**Project Title:**

AI Personalized Email Generator

**Team Name:**

Invictus

**Team Members:**

• Tranush Reddy Bokka  
• Karthik Devanga  
• Alankrutha Talsani  
• Bhavya Vippalapalli

**Phase-1: Brainstorming & Ideation**

**Objective:**

The AI Personalized Email Generator is created to assist users in writing high-quality emails suited for various contexts, tones, and uses. Through the use of AI, the system produces formatted emails that improve communication effectiveness while conserving time.

**Key Points:**

**Problem Statement**

Composing emails is a time-consuming process, and most users find it difficult to get the tone and format right. This usually leads to ambiguous, unprofessional, or ineffective communication. It takes time and effort to write well-structured and effective emails, which can be a problem for people who write messages on a regular basis for professional or academic use.

**Proposed Solution**

The email generator with AI capability resolves this problem by accepting user input in terms of context, tone, and intention of the email and using AI to compose a well-written message. The solution enables users to easily draft professional and context-suitable emails quickly without sacrificing clarity and effectiveness.

**Target Users**

The tool is intended for professionals, business people, job seekers, and companies that often write emails and need to make them more effective. It serves those who need to write emails efficiently yet ensure that their communication is clear, professional, and well-structured.

**Expected Outcome**

The email generator based on AI ensures quicker mail composition, enhanced professionalism, better personalization, and less effort in writing emails for different contexts. The users can anticipate a better quality of communication, greater efficiency, and an easier mail-writing process.

**Phase-2: Requirement Analysis**

**Objective**

The requirement analysis is used to determine the necessary technical and functional requirements for developing and deploying the AI-driven email generator. It establishes key features, establishes technical and performance specifications, and makes sure that the system serves user needs effectively and efficiently. The phase also addresses assessing possible constraints and challenges to develop a properly structured development plan.

**Key Points**

**1. Technical Requirements:**

● Programming Language: Python  
● Frontend: Streamlit  
● Backend: Flask  
● AI Model: Google’s Gemini API  
● Database: No dedicated database (API-based approach)

**2. Functional Requirements:**

• Accept user inputs for email context, tone, and purpose.  
• Generate personalized emails using AI based on user inputs.  
• Provide a simple and intuitive interface for seamless user interaction.

**3. Constraints & Challenges:**

• Dependency on an external API (Google’s Gemini API).  
• Ensuring AI-generated content remains contextually accurate.  
• Handling API rate limits and potential response delays.

**Phase-3: Project Design**

**Objective:**

Define the system’s architecture, user flow, and UI/UX considerations to ensure a seamless user experience.

**Key Points:**

**1. System Architecture Diagram:**

The system consists of a Streamlit frontend that collects user input and communicates with a Flask backend, which processes the request using the Google Gemini API and returns the generated email.

**2. User Flow:**

The user enters the email context, tone, and purpose in the Streamlit interface. Upon clicking "Generate Email," the backend processes the request and returns the AI-generated email for display.

**3. UI/UX Considerations:**

The interface is clean, intuitive, and responsive, featuring structured input fields and error handling for smooth usability across devices.

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

**Objective:**

Break down tasks into sprints using Agile methodologies for efficient development.

**Key Points:**

**1. Sprint Planning:**

Sprint 1: Backend setup and Gemini API integration.  
Sprint 2: Frontend development using Streamlit and linking with the backend.  
Sprint 3: Testing, bug fixes, and performance improvements.  
Sprint 4: Final validation, deployment, and documentation.

**2. Task Allocation:**

Backend Developer: Flask API implementation and AI integration.  
Frontend Developer: UI design, API integration, and user experience improvements.

**3. Timeline & Milestones:**

Sprint 1: Backend setup and API integration.  
Sprint 2: Frontend implementation.  
Sprint 3: Testing and debugging.  
Sprint 4: Deployment and final documentation.

**Phase-5: Project Development**

**Objective:**

This phase involves coding and integrating all project components to build a fully functional AI email generator.

**Key Points:**

**1. Technology Stack Used:**

• Frontend: Streamlit  
• Backend: Flask  
• AI Model: Google Gemini API  
• Development Tools: Python, Requests, Flask-CORS

**2. Development Process:**

The project is developed using VS Code, with the Flask backend handling API requests and AI processing, while Streamlit provides a user-friendly interface for email generation. The AI model is integrated to process inputs and generate emails dynamically.

**3. Challenges & Fixes:**

• **Challenge:** API response delays.  
**Fix:** Optimized request handling and caching mechanisms.  
• **Challenge:** Inaccurate email tone adaptation.  
**Fix:** Improved prompt engineering for better contextual understanding.

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

**Objective:**

Ensure the AI email generator functions as expected and meets performance benchmarks.

**Key Points:**

**1. Test Cases Executed:**

Testing involves a range of email generation scenarios like job applications, follow-ups, and apologies as well as confirmation of different tone choices like formal, friendly, persuasive, and apologetic. Error handling is tested in the case of missing or invalid inputs.

**2. Bug Fixes & Improvements:**

• Fixed API response handling to reduce delays.  
• Improved UI responsiveness for a smoother user experience.

**3. Final Validation:**

The system successfully generates emails with accurate tones and relevant content based on user inputs.

**4. Deployment (if applicable):**

Hosted on a local Flask server, with the potential to expand to cloud platforms like AWS or Heroku.

**Final Submission**

The final submission includes the project report based on this template and GitHub repository containing the complete source code of the AI Email Generator.