**Smart Bridge Internship**

**Generative AI With IBM Cloud**

**Project Title:**

### EduTutor AI: Personalized Learning with Generative AI and LMS Integration

**Team Members:**

1. G. Ramitha Sri
2. P. Saivamsi
3. P. Hemasree
4. P. Rishi

**EduTutor AI: Personalized Learning with Generative AI and LMS Integration**

EduTutor AI is an AI-powered personalized learning platform designed to transform how students learn and how educators monitor progress. By integrating generative AI (IBM Watsonx, Granite LLM), Google Classroom, and real-time performance analytics, the platform delivers adaptive, engaging educational experiences. This document outlines the **six core development phases** that structure the creation of EduTutor AI.

**🔹 Phase 1: Brainstorming & Ideation**

**Objective:**

Generate the core idea and envision how AI and LMS can combine to improve education.

**Key Points:**

* **Identify the problem**: Recognize gaps in traditional learning—limited personalization, delayed feedback, and static content.
* **Ideate AI-powered features**: Brainstorm dynamic quizzes, adaptive difficulty levels, real-time student feedback, and performance dashboards.
* **Visualize integration**: Identify roles of IBM Watsonx for diagnostics and Granite LLM for personalized quiz generation.
* **Discuss LMS sync**: Choose Google Classroom as the base for syncing class materials and student data.

**Explanation:**

This phase brings together educators, developers, and AI experts to conceptualize a modern solution for digital learning. The aim is to combine AI-driven personalization with classroom-level alignment through LMS integration, building a smart ecosystem that adapts to individual learning needs and provides data-driven teaching support.

**🔹 Phase 2: Requirement Analysis**

**Objective:**

Define detailed system requirements from all stakeholders.

**Key Points:**

**Functional Requirements**:

* Student dashboard with personalized quizzes
* Educator dashboard with performance insights
* Real-time feedback on quiz responses

**Non-Functional Requirements**:

* Modular and scalable architecture
* Low-latency, real-time AI processing
* Secure API integration with Google Classroom

**Technology Stack**:

* Front-end: HTML, CSS, JavaScript
* Back-end: APIs to interface with Watsonx, Granite, Pinecone
* LMS: Google Classroom for data syncing

**Explanation:**

Here, the team collects insights from students, teachers, and admins to clearly define how EduTutor AI should function and behave. The goal is to build a high-performance, secure, and scalable web app that syncs with existing education systems and supports real-time AI-driven learning.

**🔹 Phase 3: Project Design**

**Objective:**

Design the architecture and user interface of the platform.

**Key Points:**

* **Modular UI/UX design**:
  + Student interface: login, dashboard, quiz area
  + Educator interface: performance analytics, student tracking
  + Quiz engine: adaptive test generator
* **Wireframes** for:
  + Student quiz interface
  + Educator performance dashboard
* **Integration plans**:
  + Watsonx (diagnostic testing)
  + Granite LLM (quiz generation)
  + Pinecone (progress data)
  + Google Classroom (class/course sync)
* **Responsive UI patterns** for accessibility and mobile usability

**Explanation:**

Designing the platform ensures that functionality and user experience are balanced. Interfaces must be intuitive for both students and educators, and backend systems must communicate efficiently with AI services. Mobile responsiveness, accessibility, and data security are key design priorities.

**🔹 Phase 4: Project Planning (Agile Methodologies)**

**Objective:**

Break the project into manageable sprints using Agile.

**Key Points:**

* **Sprint 1**: Google Classroom Integration & Course Sync
* **Sprint 2**: Student Dashboard + Quiz Generator (Granite LLM)
* **Sprint 3**: Educator Dashboard + Performance Insights (Pinecone)
* **Sprint 4**: Diagnostic Testing & Adaptive Quizzing (Watsonx)
* Agile processes:
  + Daily stand-ups
  + Sprint reviews and retrospectives
  + Backlog grooming

**Explanation:**

Agile methodology allows the team to deliver working software incrementally while adapting to new requirements. Each sprint targets a specific function, allowing developers to implement, test, and refine the platform in iterative cycles with frequent stakeholder feedback.

**🔹 Phase 5: Project Development**

**Objective:**

Implement the features using chosen technologies.

**Key Points:**

* **HTML**: Layout and structure for login forms, dashboards, quiz panels.
* **CSS**: Responsive and accessible design for all devices.
* **JavaScript**:
  + Handle API calls to sync Google Classroom data.
  + Communicate with Watsonx and Granite APIs for quiz and diagnostics.
  + Render quizzes dynamically, validate answers, and show instant feedback.
  + Store and manage student performance using front-end state logic.
* Develop reusable components for dashboards and quizzes.

**Explanation:**

This is the build phase where the platform’s core functionalities come to life. HTML structures the app, CSS styles it for clarity and usability, and JavaScript provides interactivity. The front-end is modular and scalable, with APIs handling AI interactions and LMS sync in real-time.

**🔹 Phase 6: Functional & Performance Testing**

**Objective:**

Ensure the application works correctly and performs well.

**Key Points:**

* **Unit Testing**: Validate individual JS components (e.g., quiz rendering, data sync).
* **Integration Testing**:
  + Sync with Google Classroom
  + Fetch and process AI-generated quizzes from Granite/Watsonx
* **Functional Testing**:
  + Student quiz-taking and feedback workflows
  + Educator performance analytics
* **Performance Testing**:
  + Simulate multiple users to ensure smooth operation
  + Optimize UI across devices and browsers
* Final bug fixing, UI optimization, and deployment readiness

**Explanation:**

In this final stage, both automated and manual tests are run to ensure that all systems function correctly under different scenarios. This includes user experience testing, performance under load, and compatibility across devices and browsers. Once stable, the product is prepared for deployment.