FSD ASSIGNMENT - 5

By-Yashvardhan Tekavade
PB-15 Batch-1
Panel-1 TY-CSF

Aim: Design and develop an interactive user interface using React.

Objectives:

1. Articulate what React is and why it is useful.
2. Explore the basic architecture of a React application.
3. Use React components to build interactive interfaces

Theory

1. **What is React? Steps to run React app using create-react-app.**

React is an open-source JavaScript library for building user interfaces, particularly for single-page applications (SPAs) and dynamic web applications. It is developed and maintained by Facebook and a community of individual developers and companies. React is known for its component-based architecture and its ability to efficiently update and render components as data changes. Here are the steps to run a React app using create-react-app, a popular tool for creating and managing React applications:

Step 1: Install Node.js and npm
Before you can use create-react-app and run a React application, you need to have Node.js and npm (Node Package Manager) installed on your computer. You can download them from the official website: Node.js.

Step 2: Install create-react-app Globally
You can install create-react-app globally on your system using npm. Open your terminal or command prompt and run the following command:
bash **npm install -g create-react-app**
This command installs create-react-app globally, allowing you to create new React applications using it.

Step 3: Create a New React Application
Once create-react-app is installed, you can create a new React application by running the following command:
bash **npx create-react-app my-react-app**

Replace my-react-app with your preferred project name. This command will create a new directory with the specified name and set up a basic React application structure for you.

Step 4: Navigate to Your Project Directory
Change your working directory to the newly created React application directory:
bash **cd my-react-app**

Step 5: Start the Development Server
To run your React application in development mode, use the following command:
bash **npm start**
This command starts the development server, compiles your React code, and opens the application in your default web browser. You can access it at http://localhost:3000/. As you make changes to your React components, the application will automatically reload in the browser.

Step 6: Edit Your React App
Now, you can start editing your React app. The main entry point for your React application is the src/index.js file. You can create and modify React components in the src directory.

Step 7: Build and Deploy Your App
When you're ready to deploy your React app for production, you can build it using the following command:
bash **npm run build**

This command generates an optimized production build of your React application in the build directory. You can then deploy the contents of the build directory to a web server or a hosting service of your choice.
That's it! You've successfully created a new React application using create-react-app and started a development server to run your app locally. You can now build and expand your React application according to your project requirements.

2. Passing data through props (Small example)

Passing data through props is a fundamental concept in React. It allows you to send data from a parent component to a child component. Here's a small example demonstrating how to pass data via props in a React application:

```
// App.js (Main Application File)
import React from 'react';
import Parent from './Parent';
function App() {
  return (
    <div>
```

```
      <h1>App Component</h1>
      <Parent />
    </div>
  );
}
export default App;
```

FAQs

What are React states and hooks?

React States:
A state is a built-in feature of React components that allows you to store and manage data that can change over time. States are essential for creating dynamic and interactive components. Each React component can have its own state, and when the state data changes, React will automatically re-render the component to reflect those changes in the user interface.
To use states in a class component, you typically define the state object in the constructor and access it using this.state'

React Hooks:

React introduced hooks in React 16.8 as a way to add state and other features to functional components. Before hooks, only class components could have state. With hooks, functional components can manage state and perform other side effects, making it easier to reuse logic across components and write more concise and readable code.
The useState hook, for example, allows you to add state to a functional component.

Sample Problem Statements:

1. Design and develop a UI for calculator using React.

You can include features like-

Mathematical operations

Inserting values

Decimal values must be supported