**Hackathon Project Phases Template**

**Project Title:** AI Email Generation Assistant Using Groq and Flask

**Team Name:** Team XYZ

* **Team Members:**
* Harshith
* Mothilal Kumar
* Lehanth Madhav

### ****Phase-1: Brainstorming & Ideation****

**Objective:**  
Develop an AI-powered email assistant that generates professional, context-aware email responses using Groq and integrates seamlessly with Flask for a web-based frontend.

**Key Points:**

**Problem Statement:**

* Many professionals struggle with drafting emails efficiently while maintaining proper tone and clarity.
* Manually composing responses is time-consuming and prone to miscommunication.

**Proposed Solution:**

* A Flask-based AI application that uses Groq to generate email responses tailored to user input.

**Target Users:**

* Corporate employees looking for quick and professional email drafting.
* Customer support teams handling large volumes of emails.
* Freelancers and business owners who need well-structured communication.

**Expected Outcome:**

* A functional AI-powered email assistant that provides contextually relevant email drafts.

### ****Phase-2: Requirement Analysis****

**Objective:**  
Define the technical and functional requirements for the AI Email Assistant.

**Technical Requirements:**

* **Programming Language:** Python
* **Backend:** Flask, Groq API
* **Frontend:** HTML, CSS, JavaScript
* **Database:** Not required initially (stateless API-based processing)

**Functional Requirements:**

* Accept user inputs such as email subject, context, and preferred tone.
* Generate AI-powered email drafts using Groq.
* Allow users to edit AI-generated emails before finalizing.

**Constraints & Challenges:**

* Ensuring high-quality AI-generated responses with minimal hallucinations.
* Handling API rate limits efficiently.
* Providing a user-friendly UI for seamless interaction.

### ****Phase-3: Project Design****

**Objective:**  
Develop the architecture and user flow of the application.

**System Architecture:**

1. User enters email details in the frontend (subject, body, tone, etc.).
2. Flask backend sends input data to the Groq API.
3. Groq processes the data and returns an AI-generated email response.
4. The frontend displays the generated email for review and editing.

**User Flow:**

* Step 1: User inputs details (e.g., "Follow-up email for a job application").
* Step 2: Flask processes and sends the request to Groq.
* Step 3: AI generates an email draft and returns it to the UI.
* Step 4: User reviews and finalizes the email.

**UI/UX Considerations:**

* Minimalist, user-friendly interface for smooth navigation.
* Real-time text editing for AI-generated emails.
* Mobile-responsive design for accessibility.

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

| **Sprint** | **Task** | **Priority** | **Duration** | **Deadline** | **Assigned To** | **Dependencies** | **Expected Outcome** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Sprint 1 | Environment Setup & API Integration | 🔴 High | 6 hours | End of Day 1 | Shanawaz | Groq API Key | API connection established |
| Sprint 1 | Frontend UI Development | 🟡 Medium | 2 hours | End of Day 1 | Anwar | API response format finalized | Basic UI with input fields |
| Sprint 2 | Email Generation & Display | 🔴 High | 3 hours | Mid-Day 2 | Mohammad | API response, UI elements ready | AI-generated email drafts |
| Sprint 2 | Error Handling & Debugging | 🔴 High | 1.5 hours | Mid-Day 2 | Member 1 & 4 | API logs, UI inputs | Improved API stability |
| Sprint 3 | Testing & UI Enhancements | 🟡 Medium | 1.5 hours | Mid-Day 2 | Entire Team | API response, UI layout completed | Responsive UI, better UX |
| Sprint 3 | Final Presentation & Deployment | 🟢 Low | 1 hour | End of Day 2 | Entire Team | Working prototype | Demo-ready project |

### ****Phase-5: Project Development****

**Objective:**  
Implement core features of the AI Email Assistant.

**Technology Stack Used:**

* **Frontend:** HTML, CSS, JavaScript
* **Backend:** Flask, Groq API
* **Programming Language:** Python

**Development Process:**

* Implement API key authentication for Groq.
* Develop AI-powered email generation logic.
* Integrate UI elements for displaying and modifying emails.

**Challenges & Fixes:**

* **Challenge:** Ensuring AI-generated emails are contextually accurate.  
  **Fix:** Implement prompt tuning for better results.
* **Challenge:** API rate limits affecting performance.  
  **Fix:** Optimize API calls and cache frequently used prompts.

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

| **Test Case ID** | **Category** | **Test Scenario** | **Expected Outcome** | **Status** | **Tester** |
| --- | --- | --- | --- | --- | --- |
| TC-001 | Functional Testing | Generate a job interview follow-up email | Well-structured email draft | ✅ Passed | Shanawaz |
| TC-002 | Functional Testing | Generate a client proposal email | Contextually relevant draft | ✅ Passed | Anwar |
| TC-003 | Performance Testing | API response time under 500ms | Fast response time | ⚠ Needs Optimization | Tester 3 |
| TC-004 | Bug Fixes & Improvements | Fixed incorrect email tone | More accurate tone selection | ✅ Fixed | Developer |
| TC-005 | Final Validation | Ensure UI works across devices | Mobile & desktop compatibility | ❌ Failed - UI issues on mobile | Tester 2 |
| TC-006 | Deployment Testing | Host the app using Flask & Render | Accessible online | 🚀 Deployed | DevOps |

### ****Final Submission****

* **Project Report** (Based on this template)
* **Demo Video (3-5 Minutes)**
* **GitHub/Code Repository Link**
* **Presentation**

This updated document now reflects your AI email generation project without SendGrid. Let me know if you need further modifications!