### **🛠️ Project Development Guidelines for Volunteers**

### Dear Volunteers,

### Thank you for contributing to this learning roadmap. To enhance student engagement and practical understanding, we have added project breaks throughout the curriculum. For each project break, we need:

### In-Class Coding Project

### A hands-on project that will be built live with students during the session.

### Should introduce the concepts of that particular unit.

### Must be simple, interactive, and explainable within class time.

### Example: https://drive.google.com/drive/folders/15sSI6iMDHo6GvOIwktrcnGgTCKAdwi0v

### Student Assignment Project

### A take-home assignment for students to implement on their own.

### Should build on the same concepts but with added complexity or variation.

### Must include a brief project description, objectives, and sample expected output if possible.

### Example:https://drive.google.com/drive/folders/1m8jMWzFhNi8cRcY8-NvEBVAvFu-gnpd-

### Please make sure the projects are:

### Appropriately scoped to the topics in each section.

### Well-documented, so any instructor can easily understand and guide students through them.

### Focused on encouraging creativity and real-world application.

### **React Roadmap**

#### **1. Introduction to React**

* What is React and Why Use It?
* Brief History of React
* SPA (Single Page Applications) Concept

#### **2. Setting Up a React Environment**

* Installing Node.js & npm
* Using Create React App (CRA)
* Folder Structure Overview

#### **3. JSX Basics**

* What is JSX?
* Embedding Expressions in JSX
* JSX vs HTML Differences

Project break

Focus: React setup, component structure, and JSX.

#### **4. Components in React**

* Functional vs Class Components
* Creating & Using Components
* Props in Components
* Reusability and Composition

#### **5. State Management**

* Using useState Hook
* Updating State
* State vs Props

Project break

Focus: Props, useState, component interaction.

#### **6. Handling Events**

* Event Handlers in JSX
* Passing Parameters to Handlers

#### **7. Conditional Rendering**

* if-else, Ternary, && Operator
* Rendering Based on Props/State

#### **8. Lists and Keys**

* Rendering Lists with map()
* Importance of Keys in Lists

Project break

Focus: State toggling, event handling and conditional rendering to dynamically switch UI themes

#### **9. Forms in React**

* Controlled vs Uncontrolled Components
* Handling Form Submission
* Validating Form Data

Project break

Focus: Controlled forms, list rendering, useState

#### **10. useEffect Hook**

* Side Effects in Functional Components
* useEffect with Dependencies
* Cleanup Functions

#### **11. Lifting State Up**

* Sharing State Between Components
* Prop Drilling

Project break

Focus: useEffect, lifting state up, prop drilling.

#### **12. Component Lifecycle (Class Components)**

* Mounting, Updating, Unmounting
* Lifecycle Methods: componentDidMount, etc.

#### **13. Routing in React**

* React Router Basics
* Route, Link, useParams, useNavigate

Project break

Focus: React Router, lifecycle methods for class components.

#### **14. Context API**

* Creating and Using Context
* useContext Hook
* Avoiding Prop Drilling

#### **15. Custom Hooks**

* Creating Custom Hooks
* Best Practices

Project break

Focus: Simple login simulation using Context API and custom hooks.

#### **16. React Performance Optimization**

* Memoization (React.memo, useMemo, useCallback)
* Lazy Loading & Code Splitting

#### **17. React and APIs**

* Fetching Data with fetch/axios
* Handling Loading and Error States

#### **18. Advanced Patterns**

* Render Props
* Higher-Order Components (HOCs)
* Compound Components

Project break

Focus: useContext, useMemo, API integration, performance.

### 

### 

### 

### 

### 

### **Node.js Roadmap**

**Node.js Project Breakdown Document**

### **1. Introduction to Node.js**

* What is Node.js?
* Why Use Node.js?
* Node.js vs Browser JavaScript

No project assigned

Focus: Understanding what Node.js is, its use cases, and how it differs from browser JS.

We can set up a quiz

### **2. Installing Node.js**

* Download & Install
* Using npm and npx

No project assigned

Focus: Installing Node.js, learning npm & npx usage.

### **3. Node.js Architecture**

* Event Loop
* Non-blocking I/O
* Callbacks in Node.js

**No project assigned**

* Focus: Understanding Event Loop, Non-blocking I/O, Callbacks.
* We can set up a quiz

### **4. Creating Your First App**

* Writing and Running JS in Node
* Basic CLI Logging and Input

**Project: CLI Personal Assistant**

* Description: A command-line tool that greets the user, shows the current date/time, and can do simple tasks like math calculations.
* Concepts Covered: CLI logging, reading user input, basic Node syntax.

### **5. Modules in Node.js**

* CommonJS (require / module.exports)
* ES Modules (import/export)
* Built-in Modules (fs, path, http, etc.)

**Project: Modular Calculator**

* Description: Build a calculator where each operation (add, subtract, etc.) is in a separate module.
* Concepts Covered: CommonJS modules, import/export, code reusability.

### **6. File System Module (fs)**

* Reading and Writing Files
* Synchronous vs Asynchronous Methods

**Project: Notes App**

* Description: CLI tool to create, read, list, and delete notes stored as text files.
* Concepts Covered: fs module, sync vs async methods, file handling.

### 

### **7. Creating an HTTP Server**

* http Module Basics
* Handling Requests and Responses
* Setting Status Codes and Headers

**Project: Mini Static Website Server**

* Description: Create a server using the http module that serves HTML files based on the URL path (/, /about, /contact) and handles 404 errors.
* Concepts Covered: http module, request and response handling, status codes, file reading with fs, routing.

### **8. npm and Package Management**

* Installing Local and Global Packages
* package.json and package-lock.json
* Semantic Versioning

**No project assigned**

* Focus: Understanding local vs global packages, reading and editing package.json, semantic versioning.

#### 