■ React Learning Guide - Model 1

Designed for Learners | Styled with Creativity

■ Introduction to Vite & React Project Setup

• Vite is a modern, faster build tool compared to Create React App (CRA).

• Steps to create project: npm create vite@latest → choose React → JavaScript → cd project → npm install → npm run dev.

• Folder structure includes: node_modules, public, src, App.jsx, main.jsx, package.json, vite.config.js.

■ Introduction to ES6

• ES6 introduced major features like let, const, arrow functions, promises, and classes.

• let & const provide better scoping and immutability compared to var.

• Arrow functions give shorter syntax and support callbacks.

• Promises manage async operations (pending, fulfilled, rejected).

• Classes brought OOP to JavaScript with constructors and inheritance.

■ Introduction to JSX

• JSX = JavaScript Syntax Extension (JavaScript XML).

• It combines HTML + JavaScript for building UI.

• Browsers don’t understand JSX → needs Babel to compile to JS.

• Benefits: readable, error detection, better performance, secure (auto sanitization).

■ Introduction to Components

• Components are building blocks of React.

• 3 Types: Functional (simple, preferred), Class (older, complex), Higher-Order (reuse logic).

• Functional components now support hooks (state & lifecycle).

■■ Class Components, Props & Events

• Class components extend React.Component, support state & lifecycle.

• State: defined in constructor, updated with setState.

• Props: passed from parent to child, read-only.

• Event handling via attributes like onClick with arrow functions.

■ States in React

• State stores component-specific data, re-renders UI when changed.

• Local state vs Shared state.

• Props = external, read-only; State = internal, mutable.

• Use props to pass data, state to manage data.

■ Class Component Lifecycle

• 3 phases: Mounting, Updating, Unmounting.

• Mounting: constructor, render, componentDidMount.

• Updating: getDerivedStateFromProps, shouldComponentUpdate, render, getSnapshotBeforeUpdate, componentDidUpdate.

• Unmounting: componentWillUnmount for cleanup.

■ Passing Data Between Components

• Parent → Child: via props.

• Child → Parent: via callbacks.

• Siblings → Siblings: via Redux (not in scope here).

■ State vs Props Comparison

Aspect	State	Props
Definition	Internal data managed by component	Data passed from parent to child
Mutability	Mutable (can change with setState)	Immutable (read-only)
Scope	Local to component	External, received from parent
Usage	For dynamic, interactive behavior	For configuration & data passing

■ This guide is styled in Model 1 design – colorful, visual, and learner-friendly.