. I also explored JSX syntax, functional components, and how to pass data using props, gaining a foundational understanding of React's component-based architecture.

Content Covered:

The content covered today is:

- Introduction to React
- Create-react-app setup
- JSX (JavaScript XML)
- Components (Functional)
- Props (Properties)

1. Introduction to React:

React is a **JavaScript library** developed by Facebook for building **user interfaces (UI)**. It allows you to create reusable UI components, making development faster and easier.

React uses a component-based architecture, meaning your app is divided into small, independent, reusable pieces. It uses a virtual DOM, which efficiently updates the UI when data changes, improving performance.

Why Use React?

Reusable Components: Build once, use multiple times.

Fast: Virtual DOM only updates parts of the page that change.

Interactive: Makes websites dynamic (e.g., live search, chats).

Declarative: You describe what you want to see and React handles the DOM updates.

Key Concepts:

- **JSX:** A syntax that lets you write HTML-like code inside JavaScript.
- **Components:** Building blocks of a React app (can be functional or class-based).
- **Props:** Data passed from a parent to a child component.
- **State:** Data that changes over time within a component.
- **Lifecycle:** How a component is created, updated, and removed.

2. Create-react-app Setup

Create-react-app is a tool that sets up everything you need to start a React project without manual configuration.

Steps to Create a React App:

1. Install Node.js & npm (if not already installed)

Check with `node -v` `npm -v`

2. Create a new React project:

`npx create-react-app my-app`

3. Navigate to the project folder:

`cd my-app`

4. Start the development server:

npm start (Opens the app in the browser at <http://localhost:3000>)

```
Select Windows PowerShell

C:\Users\PMLS>cd documents

C:\Users\PMLS\Documents>npx create-react-app my-app
Need to install the following packages:
create-react-app@5.1.0
Ok to proceed? (y) y

npm warn deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if you want a good and tested way to coalesce async requests by a key value, which is much more comprehensive and powerful.
npm warn deprecated uid-number@0.0.6: This package is no longer supported.
npm warn deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
npm warn deprecated fstream@1.0.12: This package is no longer supported.
npm warn deprecated fstream-ignore@1.0.5: This package is no longer supported.
npm warn deprecated rimraf@2.7.1: Rimraf versions prior to v4 are no longer supported
npm warn deprecated tar@2.2.2: This version of tar is no longer supported, and will not receive security updates. Please upgrade asap.
create-react-app is deprecated.

You can find a list of up-to-date React frameworks on react.dev
For more info see:https://react.dev/link/cra

This error message will only be shown once per install.

Creating a new React app in C:\Users\PMLS\Documents\my-app.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...

added 1323 packages in 11m

269 packages are looking for funding
  run `npm fund` for details
Git repo not initialized Error: Command failed: git --version
    at genericNodeError (node:internal/errors:983:15)
    at wrappedFn (node:internal/errors:537:14)
    at checkExecSyncError (node:child_process:882:11)
    at execSync (node:child_process:954:15)
    at tryGitInit (C:\Users\PMLS\Documents\my-app\node_modules\react-scripts\scripts\init.js:46:5)
    at module.exports (C:\Users\PMLS\Documents\my-app\node_modules\react-scripts\scripts\init.js:276:7)
    at [eval]:3:14
    at runScriptInThisContext (node:internal/vm:209:10)
    at node:internal/process/execution:449:12
    at [eval]-wrapper:6:24 {
  status: 1,
  signal: null,
  output: [ null, null, null ],
  pid: 59048,
  stdout: null,
  stderr: null
}
```

```
Select Windows PowerShell

Git repo not initialized Error: Command failed: git --version
    at genericNodeError (node:internal/errors:983:15)
    at wrappedFn (node:internal/errors:537:14)
    at checkExecSyncError (node:child_process:882:11)
    at execSync (node:child_process:954:15)
    at tryGitInit (C:\Users\PMLS\Documents\my-app\node_modules\react-scripts\scripts\init.js:46:5)
    at module.exports (C:\Users\PMLS\Documents\my-app\node_modules\react-scripts\scripts\init.js:276:7)
    at [eval]:3:14
    at runScriptInThisContext (node:internal/vm:209:10)
    at node:internal/process/execution:449:12
    at [eval]-wrapper:6:24 {
  status: 1,
  signal: null,
  output: [ null, null, null ],
  pid: 59048,
  stdout: null,
  stderr: null
}

Installing template dependencies using npm...

added 17 packages, and changed 1 package in 1m

269 packages are looking for funding
  run `npm fund` for details
Removing template package using npm...

removed 1 package, and audited 1340 packages in 32s

269 packages are looking for funding
  run `npm fund` for details

9 vulnerabilities (3 moderate, 6 high)

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

Success! Created my-app at C:\Users\PMLS\Documents\my-app
```

```
Select Windows PowerShell
Success! Created my-app at C:\Users\PMLS\Documents\my-app
Inside that directory, you can run several commands:

  npm start
    Starts the development server.

  npm run build
    Bundles the app into static files for production.

  npm test
    Starts the test runner.

  npm run eject
    Removes this tool and copies build dependencies, configuration files
    and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

  cd my-app
  npm start

Happy hacking!

C:\Users\PMLS\Documents>cd my-app
C:\Users\PMLS\Documents\my-app>npm start

> my-app@0.1.0 start
> react-scripts start

(node:7096) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' option.
(Use `node --trace-deprecation ...` to show where the warning was created)
(node:7096) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' option.
Starting the development server...
Compiled successfully!

You can now view my-app in the browser.

Local:    http://localhost:3000
```

3. JSX (JavaScript XML)

JSX is a syntax extension for JavaScript used in React. It looks like HTML but is actually JavaScript under the hood. It allows you to write UI elements directly inside JavaScript.

Example:

Without JSX:

```
const element = React.createElement('h1', null, 'Hello, React!');
```

With JSX:

```
const element = <h1>Hello, React!</h1>;
```

Rules

- JSX must have one parent element (wrap multiple elements in a div or `<>` fragment).
- Use `className` instead of `class` for CSS.
- Embed JavaScript using curly braces `{}`:

```
const name = "Zainab";
const element = <h1>Hello, {name}</h1>;
```

4. Components (Functional)

- Components are building blocks of a React app.
- A component is a function or class that returns JSX.
- Functional components are simple JavaScript functions.

Example:

```
function Welcome() {  
  return <h2>Welcome to React!</h2>;  
}
```

You can use this component inside another like an HTML tag:

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

5. Props (Properties)

- Props are a way to pass data from a parent component to a child component.
- Think of it like HTML attributes.
- Props are read-only (cannot be modified by the child component).

Example:

```
function Welcome(props) {  
  return <h2>Hello, {props.name}</h2>;  
}  
  
function App() {  
  return (  
    <div>  
      <Welcome name="Zainab" />  
      <Welcome name="Ali" />  
    </div>  
  );  
}
```

- name="Zainab" → passed as a prop to Welcome.
- Output:

Hello, Zainab

Hello, Ali

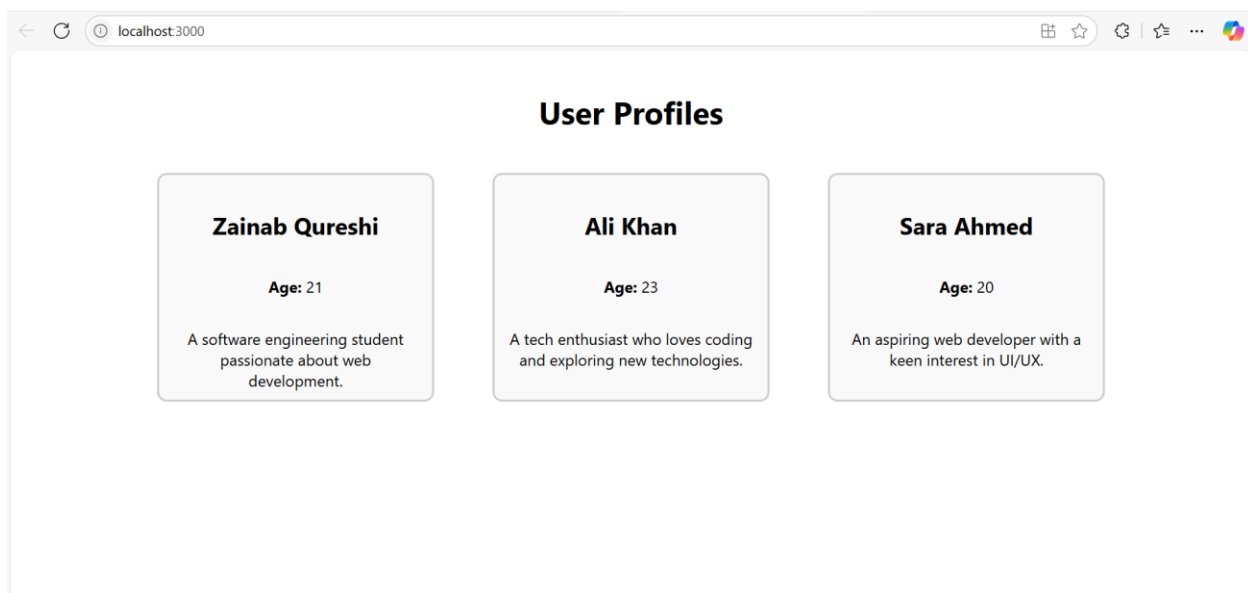
Inside the src/ folder, we'll mainly work with these:

- App.js → Main component displayed on the screen (we'll often start editing here).
- App.css → Styles for the App component.
- index.js → Entry point of the application; usually not modified often.
- index.css → Global styles for the app.

Practice Code:

```
JS App.js x
src > JS App.js > UserCard
1 function UserCard(props) {
2   return (
3     <div style={{
4       border: "2px solid #ccc",
5       borderRadius: "10px",
6       padding: "15px",
7       margin: "20px",
8       width: "250px",
9       height: "200px",
10      backgroundColor: "#f9f9f9",
11      display: "flex",
12      flexDirection: "column",
13      justifyContent: "space-between"
14    }}>
15      <h2>{props.name}</h2>
16      <p><strong>Age:</strong> {props.age}</p>
17      <p style={{ flexGrow: 1 }}>{props.bio}</p>
18    </div>
19  );
20 }
21
22 function App() {
23   return (
24     <div style={{ textAlign: "center", padding: "20px" }}>
25       <h1>User Profiles</h1>
26
27       <div style={{
28         display: "flex",
29         justifyContent: "center",
30         alignItems: "stretch",
```

```
JS App.js x
src > JS App.js > UserCard
22 function App() {
26
27   <div style={{
28     display: "flex",
29     justifyContent: "center",
30     alignItems: "stretch",
31     gap: "20px"
32   }}>
33     <UserCard
34       name="Zainab Qureshi"
35       age="21"
36       bio="A software engineering student passionate about web development."
37     />
38
39     <UserCard
40       name="Ali Khan"
41       age="23"
42       bio="A tech enthusiast who loves coding and exploring new technologies."
43     />
44
45     <UserCard
46       name="Sara Ahmed"
47       age="20"
48       bio="An aspiring web developer with a keen interest in UI/UX."
49     />
50   </div>
51 </div>
52 };
53 }
54
```



Conclusion:

This task helped me understand the core concepts of React.js and how it simplifies building dynamic, reusable user interfaces. I successfully created a React project, practiced JSX, built functional components, and used props for passing data.