**-Project Name:**

*Codeblazer*

**Project Overview:**

**Objective**:

The proposed project aims to address a gap in programming education. Current courses often rely on random, abstract examples to teach concepts, which, while comprehensive, can be confusing and less effective. Our solution focuses on making the learning process more engaging and relevant by developing a complete application that mimics real-world scenarios. By the end of the course, learners will have built a practical, real-life application, making programming education more interactive and meaningful.

By developing the application in its entirety, learners will also apply concepts beyond programming, such as software architecture, business logic, databases, and more. The chosen concepts will be applied through the construction of an e-commerce platform from beginning to end. This approach emphasizes a holistic understanding of real-world software development. The main goal of the project is to engage, innovate, and enhance the learning experience by integrating multiple disciplines and providing a practical, hands-on approach to education.

**Project Summary**:

The project aims to develop an innovative programming education platform that enhances the learning process by focusing on real-world application development. Unlike traditional programming courses that rely on abstract examples, this platform will guide learners through the creation of a complete, practical application.

The main deliverable is a fully-fledged e-commerce application incorporating features commonly used in industry operations. Targeted at both beginner and advanced programmers, this platform will also serve as a valuable tool for institutions seeking to provide a more hands-on, comprehensive programming education. The project will be developed over one year, with key milestones including research and planning, course structure design, prototype development, testing and feedback, and final deployment.

**Business Goals & Objectives:**

* **Goal 1**: *Significate learning by applying a real-world example to the programming process.*
* **Goal 2**: *Encourage users to prioritize learning concepts other than focusing mainly on syntax by teaching how to use AI.*

**Stakeholders:**

**Primary Stakeholders**:

* Users enrolled in the course.

**Secondary Stakeholders**:

* Educational institutions that adopt the platform.
* Course designers and instructors.
* Companies seeking employees with practical programming skills.

**Target Market and Audience:**

**Target Audience**:

* Individuals who want a more meaningful, hands-on learning experience when learning to program, from beginners to advanced learners.

**Market Needs**:

* *The project addresses the need for programming education tools that focus on real-world applications, offering practical knowledge beyond abstract coding examples. Traditional programming courses often fail to show the relevance of coding in real-world scenarios, making this platform a valuable alternative.*

**Key Features:**

**Feature 1**:

* Step-by-step guidance for developing a complete e-commerce application, incorporating industry-relevant features like user authentication, shopping carts, and payment processing.

**Feature 2**:

* AI integration for optimizing code and providing smart suggestions, encouraging students to use AI as a tool in their programming.

**Feature 3**:

* Progress tracking and feedback, allowing learners to see their development over time, with detailed assessments and personalized learning paths.

**Constraints & Assumptions:**

* **Constraints**:
  + The project must be completed within a one-year timeframe.
* Limited funding may restrict the use of premium tools or services.
* **Assumptions**:
  + Users will have reliable internet access and basic knowledge of programming concepts.
  + Educational institutions and users will find value in real-world application development as a learning tool.

**Success Criteria:**

* **Criterion 1**: The platform should successfully guide users in developing a fully functioning e-commerce application by the end of the course. Therefore, they should have knowledge in all covered concepts from beginner to advanced.
* **Criterion 2**: Users should be able to apply broader programming-related concepts such as AI integration, database management, and software architecture.

**Risk Factors:**

* **Risk 1**: *Development delays due to unforeseen technical challenges, such as integrating real-world e-commerce features or AI-based tools. Also, payment and security can be a challenge to be implemented in the proposed application.*
* **Risk 2**: *Potential difficulty in ensuring that learners of varying skill levels can navigate the platform effectively.*

**Timeline:**

* **Milestone 1:** Research and planning, including gathering user requirements and outlining course content (Months 1-2).
* **Milestone 2:** Course structure design and initial wireframes for the platform (Months 3-4).
* **Milestone 3:** Prototype development, focusing on core features and the e-commerce application module (Months 5-6).
* **Milestone 4:** Testing and feedback phase, where users test the platform and provide feedback for improvements (Months 7-9).
* **Milestone 5:** Final adjustments and platform deployment, including user support and course materials (Months 10-12).

**Learning path documentation for Programming Education Platform**

**Overview:**

This document outlines the structured learning paths for the programming education platform, categorized into three levels: Beginner, Intermediate, and Advanced. Each section focuses on specific topics and skills to ensure a comprehensive learning experience.

**Learning Path**

* 1. **Beginner Level**

**Objective**: Introduce foundational programming concepts and basic coding skills.

**Core Topics:**

* + - **Case Study**: Translating business rules into the basic applicable coding concepts.
    - **HTML/ CSS**:
      * Bootstrap/ Tailwind
    - **JS**: DOM
      * **OOP + Best Practices:** encapsulation, inheritance, and polymorphism
    - **Flow Control and Programming Basics**
    - **GitHub**
      * **Version Control**
  1. **Intermediate Level**

**Objective**: Build on foundational knowledge and introduce more complex programming concepts.

**Core Topics:**

* + - **Intermediate Case Study**
    - **UI/UX**
      * **Design Principles**
      * **User research**
      * **Wireframing and Prototyping**
      * **Accessibility**
    - **React** 
      * **CSS Processor** 
        + **Sass, PostCSS**
      * **TYPESCRIPT**
        + **Integration with JavaScript**
        + **Generics (creating reusable components)**
    - **NODE**
      * **Asynchronous programming**
      * **Modules**
      * **File System**
      * **Express.js**
    - **REST API and APIs**
      * **HTTP methods**
      * **Status Codes**
      * **Endpoints**
      * **Authentication (API keys, OAuth)**
    - **MONGO DB/POSTGRES**
      * **CRUD operations**
      * **Data Modelling (MongoDB (document-based) vs. PostgreSQL (table-based))**
    - **ORM**
      * **Mapping Classes to Tables**
      * **Data Manipulation (CRUD operations)**
      * **Tools and Libraries (Sequelize, TypeORM, Mongoose)**
  1. **Advanced Level**

**Objective**: Master advanced programming concepts and practices.

**Core Topics**:

* + - **Advanced/Complete Case Study**
    - **STATE MANAGEMENT LIBRARY**
      * **Managing states**
      * **REDUX**
        + **Integrating REDUX with React**
    - **ADVANCED UI/UX**
      * **Common design patterns (Singleton, Factory, Observer)**
      * **Advanced layout and responsive design**
      * **Design thinking and user testing**
      * **Performance optimization for UI**
    - **MIDDLE WARE**
      * **Role of middleware**
      * **Express.js**
      * **Custom middleware for handling requests**
    - **QUEUE**
      * **Implementation and use cases**
      * **Tasks scheduling and background processing using queues**
      * **Libraries and tools (RabbitMQ, Amazon SQS, Redis)**
    - **CLOUD**
      * **Service models: IaaS, PaaS, SaaS**
      * **AWS**
      * **Benefits of using cloud infrastructure for application deployment**
    - **KUBERNETS**
      * **Architecture (pods, services, deployments)**
      * **Setting up and managing Kubernetes cluster**
      * **Best practices for deploying and scaling applications in Kubernetes**
    - **DOCKER/GITHUB ACTIONS**
      * **Architecture concepts** 
        + **architectural styles (monolithic, microservices, serverless)**
        + **Trade-offs involved in different architectural decisions**
      * **Microservices and their benefits**
        + **Communication between microservices (API gateways, service discovery)**
      * **Containerization and its benefits.**
    - **TESTING**
      * **Testing frameworks and tools (e.g., Jest, Mocha, Cypress)**
      * **Different types of testing (unit, integration, end-to-end).**
      * **Learn how to query AI written Test Cases.**
    - **DEPLOYMENT** 
      * **Best practices for application deployment.**
      * **Using CI/CD pipelines to automate the deployment process**
    - **MAINTENANCE** 
      * **Updating and scaling applications in response to user feedback and changes in requirements.**
      * **Importance of monitoring and logging for application performance.**

**Conclusion**

This structured learning path is designed to provide a comprehensive educational experience in programming. Each level builds upon the previous one, ensuring that learners acquire the necessary skills and knowledge to progress effectively. The curriculum is aimed at fostering not only technical skills but also a deeper understanding of how these skills apply in real-world scenarios.