Week 6
React

Hands on 1 :

Quick help & options

1. **Clean, bulletized step-by-step lab instructions** (for a handout or checklist).

2. **Generate the minimal `App.js` code** that displays the heading "welcome to the first session of React."

3. **Walk you through environment setup** (installing Node.js/NPM, creating the app, running it).

4. **Explain concepts** from the doc (SPA vs MPA, virtual DOM, pros/cons of React, etc.) in plain language or for a presentation.

5. **Convert it into a polished lab guide or worksheet** (Markdown, HTML, etc.).

6. **Troubleshoot common issues** if you're stuck running `npm start` or similar.

Sample: minimal `App.js` to show the heading

IF YOU'VE ALREADY RUN:

```
npx create-react-app myfirstreact
cd myfirstreact
```

THEN REPLACE THE CONTENTS OF `SRC/APP.JS` WITH:

```
import React from 'react';

function App() {
  return (
    <div style={{ textAlign: 'center', marginTop: '50px' }}>
      <h1>welcome to the first session of React</h1>
    </div>
  );
}

export default App;
```

THEN RUN:

```
npm start
```

Week 6

And open `http://localhost:3000` in your browser.

Environment prerequisites (if you want the full setup spelled out)

- Install Node.js (which includes NPM) from the official site (ensure you get a current LTS version).
- Verify installation :
  ```
  node --version
  ```

-
  ```
  npm --version
  ```
-
- Create the app:
  ```
  npx create-react-app myfirstreact
  ```

-
  ```
  cd myfirstreact
  ```

-
  ```
  npm start
  ```

Hands on 2 :

Great. Here’s a minimal copy-paste-ready version assuming you’ve done:

```
npx create-react-app StudentApp
cd StudentApp
```

Then under `src/` create the three component files and edit `App.js` as below.

src/Components/Home.js

```
import React from 'react';

function Home() {
  return (
    <div>
      <h2>Welcome to the Home page of Student Management
Portal</h2>
    </div>
  );
}

export default Home;
```

src/Components/About.js

Week 6

```
import React from 'react';

function About() {
  return (
    <div>
      <h2>Welcome to the About page of the Student Management 
Portal</h2>
    </div>
  );
}

export default About;
```

src/Components/Contact.js

```
import React from 'react';

function Contact() {
  return (
    <div>
      <h2>Welcome to the Contact page of the Student 
Management Portal</h2>
    </div>
  );
}

export default Contact;
```

src/App.js

```
import React from 'react';
import Home from './Components/Home';
import About from './Components/About';
import Contact from './Components/Contact';

function App() {
  return (
    <div style={{ padding: '20px', fontFamily: 'sans-
serif' }}>
      <h1>Student Management Portal</h1>
      <hr />
      <Home />
      <About />
      <Contact />
    </div>
  );
}

export default App;
```

(Optional) src/index.js — default from CRA is fine, but for completeness:

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import App from './App';

const root =
ReactDOM.createRoot(document.getElementById('root'));
root.render(<App />);
```

To run:

```
npm start
```

Then open `http://localhost:3000` in your browser; you'll see the portal heading and the three component messages.

Hands on 3 :

Got it — this third lab is about creating a **function component** in a React project called **scorecalculatorapp**, styling it with a CSS file, and calculating/displaying the average score.

Here's a **complete, copy-paste-ready** version of the lab:

1️⃣ Create the project

```
npx create-react-app scorecalculatorapp
cd scorecalculatorapp
```

2️⃣ Create the component folder and file

src/Components/CalculateScore.js

```
import React from 'react';
import '../Stylesheets/mystyle.css';

function CalculateScore({ name, school, total, goal })
```

Week 6

```
{
  const average = total / goal;
  return (
    <div className="score-card">
      <h2>Student Score Calculator</h2>
      <p><strong>Name:</strong> {name}</p>
      <p><strong>School:</strong> {school}</p>
      <p><strong>Total Marks:</strong> {total}</p>
      <p><strong>Goal Marks:</strong> {goal}</p>
      <p className="highlight"><strong>Average Score:</
strong> {average.toFixed(2)}</p>
    </div>
  );
}

export default CalculateScore;
```

3️⃣ Create the stylesheet folder and file

src/Stylesheets/mystyle.css

```
.score-card {
  border: 2px solid #333;
  padding: 20px;
  margin: 20px auto;
  width: 320px;
  border-radius: 10px;
  background-color: #f4f4f4;
  font-family: Arial, sans-serif;
}

.score-card h2 {
  text-align: center;
  color: #444;
}

.score-card p {
  margin: 5px 0;
}

.highlight {
  color: #0066cc;
  font-weight: bold;
}
```

Week 6

4️⃣ Edit App.js to render the component

src/App.js

```
import React from 'react';
import CalculateScore from './Components/CalculateScore';

function App() {
  return (
    <div>
      <h1 style={{ textAlign: 'center' }}>Student Management
Portal</h1>
      <CalculateScore
        name="John Doe"
        school="ABC High School"
        total={450}
        goal={500}
      />
    </div>
  );
}

export default App;
```

5️⃣ Run the app

```
npm start
```

Open **http://localhost:3000** and you'll see a styled score card with the student's details and average score.

Hands on 4 :

Got it — this fourth lab is about **React class components** and lifecycle methods, specifically `componentDidMount()` for data fetching and `componentDidCatch()` for error handling, using a `blogapp` project.

Here's a complete working version:

1️⃣ Create the project

```
npx create-react-app blogapp
cd blogapp
```

Week 6

2️⃣ Create `Post.js` (model-like file)

src/Post.js

```
class Post {
  constructor(userId, id, title, body) {
    this.userId = userId;
    this.id = id;
    this.title = title;
    this.body = body;
  }
}

export default Post;
```

3️⃣ Create `Posts.js` (class component)

src/Posts.js

```
import React, { Component } from 'react';
import Post from './Post';

class Posts extends Component {
  constructor(props) {
    super(props);
    this.state = {
      posts: [],
      hasError: false
    };
  }

  loadPosts = () => {
    fetch('https://jsonplaceholder.typicode.com/posts')
      .then(response => response.json())
      .then(data => {
        const posts = data.map(
          p => new Post(p.userId, p.id, p.title, p.body)
        );
        this.setState({ posts });
      })
      .catch(error => {
        console.error('Error fetching posts:', error);
      });
  };

  componentDidMount() {
    this.loadPosts();
  }

  componentDidCatch(error, info) {
    console.error('Error caught in component:', error, info);
    alert('An error occurred while rendering posts.');
    this.setState({ hasError: true });
  }

  render() {
    if (this.state.hasError) {
      return <h2>Something went wrong.</h2>;
    }

    return (
      <div style={{ padding: '20px' }}>
        <h1>Blog Posts</h1>
        {this.state.posts.map(post => (
          <div key={post.id} style={{ marginBottom: '20px' }}
>
            <h3>{post.title}</h3>
            <p>{post.body}</p>
            <hr />
          </div>
        ))}
      </div>
    );
  }
}

export default Posts;
```

4️⃣ Edit `App.js` to use `Posts`

src/App.js

```
import React from 'react';
import Posts from './Posts';

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

export default App;
```

5️⃣ Run the app

```
npm start
```

Open http://localhost:3000 — the app will fetch and display blog posts from `jsonplaceholder.typicode.com`.

Hands on 5 :

Perfect. Here's a minimal copy-paste-ready implementation for **Lab 5** (styling with a CSS Module and inline conditional styling).

1. Create the CSS Module

src/CohortDetails.module.css

```
.box {
  width: 300px;
  display: inline-block;
  margin: 10px;
  padding: 10px 20px;
  border: 1px solid black;
  border-radius: 10px;
}

dt {
  font-weight: 500;
}
```

2. Example React component using it

src/components/CohortDetails.js (adjust path if your folder structure differs)

Week 6

```
import React from 'react';
import styles from '../CohortDetails.module.css';

function CohortDetails({ name, status, startDate, endDate })
{
  // Determine heading color
  const headingStyle = {
    color: status.toLowerCase() === 'ongoing' ? 'green' :
'blue',
  };

  return (
    <div className={styles.box}>
      <h3 style={headingStyle}>{name}</h3>
      <dl>
        <dt>Status:</dt>
        <dd>{status}</dd>
        <dt>Start Date:</dt>
        <dd>{startDate}</dd>
        <dt>End Date:</dt>
        <dd>{endDate}</dd>
      </dl>
    </div>
  );
}

export default CohortDetails;
```

3. Sample usage in `App.js`

src/App.js

```
import React from 'react';
import CohortDetails from './components/CohortDetails';

function App() {
  return (
    <div style={{ padding: '20px' }}>
      <h1>Academy Cohorts Dashboard</h1>
      <CohortDetails
        name="Frontend Cohort"
        status="ongoing"
        startDate="2025-07-01"
        endDate="2025-09-30"
      />
      <CohortDetails
        name="Backend Cohort"
        status="completed"
        startDate="2025-03-01"
        endDate="2025-05-30"
      />
    </div>
  );
}

export default App;
```

4. Run the app

```
npm install    # if not already done
npm start
```

Open `http://localhost:3000` to see styled cohort boxes. The `<h3>` for the ongoing cohort will be green; others blue. `<dt>` elements will have font-weight 500 per the CSS module.