**Exercise 1: Greeting with Props**

**Task**:

* Create a Greeting component that takes a name prop and displays "Hello, [name]!".
* Use default props if no name is provided.

**Solution**:

import React from "react";

const Greeting = ({ name = "Guest" }) => {

return <h1>Hello, {name}!</h1>;

};

export default Greeting;

**Greeting.css**

.greeting {

color: #4caf50;

font-size: 2rem;

text-align: center;

margin-top: 20px;

}

**Exercise 2: Counter with State**

**Task**:

* Create a Counter component with a button to increment a count stored in the state.

**Solution**:

import React, { useState } from "react";

const Counter = () => {

const [count, setCount] = useState(0);

return (

<div>

<h2>Count: {count}</h2>

<button onClick={() => setCount(count + 1)}>Increment</button>

</div>

);

};

export default Counter;

**Counter.css**

.counter {

text-align: center;

margin-top: 30px;

}

.increment-btn {

padding: 10px 20px;

background-color: #007bff;

color: white;

border: none;

cursor: pointer;

}

.increment-btn:hover {

background-color: #0056b3;

}

**Exercise 3: Toggle Visibility**

**Task**:

* Create a ToggleText component that toggles the visibility of some text when a button is clicked.

**Solution**:

**ToggleText.js**

import React, { useState } from "react";

const ToggleText = () => {

const [isVisible, setIsVisible] = useState(true);

return (

<div>

<button onClick={() => setIsVisible(!isVisible)}>

{isVisible ? "Hide" : "Show"} Text

</button>

{isVisible && <p>This is the text to show or hide!</p>}

</div>

);

};

export default ToggleText;

**ToggleText.css**

.toggle-text {

text-align: center;

margin-top: 20px;

}

.toggle-btn {

padding: 10px 15px;

background-color: #28a745;

color: white;

border: none;

cursor: pointer;

}

.toggle-btn:hover {

background-color: #218838;

}

.text {

font-size: 1.2rem;

color: #333;

}

**App.js**

import React from "react";

import ToggleText from "./components/ToggleText";

const App = () => {

return <ToggleText />;

};

export default App;

**Exercise 4: Passing Props to Child Components**

**Task**:

* Create a Card component that takes title and description as props and displays them.

**Solution**:

**Card.js**

import React from "react";

const Card = ({ title, description }) => {

return (

<div>

<h3>{title}</h3>

<p>{description}</p>

</div>

);

};

export default Card;

**Card.css**

.card {

padding: 20px;

border: 1px solid #ddd;

border-radius: 8px;

margin: 10px 0;

box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

}

.card-title {

font-size: 1.5rem;

color: #333;

}

.card-description {

color: #555;

font-size: 1.1rem;

}

**App.js**

import React from "react";

import Card from "./components/Card";

const App = () => {

return (

<div>

<Card title="Card Title 1" description="This is the first card." />

<Card title="Card Title 2" description="This is the second card." />

</div>

);

};

export default App;

**Exercise 5: Dynamic Props with State**

**Task**:

* Create a Profile component with name and age props. Use a button to update the age from the parent component.

**Solution**:

**Profile.js**

import React from "react";

const Profile = ({ name, age }) => {

return (

<div>

<h3>Name: {name}</h3>

<p>Age: {age}</p>

</div>

);

};

export default Profile;

**Profile.css**

.profile {

text-align: center;

margin-top: 20px;

}

.profile h3 {

color: #007bff;

}

.profile p {

color: #333;

}

**App.js**

import React, { useState } from "react";

import Profile from "./components/Profile";

const App = () => {

const [age, setAge] = useState(25);

return (

<div>

<Profile name="John Doe" age={age} />

<button onClick={() => setAge(age + 1)}>Increase Age</button>

</div>

);

};

export default App;

**Exercise 6: Form with State**

**Task**:

* Create a Login form with email and password fields. Display the entered values when the form is submitted.

**Solution**:

**Login.js**

import React, { useState } from "react";

const Login = () => {

const [email, setEmail] = useState("");

const [password, setPassword] = useState("");

const handleSubmit = (e) => {

e.preventDefault();

console.log("Email:", email);

console.log("Password:", password);

};

return (

<form onSubmit={handleSubmit}>

<div>

<label>Email: </label>

<input type="email" value={email} onChange={(e) => setEmail(e.target.value)} />

</div>

<div>

<label>Password: </label>

<input

type="password"

value={password}

onChange={(e) => setPassword(e.target.value)}

/>

</div>

<button type="submit">Login</button>

</form>

);

};

export default Login;

**App.js**

import React from "react";

import Login from "./components/Login";

const App = () => {

return <Login />;

};

export default App;

**Exercise 7: Parent-Child Interaction**

**Task**:

* Create a Parent component that passes a color prop to a Child component. Use a button in the Parent to update the color dynamically.

**Solution**:

**Parent.js**

import React, { useState } from "react";

import Child from "./Child";

const Parent = () => {

const [color, setColor] = useState("blue");

return (

<div>

<Child color={color} />

<button onClick={() => setColor("red")}>Change to Red</button>

<button onClick={() => setColor("green")}>Change to Green</button>

</div>

);

};

export default Parent;

**File: Child.js**

import React from "react";

const Child = ({ color }) => {

return <h1 style={{ color }}>This is a {color} text!</h1>;

};

export default Child;

**File: App.js**

import React from "react";

import Parent from "./components/Parent";

const App = () => {

return <Parent />;

};

export default App;

**Exercise 8: List Rendering with Props**

**Task**:

* Create a TaskList component that takes a list of tasks as props and displays them.

**Solution**:

**TaskList.js**

import React from "react";

const TaskList = ({ tasks }) => {

return (

<ul>

{tasks.map((task, index) => (

<li key={index}>{task}</li>

))}

</ul>

);

};

export default TaskList;

**App.js**

import React from "react";

import TaskList from "./components/TaskList";

const App = () => {

const tasks = ["Task 1", "Task 2", "Task 3"];

return (

<div>

<h2>My Tasks</h2>

<TaskList tasks={tasks} />

</div>

);

};

export default App;