**Hackathon Project Phases Template** for the **AI Study Planner** project.

Hackathon Project Phases Template

# Project Title: AI Personalized Email Generator.

# Team Name: Kanya Rashi

# Team Members:

* G. Nikhil
* B. Shiva Ram
* G. Saikrishna
* E. Akshitha
* A. Gayathri

# Phase-1: Brainstorming & Ideation

## Objective:

Create a custom email-sending application.

## Key Points:

1. **Problem Statement:**
   * AI Personalized Email Generator is an innovative project designed to revolutionize email communication by automating the process of drafting personalized emails.
   * This system uses state-of-the-art language models to generate tailored email content based on user inputs, such as the recipient's name, event details, and special instructions.
2. **Proposed Solution:**
   * The AI Personalized Email Generator streamlines the email drafting process by allowing users to input specific details such as the recipient's name, event name, and special instructions.
   * It then generates a customized email body, saving users time and effort in composing individualized messages.
   * This feature is particularly beneficial in professional settings where personalized communication is crucial for building and maintaining relationships.
3. **Target Users:**
   * WORKING PROFESSIONALS
   * STUDENTS
4. **Expected Outcome:**
   * The AI Personalized Email Generator generates a customized email body, saving users time and effort in composing individualized messages.

# Phase-2: Requirement Analysis

## Objective:

Define the technical and functional requirements for the AI Personalized Email Generator.

## Key Points:

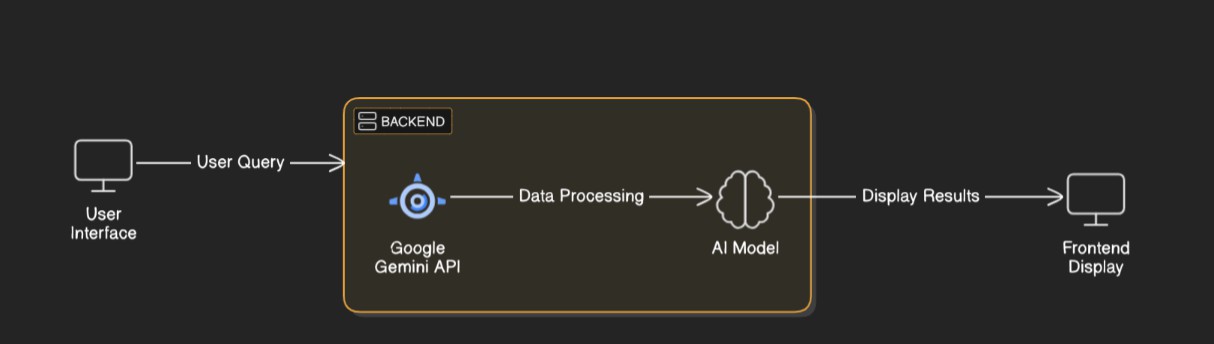
1. **Technical Requirements:**
   * Programming Language: typescript
   * Backend: ESP integration, LLM API’s
   * Frontend: React, tailwind css
   * Database: postgreSQL
2. **Functional Requirements:**
   * Accepts a customizable prompt for personalizing each email.
   * Customizes and sends emails using an LLM or other content-generation approach.
   * Allows users to connect their email account.
   * Generated content saved to Database.

1. **Constraints & Challenges:**
   * Email Integration.
   * Email Customization and Sending.
   * Column Detection and Dynamic Field Replacement.

# Phase-3: Project Design

## Objective:

Develop the architecture and user flow of the application.



## Key Points:

1. **System Architecture:**
   * User enters input specific details such as the recipient's name, event name, and special instructions via UI.
   * Input is processed using **Google Gemini API**.
   * AI model fetches and processes the data.
   * The frontend displays customized email.
2. **User Flow:**
   * Step 1: User enters a input(e.g., recipient's name, event name).
   * Step 2: The backend **calls the Gemini Flash API** to fetch the content.
   * Step 3: The processes the data and **displays results.**
3. **UI/UX Considerations:**
   * **user-friendly interface.**

# Phase-4: Project Planning (Agile Methodologies)

## Objective:q2

Break down development tasks for efficient completion.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Task** | **Priority** | **Duration** | **Deadline** | **Assigned To** | **Dependencies** | **Expected Outcome** |
| Sprint 1 | Environment Setup | 🔴 High | 6 hours  (Day 1) | End of Day 1 | Member 1 | React, node, tailwind | API connection established & working |
| Sprint 1 | Frontend UI Development | 🟡  Medium | 2 hours  (Day 1) | End of Day 1 | Member 2 | API response format finalized | Basic UI with input fields |
| Sprint 2 | API Integration | 🔴 High | 3 hours  (Day 2) | Mid-Day 2 | Member 1& 2 | API response, UI elements ready | Search functionality with filters |
| Sprint 2 | Error Handling & Debugging | 🔴 High | 1.5 hours  (Day 2) | Mid-Day 2 | Member 1&4 | API logs, UI inputs | Improved API stability |
| Sprint 3 | Testing & UI Enhancements | 🟡  Medium | 1.5 hours  (Day 2) | Mid-Day 2 | Member 2& 3 | API response, UI layout completed | Responsive UI, better user experience |
| Sprint 3 | Final Presentation & Deployment | 🟢 Low | 1 hour  (Day 2) | End of Day 2 | Entire Team | Working prototype | Demo-ready project |

## Sprint Planning with Priorities

**Sprint 1 – Setup & Integration (Day 1)**

**(**🔴 **High Priority)** Set up the **environment** & install dependencies.

**(**🔴 **High Priority)** Integrate **Google Gemini API**.

**(**🟡 **Medium Priority)** Build a **basic UI with input fields**.

## Sprint 2 – Core Features & Debugging (Day 2)

**(**🔴 **High Priority)** Implement **search & comparison functionalities**. **(**🔴 **High Priority)** Debug API issues & handle **errors in queries**.

## Sprint 3 – Testing, Enhancements & Submission (Day 2)

**(**🟡 **Medium Priority)** Test API responses, refine UI, & fix UI bugs.

**(**🟢 **Low Priority)** Final **demo preparation & deployment**.

# Phase-5: Project Development

## Objective:

Implement core features of the Email-Editor App.

## Key Points:

1. **Technology Stack Used:**
   * **Frontend:** React, tailwind css
   * **Backend:** ESP integration, LLM API’s
   * **Programming Language:** typescript
2. **Development Process:**
   * Implement **API key authentication** and **Gemini API integration**.
   * Optimize **search queries for performance and relevance**.
3. **Challenges & Fixes:**
   * **Challenge:** Delayed API response times.

**Fix:** Implement **caching** to store frequently queried results.

* + **Challenge:** Limited API calls per minute.

**Fix:** Optimize queries to fetch **only necessary data**.

# Phase-6: Functional & Performance Testing

## Objective:

Ensure that the Email-Editor App works as expected.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Category** | **Test Scenario** | **Expected Outcome** | **Status** | **Tester** |
| TC-001 | Functional Testing | Environment Setup and UI build | Setup done and basic UI completed | ✅ Passed | Tester 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TC-002 | Performance Testing | API response time under 500ms | API should return results quickly. | ⚠ Needs Optimization | Tester 3 |
| TC-003 | Bug Fixes & Improvements | Fixed incorrect API responses. | Data accuracy should be improved. | Semi-Fixed | Developer |
| TC-004 | Final Validation | Ensure UI is responsive across devices. | UI should work on mobile & desktop. | ❌ Failed - UI broken on mobile | Tester 2 |
| TC-005 | Deployment Testing |  | App should be accessible online. |  | Netlify |

# Final Submission

1. **Project Report Based on the templates**
2. **Demo Video (3-5 Minutes)**
3. **GitHub/Code Repository Link**
4. **Presentation**