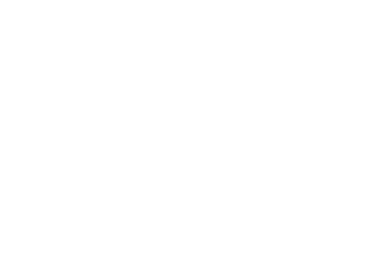
**Subject:** Integrative Programming Techniques 2



1-Month Front-End Development Project: Interactive Learning Platform

**Level:** 3rd Year Level

**Duration:** 4 Weeks (Self-Paced)

**Project Title:** "CodeLearn Hub" - An Interactive Programming Tutorial Platform

# 📌 Project Overview

Students will build an interactive platform where users can:  Browse programming tutorials

 Complete interactive coding exercises  Track learning progress

 Save/bookmark favorite tutorials

# 🗓 Weekly Breakdown

## Week 1: Project Setup & Core Structure

### Learning Objectives:

 Semantic HTML5 structure  CSS architecture planning  Basic JavaScript setup

### Tasks:

1. Create project repository with proper folder structure
2. Design wireframes for all pages (use Figma/Adobe XD or paper)
3. Build:

 Homepage with featured tutorials section  Navigation system

 Basic responsive layout

### Deliverables:

 Wireframe designs

 Static HTML/CSS structure  GitHub repository setup

## Week 2: Interactive UI Components

### Learning Objectives:

 DOM manipulation  Event handling

 CSS animations/transitions

### Tasks:

1. Implement:

 Interactive tutorial cards with hover effects  Filtering system (by language/difficulty)

 Dark/light mode toggle

1. Create:

 Animated loading states

 Responsive navigation menu

### Deliverables:

 Functional filtering system

 Theme switcher implementation  Smooth UI animations

## Week 3: Core Functionality

### Learning Objectives:

 Local storage API  Form handling

 Async operations

### Tasks:

1. Build:

 User progress tracking system  Bookmarking functionality

 Interactive code editor (simple version)

1. Implement:

 Form validation for user signup

 Local persistence for user preferences

### Deliverables:

 Working progress tracker

 Functional bookmark system  Code editor component

## Week 4: API Integration & Polish

### Learning Objectives:

 Fetch API

 Error handling

 Performance optimization

### Tasks:

1. Integrate with a coding API (e.g., Repl.it, CodePen)
2. Implement:

 Error boundaries and loading states  Accessibility improvements

 Performance optimizations

1. Final testing and debugging

### Deliverables:

 API-connected tutorial system  Fully accessible interface

 Optimized production build

# 🛠 Technical Requirements

### Core Stack:

 HTML5 (semantic markup)

 CSS3 (Flexbox/Grid, variables, animations)  Vanilla JavaScript (ES6+)

### Optional Advanced Features:

 Web Components

 IndexedDB for offline support

 Service Workers for PWA capabilities

# 📝 Assessment Criteria

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

💡 **Project Extensions (Bonus)**

**Category**

**Functionality Code Quality UI/UX**

**Innovation**

**Documentation**

**Point**

**s 30**

**25**

**20**

**15**

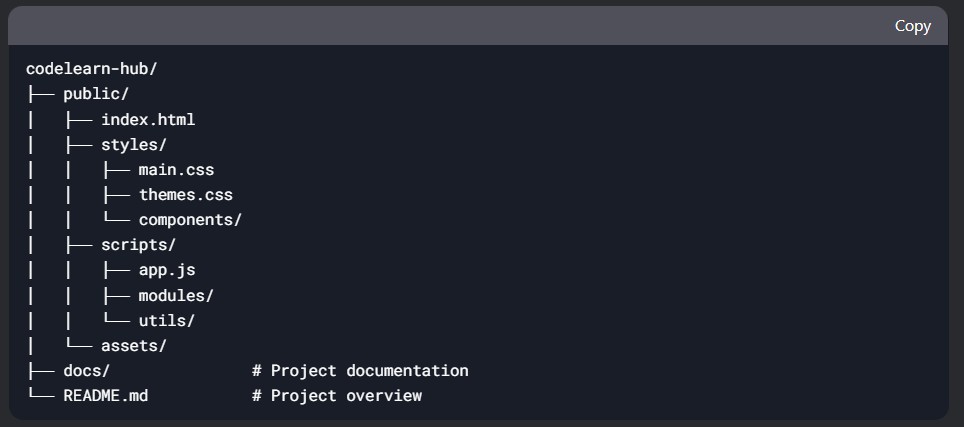
**10**

**Description**

**All core features work as intended Clean, modular, well-commented code Intuitive, responsive, visually appealing Creative solutions and extra features Clear README and code comments**

1. Implement user authentication (Firebase)
2. Add real-time collaboration features
3. Create an admin dashboard for content management
4. Build a Chrome extension version

# 📂 Suggested Project Structure

****

🎯 **Learning Outcomes**

By completing this project, students will:

1. Master complex DOM manipulation
2. Understand state management in vanilla JS
3. Learn API integration patterns
4. Develop skills in creating maintainable CSS
5. Gain experience with full project lifecycle

**Teacher's Note:** "This project simulates real-world front-end development challenges. Focus on writing clean, modular code and pay attention to user experience details. Don't hesitate to explore beyond the basic requirements!"