

## JavaScript and React Learning Plan

Day 1: 20th December 2024 Topic: JavaScript Fundamentals

Key Concepts:

Variables: var, let, and const for declaring variables.

Example:

```
let name = 'John';
```

```
const age = 30;
```

```
console.log(name, age);
```

Practice Question: Declare a variable for your favorite color and log it to the console.

Loops: Use for, while, and forEach to iterate data.

Example:

```
let fruits = ['Apple', 'Banana', 'Mango'];
```

```
fruits.forEach(fruit => console.log(fruit));
```

Practice Question: Print numbers from 1 to 10 using a loop.

Use Case: Used in data manipulation, iterating over API responses, and rendering lists in React.

Day 2: 21st December 2024 Topic: JavaScript Advanced Concepts

Key Concepts:

Functions: Regular, Arrow Functions, and IIFE.

Example:

```
const sum = (a, b) => a + b;
```

```
console.log(sum(5, 3));
```

Practice Question: Create a function that multiplies two numbers.

Async/Await:

Example:

```
async function fetchData() {  
  const response = await fetch('https://api.example.com/data');  
  const data = await response.json();  
  console.log(data);  
}  
  
fetchData();
```

Practice Question: Write an async function to fetch user data from a public API.

Use Case: Crucial for API calls in React applications to handle asynchronous data fetching.

Day 3: 22nd December 2024 Topic: React Basics

Key Concepts:

JSX:

Example:

```
const element = <h1>Hello, World!</h1>;
```

```
ReactDOM.render(element, document.getElementById('root'));
```

Practice Question: Create a JSX button that logs "Clicked" on click.

Functional Components:

Example:

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

Practice Question: Create a component to display your name.

Use Case: Foundational for building React UI components.

Day 4: 23rd December 2024 Topic: State Management with useState

Key Concepts:

useState Hook:

Example:

```
const [count, setCount] = useState(0);  
  
return (  
  <button onClick={() => setCount(count + 1)}>{count}</button>  
);
```

Practice Question: Create a counter application.

Use Case: Essential for managing dynamic UI changes based on user interactions.

Day 5: 24th December 2024 Topic: Props and Component Communication

Key Concepts:

Passing Props:

Example:

```
function Greeting({ name }) {  
  return <h1>Hello, {name}!</h1>;  
}  
  
<Greeting name="Sunny" />;
```

Practice Question: Pass a "title" prop to display it in a child component.

Use Case: Helps in transferring data between parent and child components.

Day 6: 25th December 2024 Topic: Redux Basics

Key Concepts:

Actions and Reducers:

Example:

```
const increment = () => ({ type: 'INCREMENT' });  
  
const counterReducer = (state = 0, action) => {  
  switch (action.type) {  
    case 'INCREMENT':  
      return state + 1;  
    default:  
      return state;  
  }  
};
```

Practice Question: Write an action to decrement a counter and a reducer to handle it.

Use Case: Used for managing complex application state in large projects.

Day 7: 26th December 2024 Topic: Redux Toolkit with createSlice

Key Concepts:

createSlice:

Example:

```
const counterSlice = createSlice({  
  name: 'counter',  
  initialState: 0,  
  reducers: {  
    increment: state => state + 1,  
    decrement: state => state - 1  
  }  
});
```

Practice Question: Create a slice to handle a to-do list.

Use Case: Simplifies Redux setup and reduces boilerplate code.

Day 8: 27th December 2024 Topic: Middleware with Redux Saga

Key Concepts:

Generator Functions:

Example:

```
function* fetchData() {  
  const data = yield call(apiCall);
```

```
yield put({ type: 'DATA_FETCHED', payload: data });  
}
```

Practice Question: Write a saga to fetch data and handle errors.

Use Case: Handles side effects like data fetching in a clean and testable way.

Day 9: 28th December 2024 Topic: API Integration with Axios

Key Concepts:

Data Fetching:

Example:

```
axios.get('https://api.example.com/data')  
  .then(response => console.log(response.data))  
  .catch(error => console.error(error));
```

Practice Question: Fetch a list of users and display them in a React component.

Use Case: Integrates frontend with backend services seamlessly.

Day 10: 29th December 2024 Topic: Error Handling and Optimizations

Key Concepts:

Error Boundaries:

Example:

```
class ErrorBoundary extends React.Component {  
  constructor(props) {  
    super(props);
```

```

    this.state = { hasError: false };
  }
  static getDerivedStateFromError(error) {
    return { hasError: true };
  }
  render() {
    if (this.state.hasError) {
      return <h1>Something went wrong.</h1>;
    }
    return this.props.children;
  }
}

```

Practice Question: Implement an error boundary in your application.

Use Case: Enhances user experience by gracefully handling errors.

Day 11: 30th December 2024 Topic: Optimizing React Performance

Key Concepts:

React.memo and useMemo:

Example:

```

const MemoizedComponent = React.memo(Component);

const result = useMemo(() => computeExpensiveValue(a, b), [a, b]);

```

Practice Question: Optimize a component to prevent unnecessary re-renders.

Use Case: Improves application performance, especially in large-scale projects.

Day 12: 31st December 2024 Topic: Building a Complete Feature

Task:

Build a To-Do List Application using React, Redux Toolkit, and Axios for API integration.

Include error handling, state management, and performance optimizations.

Use Case: Applies all learned concepts to create a practical and fully functional application.