Introduction to REACT.js

1. React overview

React is a popular open-source JavaScript library used for building user interfaces (UIs), particularly for single-page applications where the user interacts with the app without requiring full page reloads. It was developed by Facebook and is maintained by Facebook and a community of developers.

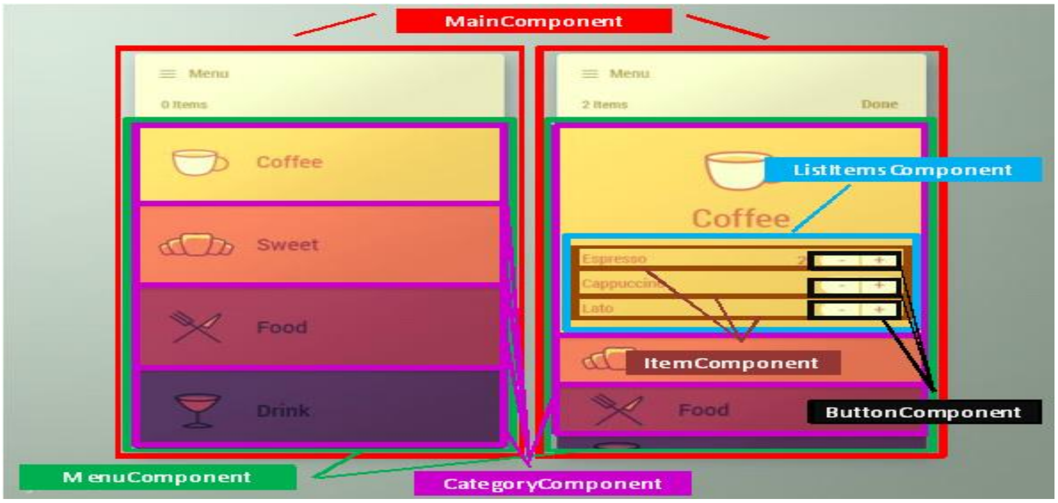
Key features of React:

1. **Component-Based Architecture**: React breaks down the UI into smaller, reusable components. Each component is responsible for rendering a part of the UI and can be independently updated when its state changes.
2. **Declarative**: In React, you describe what the UI should look like for each state of the application. React automatically updates the UI when the state or data changes, making it easier to manage dynamic user interfaces.
3. **Virtual DOM**: React uses a virtual DOM, which is a lightweight copy of the actual DOM (Document Object Model). When a component’s state changes, React updates the virtual DOM first and then compares it to the real DOM. Only the parts of the UI that need to change are updated, making the app more efficient.
4. **JSX (JavaScript XML)**: React allows you to write components using JSX, which is a syntax extension that lets you write HTML-like code within JavaScript. While it's not required, JSX makes it easier to define the structure of your components in a readable and familiar way.

2. React Components

React components are the building blocks of a React application.

They are JavaScript functions or classes that return a part of the UI (usually in the form of JSX). Components allow you to break down your UI into smaller, reusable, and self-contained pieces of code, making it easier to maintain and scale.



3. React Installation

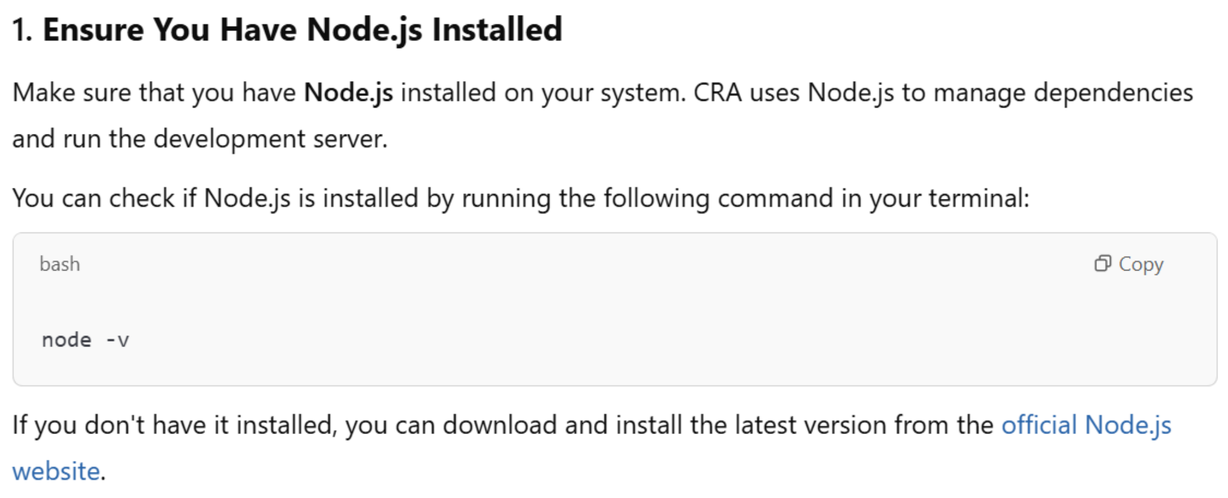
There are 2 ways to install react

1. Using create-react-app (deprecated)

Create React App is a comfortable environment for learning React, and is the best way to start building a new single-page application in React.

It sets up your development environment so that you can use the latest JavaScript features, provides a nice developer experience, and optimizes your app for production.

Create React App is an officially supported way to create single-page React applications. It offers a modern build setup with no configuration.



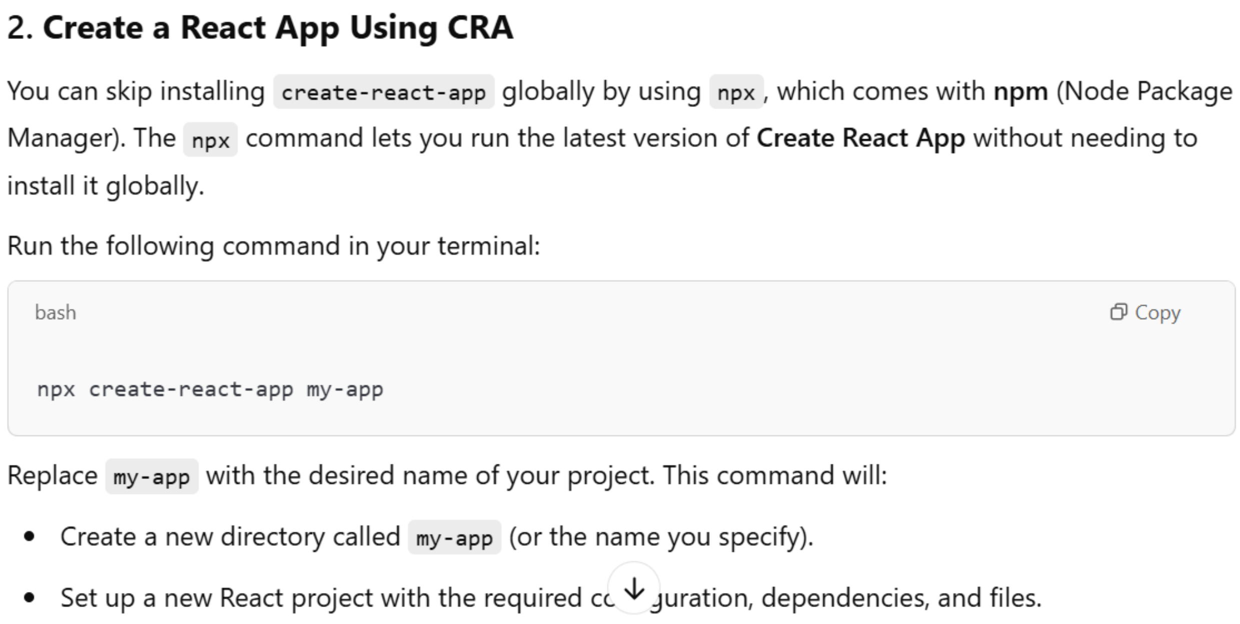
1. Ensure You Have Node.js Installed

Make sure that you have **Node.js** installed on your system. CRA uses Node.js to manage dependencies and run the development server.

You can check if Node.js is installed by running the following command in your terminal:

```bash
node -v
```

If you don't have it installed, you can download and install the latest version from the official Node.js website.



2. Create a React App Using CRA

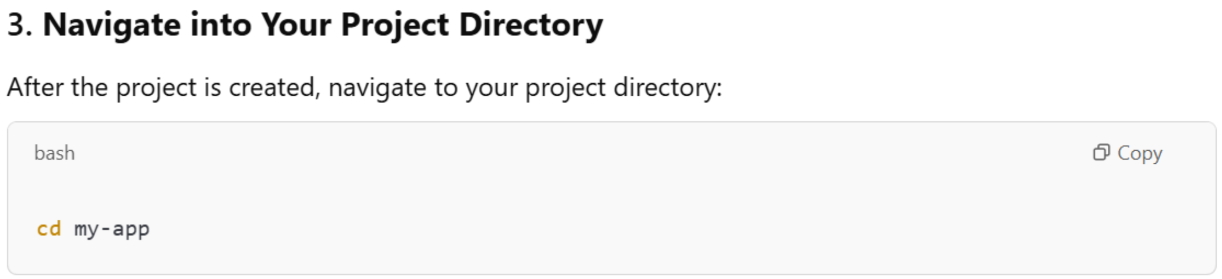
You can skip installing `create-react-app` globally by using `npx`, which comes with **npm** (Node Package Manager). The `npx` command lets you run the latest version of **Create React App** without needing to install it globally.

Run the following command in your terminal:

```bash
npx create-react-app my-app
```

Replace `my-app` with the desired name of your project. This command will:

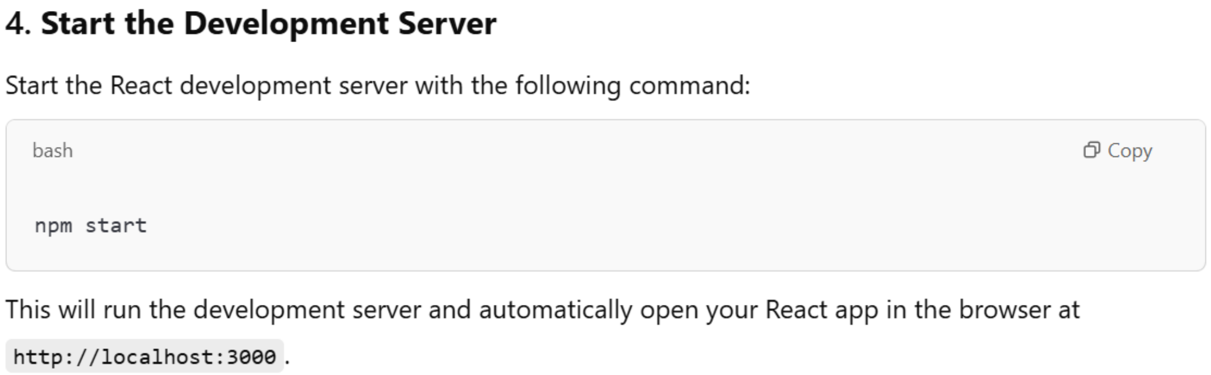
- Create a new directory called `my-app` (or the name you specify).
- Set up a new React project with the required configuration, dependencies, and files.



3. Navigate into Your Project Directory

After the project is created, navigate to your project directory:

```bash
cd my-app
```

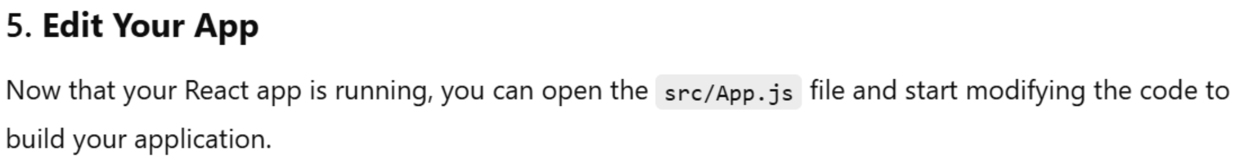


4. Start the Development Server

Start the React development server with the following command:

```bash
npm start
```

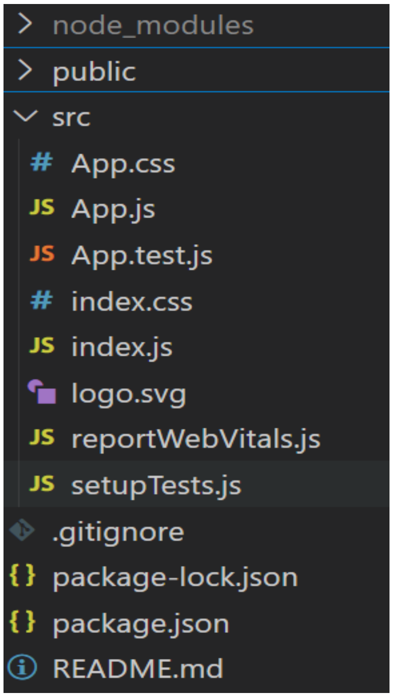
This will run the development server and automatically open your React app in the browser at `http://localhost:3000`.



5. Edit Your App

Now that your React app is running, you can open the `src/App.js` file and start modifying the code to build your application.

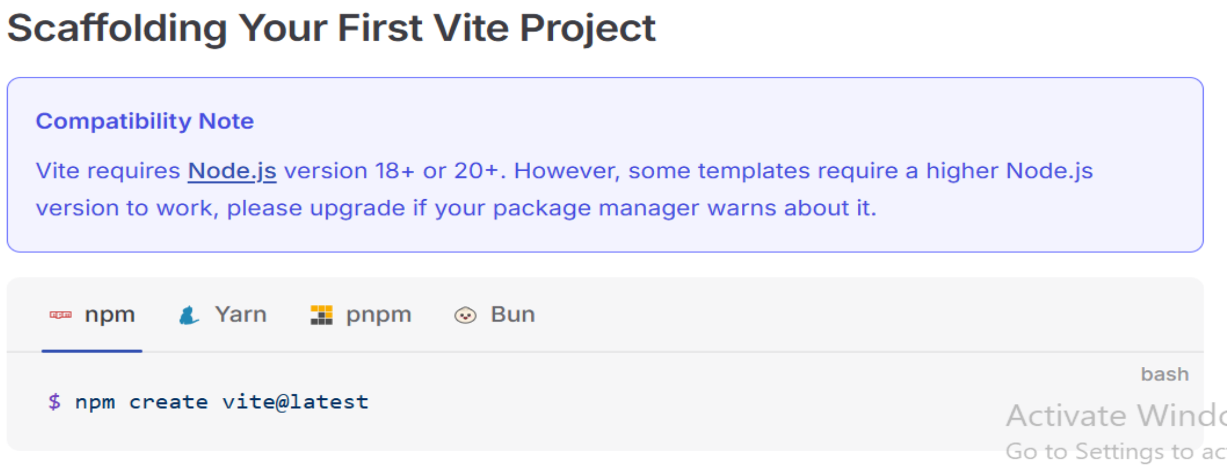
The project folder structure will look like this



```
> node_modules
> public
v src
    App.css
    App.js
    App.test.js
    index.css
    index.js
    logo.svg
    reportWebVitals.js
    setupTests.js
  .gitignore
  package-lock.json
  package.json
  README.md
```

2. Using Vite

Vite is a build tool that aims to provide a faster and leaner development experience for modern web projects. It consists of a dev server that provides rich feature enhancements over native ES modules, for example extremely fast Hot Module Replacement (HMR).



Scaffolding Your First Vite Project

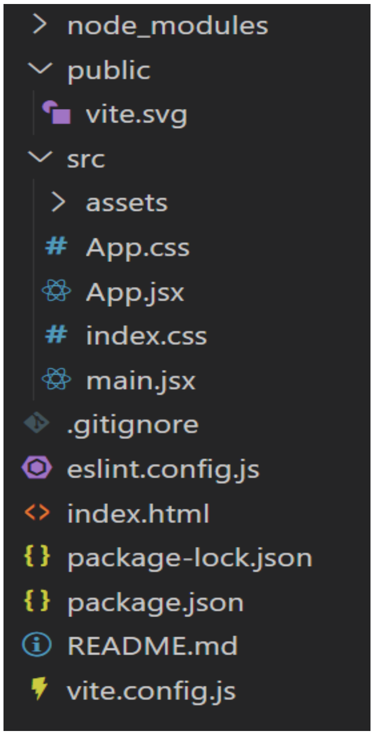
Compatibility Note

Vite requires Node.js version 18+ or 20+. However, some templates require a higher Node.js version to work, please upgrade if your package manager warns about it.

```bash
$ npm create vite@latest
```

Then follow the prompts!

The project folder structure will look like this



```
> node_modules
∨ public
    vite.svg
∨ src
  > assets
    # App.css
    App.jsx
    # index.css
    main.jsx
  .gitignore
  eslint.config.js
  <> index.html
  {} package-lock.json
  {} package.json
  ⓘ README.md
  vite.config.js
```

Further we can add more components and customize it.

Note: To create this handout, references are taken from different sources. Use it within organization. Do not share.