**Hackathon Project Phases Template** for the **Flavour Fusion** project.

# Hackathon Project Phases Template

**Project Title:**

**Flavour Fusion - AI-Powered Recipe Generator & Blog Hub**

**Team Name:**

**Team AI Titans**

**Team Members:**

* K.Charandeep Reddy
* M.Yashwanth Goud
* M.Srikar
* Mohammed Abdul Nazeeb
* P.Akhil Reddy

## Phase-1: Brainstorming & Ideation

**Objective:**

Develop an AI-powered recipe generator that provides users with customized recipes based on their dietary needs, meal type (breakfast, main course, dessert, etc.), and available ingredients. Additionally, users can share their culinary experiences by posting their own recipe blogs on the platform.

**Key Points:**

1. **Problem Statement:**

* Many individuals struggle with deciding what to cook based on available ingredients.
* Users often seek recipes that align with specific dietary preferences (e.g., vegan, keto, gluten-free).
* There is a lack of personalized meal planning tools that dynamically generate recipes based on user needs.
* Food bloggers and home cooks need a platform to share their unique recipes and experiences.

1. **Proposed Solution:**

* An AI-powered web application that generates recipes based on user input.
* Utilizes the **Gemini API** to provide dynamic, unique, and creative recipe suggestions.
* Offers advanced filters for **meal type, cuisine, dietary preferences, and available ingredients** to enhance personalization.
* Includes a **recipe blog feature**, allowing users to share their own culinary creations and experiences.

1. **Target Users:**

* **Home cooks** looking for new meal ideas and inspiration.
* **Health-conscious individuals** with specific dietary restrictions.
* **Busy individuals** who need quick meal suggestions based on what they have at home.
* **Food bloggers and enthusiasts** who want to share and explore recipes within a community.
* **People aiming to reduce food waste** by maximizing the use of available ingredients.

1. **Expected Outcome:**

○ A fully functional **AI-powered recipe generator** with smart filtering options.  
 ○ A **user-friendly interface** that simplifies meal customization  
 ○ A **community-driven recipe blog** section for sharing and discovering recipes.

## Phase-2: Requirement Analysis

**Objective:**   
Define the technical and functional requirements for Flavour Fusion.

**Key Points:**

1. **Technical Requirements:**

* Project Language: JavaScript

○ Backend: Express.js with integration to the Gemini API.

○ Frontend: React with TypeScript and Tailwind CSS

○ Database: MongoDB.

1. **Functional Requirements:**

* Allow users to input meal preferences and ingredients.
* Fetch and display AI-generated recipes from the Gemini API.
* Offer filters for cuisine, dietary restrictions, and difficulty level.
* Provide step-by-step cooking guides with ingredient quantities.

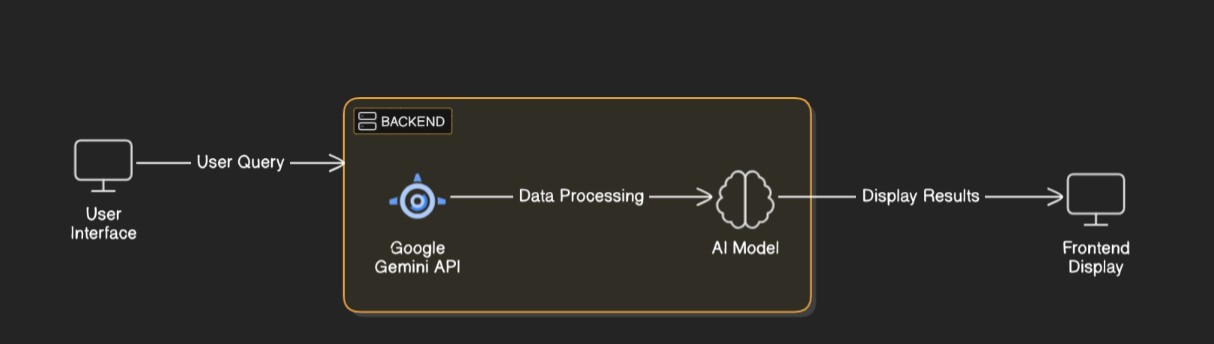
1. **Constraints & Challenges:**

○ Ensuring accurate and diverse recipe generation.  
 ○ Handling API limits while maintaining real-time responsiveness.  
 ○ Providing a seamless user experience across devices.

## Phase-3: Project Design

**Objective:**

Develop the architecture and user flow of the application.



**Key Points:**

1. **System Architecture:**

* 1. User enters Recipe-related query via UI.

○ Query is processed using **Google Gemini API**.

○ AI model fetches and processes the data.

○ The frontend displays **Recipe details**.

1. **User Flow:**

○ **Step 1:** User inputs meal type, available ingredients, and dietary preferences.  
 ○ **Step 2:** Frontend sends user input to the backend.  
 ○ **Step 3:** Backend processes input and requests a recipe from the Gemini API.  
 ○ **Step 4:** Gemini API returns a recipe to the backend.  
 ○ **Step 5:** Backend processes the recipe data and sends it to the frontend.  
 ○ **Step 6:** Frontend displays the recipe with cooking instructions to the user.

1. **UI/UX Considerations:**

○ Clean, user-friendly interface with intuitive filters.  
 ○ Responsive design to ensure compatibility across devices.  
 ○ Options to save favourite recipes and adjust serving sizes.

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

**Objective:**

Break down development tasks for efficient completion.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Task** | **Priority** | **Duration** | **Deadline** | **Assigned To** | **Dependencies** | **Expected**  **Outcome** |
| Sprint 1 | Environment Setup  & API Integration | 🔴 High | 6 hours  (Day 1) | End of Day  1 | Yashwanth | Google Gemini API, Backend setup | API connection established & working |
| Sprint 1 | Frontend UI Development | 🟡  Medium | 2 hours  (Day 1) | End of Day  1 | Srikar | API response format finalized | Basic UI with input fields |
| Sprint 2 | Recipe Generation Logic | 🔴 High | 3 hours  (Day 2) | Mid-Day 2 | Charandeep | API response, backend functions | Search functionality with filters |
| Sprint 2 | Error Handling &  Debugging | 🔴 High | 1.5 hours  (Day 2) | Mid-Day 2 | Charandeep | API logs, UI inputs | Improved API stability |
| Sprint 3 | Testing & UI  Enhancements | 🟡  Medium | 1.5 hours  (Day 2) | Mid-Day 2 | Akhil and Nazeeb | 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 development environment and install dependencies.  
 ○ Initialize the React project with TypeScript and Tailwind CSS.  
 ○ Integrate the Gemini API into the backend.

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

🔴 **High Priority**  
 ○ Implement user input forms and filters on the frontend.  
 ○ Develop backend logic to process user input and fetch recipes from the Gemini API.  
 ○ Ensure proper data flow between frontend and backend.  
 ○ Debug any issues related to API integration and data handling.

**Sprint 3 – Testing, Enhancements & Submission (Day 3)**

🟡 **Medium Priority**  
 ○ Conduct functional and performance testing.  
 ○ Refine UI components for better user experience.  
 ○ Implement additional features such as saving favourite recipes.  
 ○ Prepare final documentation and deployment.

## Phase-5: Project Development

**Objective:**

Implement core features of Flavour Fusion.

**Key Points:**

1. **Technology Stack Used:**

○ **Frontend:** React with TypeScript and Tailwind CSS.  
 ○ **Backend:** Express.js with integration to the Gemini API and MongoDB.  
 ○ **API:** Gemini API for recipe generation**.**

1. **Development Process:**

○ Set up the frontend project structure with React and Tailwind CSS.  
 ○ Develop user input forms and filters for meal customization.  
 ○ Implement backend services to handle API requests and process data.  
 ○ Integrate the Gemini API to fetch AI-generated recipes.  
 ○ Ensure seamless communication between frontend and backend.

1. **Challenges & Fixes:**

○ **Challenge:** Managing state and data flow in a complex frontend.  
 **Fix:** Utilize state management libraries such as Redux or Context API.  
 ○ **Challenge:** Handling asynchronous API calls and potential errors.  
 **Fix:** Implement proper error handling and loading states.

## Phase-6: Functional & Performance Testing

**Objective:**

Ensure that the Fusion Flavour App works as expected.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Category** | **Test Scenario** | **Expected Outcome** | **Status** | **Tester** |
| TC-001 | Functional  Testing | Query "Vegan pasta recipe" | Display relevant vegan pasta recipe | ✅ Passed | Yashwanth |
| TC-002 | Functional  Testing | Query "Dessert with chocolate" | Suggest a chocolate dessert | ✅ Passed | Akhil |
| TC-003 | Performance  Testing | API response time under 500ms | Quick API response | ✅ Passed | Srikar |
| TC-004 | Bug Fixes & Improvements | Fixed incorrect API responses. | All bugs are fixed. | ✅ Fixed | Charandeep |
| TC-005 | Final Validation | Ensure UI is responsive across devices. | UI is working on desktop and mobile. | ✅ Passed | Nazeeb |
| TC-006 | Deployment  Testing | Host the app using  Vercel Sharing | App should be accessible online. | 🚀 Deployed | DevOps |

**Final Submission:**

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