React Practice Sessions - Full Guide

Academic Session 1 (45 minutes)

Goal: Reinforce theoretical understanding and do hands-on review of React basics.

PART 1: Quick Recap and Discussion (15 minutes)

1. What is React?

- React is a JavaScript library for building user interfaces
- Created by Facebook in 2013
- Used widely in production: Facebook, Instagram, Netflix, Airbnb, etc.
- React is declarative, component-based, and uses a virtual DOM

2. Why is React so popular?

- Fast rendering
- Reusable components
- Backed by a strong community and regular updates

3. What is JSX?

- JSX stands for JavaScript XML
- Allows writing HTML-like syntax in JavaScript
- JSX must have one root element and uses className, htmlFor, etc.

4. What are Functional Components?

- React components written as simple JavaScript functions
- Use return to display JSX

Example:

```
function Welcome() {
  return <h1>Hello, React!</h1>;
}
```

PART 2: Practical Activities (30 minutes)

Activity 1: Set Up React Project

- Use npx create-react-app practice-session
- Open with VS Code
- Run with npm start

React Practice Sessions - Full Guide

Activity 2: Create a Functional Component

- Create a file Welcome.js
- Export and import it into App.js
- Render it inside <App />

Activity 3: JSX Practice

- Add JSX with expressions like const name = "John"; and render <h2>Hello, {name}</h2>
- Add inline styles and use className

Academic Session 2 (45 minutes)

Goal: Review props, useState hook, and intro to Git/GitHub interactions

PART 1: Mini Lecture and Examples (20 minutes)

1. Props in Functional Components

Props are arguments passed to components

Example:

```
function Greeting(props) {
  return <p>Hello, {props.name}!</p>;
}
```

2. Hooks in React

Special functions that let you use state and other features in functional components

useState is used to store and update values

Example:

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

function Counter() {
  const [count, setCount] = useState(0);
  return (
    <div>
      <p>Count: {count}</p>
      <button onClick={() => setCount(count + 1)}>Increase</button>
    </div>
```

```
  );
}
```

3. GitHub Basics

- git clone [repo]: download repo to local
- git pull: update local repo
- git push: upload changes
- Use Access Token instead of password (especially on a new device)

PART 2: Hands-on Practice (25 minutes)

Activity 1: Create a ProfileCard Component with Props

- Props: name, bio, image
- Render inside App.js

Activity 2: Create a Counter with useState

- Add Increase and Reset buttons

Activity 3: GitHub Push Practice

- Initialize git in project
- Connect to GitHub repo
- git add .
- git commit -m "Initial commit"
- git push origin main

Academic Session 3 (45 minutes)

Goal: Understand and practice React event handling

PART 1: Handling Events in React (20 minutes)

1. What are Events in React?

- Events are actions like clicks, typing, mouse movement, etc.
- React uses camelCase for event handlers like onClick, onChange, etc.

2. Using onClick

```
function ClickButton() {
  const handleClick = () => {
```

```
    alert("Button clicked!");
  };
  return <button onClick={handleClick}>Click me</button>;
}
```

3. Using onChange

```
function InputBox() {
  const handleChange = (e) => {
    console.log(e.target.value);
  };
  return <input type="text" onChange={handleChange} />;
}
```

Task for Students:

- Create an input field and display the text live while typing
- Create a button that when clicked changes a greeting message

Academic Session 4 (45 minutes)

Goal: Build a functional counter app using React state and events

PART 1: Building the Counter App (25 minutes)

1. Create a new component MyCounter.js

Use useState to manage count

Add 3 buttons: Increase, Decrease, Reset

```
function MyCounter() {
  const [count, setCount] = useState(0);
  return (
    <div>
      <h2>Count: {count}</h2>
      <button onClick={() => setCount(count + 1)}>Increase</button>
      <button onClick={() => setCount(count - 1)}>Decrease</button>
      <button onClick={() => setCount(0)}>Reset</button>
    </div>
  );
```

}

2. Import and render the component inside App.js

PART 2: Student Practice (20 minutes)

Tasks for Students:

- Extend the counter app to allow the user to input a number and increase the count by that number
- Add a message that says You reached 10! when count hits 10

Notes

- Ensure all students run npm install if cloning a React repo
- Use .gitignore to ignore node_modules
- Check React DevTools installed for browser