

Hackathon Project Phases Template

Project Title:

Blog Generation Using Gemini Ai Flash 2.0 and Streamlit

Team Name:

Deep coders

Team Members:

- Sai Pradyumna (Team Leader)
 - G. Paramesh Baba
 - G. Sampath
-

Phase-1: Brainstorming & Ideation

Objective:

The objective is to develop an **AI-powered blog generation** tool using the **Gemini Ai Flash 2.0** model that enables users to create blog content based on their input. Built with **Streamlit**, this application provides an interactive web interface where users can specify the topic, word count, and target audience. The model generates coherent and contextually relevant blog posts tailored to the specified parameters. This tool aims to assist content creators, researchers, and professionals in quickly producing high-quality written content.

Key Points:

1.Problem Statement:

Users often struggle with creating high-quality blog content due to time constraints and a lack of writing expertise. Generating engaging and well-structured articles can be challenging, especially for non-writers. There is a need for an AI-powered solution that simplifies and accelerates content generation while ensuring relevance and customization.

2. Proposed Solution:

Develop an AI-powered blog generation tool using the Gemini Ai Flash 2.0 model, integrated with Streamlit for an interactive user experience. This tool will allow users to specify the topic,

word count, and target audience, enabling the model to generate coherent and contextually relevant blog posts. By automating content creation, the solution will help users quickly produce high-quality articles, reducing effort and enhancing productivity.

3. Target Users:

The tool will serve content creators, bloggers, and digital marketers who need to generate high-quality articles efficiently. It will also benefit researchers and professionals looking for quick, well-structured written content for reports, publications, or knowledge sharing. Additionally, businesses and individuals seeking to streamline content creation without advanced writing skills will find this tool valuable.

4. Expected Outcome:

The outcome will be an AI-powered tool that generates high-quality, contextually relevant blog content based on user input. This will streamline content creation, saving time and effort while ensuring customization and coherence.

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for building Blog generation, an AI-powered tool that leverages Gemini Ai Flash 2.0 to enable coherent and contextually relevant blog posts and provide real-time insights.

Key Points:

1. Technical Requirements:

- **Programming Language:** Python will be used for backend development, ensuring compatibility with the Gemini Ai Flash 2.0 model and content generation APIs.
- **Backend:** The backend will integrate the Gemini Ai Flash 2.0 model to process user inputs, generate high-quality blog content, and manage customization features.
- **Frontend:** The Streamlit web framework will be used to create an interactive, user-friendly interface for specifying blog topics, word count, and target audience.
- **Database:** MongoDB (NoSQL database) will be used to store generated blog drafts, user preferences, and interaction history for future reference.

2. Functional Requirements:

- **Blog Content Generation:** Users should be able to input a topic, word count, and target audience to generate high-quality, structured blog content.
- **Real-Time Generation:** The tool will provide instant content creation, ensuring users receive coherent and contextually relevant blog posts without delays.
- **Customization & Refinement:** Users can adjust tone, style, and keywords to personalize the generated content according to their preferences.
- **Editing & Exporting:** The system will allow users to edit generated content and export blogs in various formats, such as text or PDF.
- **Search & Content History:** Users will have the ability to search previously generated blogs and access a history of content for future reference.

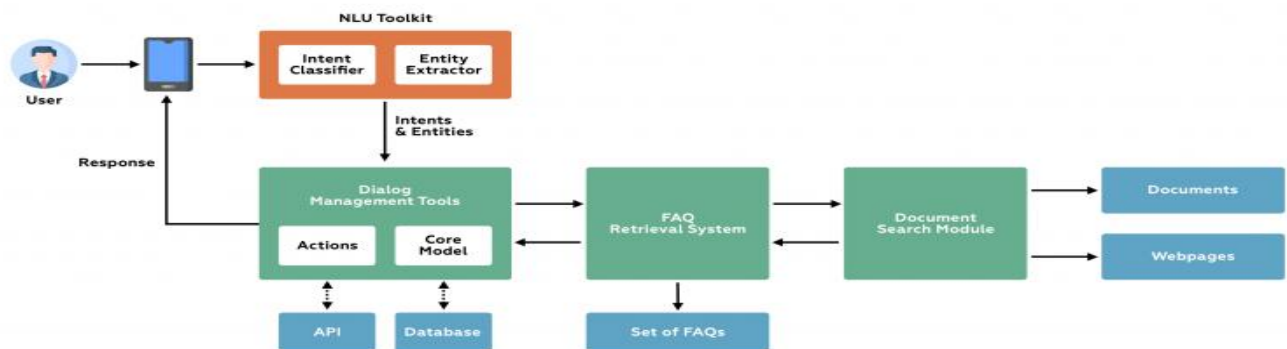
3.Constraints & Challenges:

- **Model Response Time:** Ensuring that the Gemini Ai flash 2.0 model generates high-quality blog content quickly while maintaining coherence and relevance.
- **Content Quality & Accuracy:** Maintaining contextual accuracy and avoiding biased or incorrect information in generated blog posts.
- **Customization Limitations:** Balancing user customization options (tone, style, keywords) without compromising content fluency and readability.
- **Streamlit UI Optimization:** Ensuring the Streamlit interface remains responsive and user-friendly while handling multiple user requests efficiently.
- **Scalability & Resource Management:** Managing computational resources effectively to support multiple content generation requests without performance degradation.

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.



Key Points:

1. System Architecture:

User inputs generate blog posts via the UI.

The backend processes the blogs using **Gemini Ai Flash 2.0API** to retrieve and blog data.

Results (user input related output text, audio, image) are displayed in the frontend.

2. User Flow:

Step 1: User enters an input (e.g., "give me an article on Pulwama attack").

Step 2: The backend uses **Gemini Ai Flash 2.0 API** to fetch data.

Step 3: **Blog posts** is displayed in a user-friendly format

3.UI/UX Considerations:

Simple and Intuitive UI: A clean, minimalistic interface using Streamlit for seamless blog generation and navigation.







Customization Options: Users can adjust tone, style, and keywords to refine blog content according to their needs.

Dark and Light Mode: Theme options to enhance user comfort and accessibility.

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	Member 1	Gemini ai flash 2.0, python , Streamlit, transformers	API connection established & working
Sprint 1	Frontend UI Development	 Medium	2 hours (Day 1)	End of Day 1	Member 2	Install & running Streamlit ready	Interactive & structured blog
Sprint 2	Vehicle Search & Comparison	 High	3 hours (Day 2)	Mid-Day 2	Member 1 & 2	API response, Processed user inputs	Extracted insights from inputs
Sprint 2	Error Handling & Debugging	 High	1.5 hours (Day 2)	Mid-Day 2	Member 1 & 3	API logs, Blog outputs	Improved API stability
Sprint 3	Testing & UI Enhancements	 Medium	1.5 hours (Day 2)	Mid-Day 2	Member 2 & 3	blog results, output as per user	Generates coherent and contextually relevant blog posts
Sprint 3	Final Presentation & Deployment	 Low	1 hour (Day 2)	End of Day 2	Entire Team	Working prototype	Demo-ready project & final presentation

Sprint Planning with Priorities



Sprint 1 – Setup & Integration (Day 1)

-  **High Priority:** Set up the environment & install dependencies (Python, Streamlit, Gemini Ai Flash 2.0 API).
-  **High Priority:** Integrate Gemini Ai Flash 2.0 API key for generating blog posts.
-  **Medium Priority:** Perform initial tasks of text generation as per user input

Sprint 2 – Core Features & Debugging (Day 1)

-  **High Priority:** Implement blog execution & analysis using Gemini Ai Flash 2.0 API in blog generation.
-  **High Priority:** Debug API issues & handle errors in blog generation.

Sprint 3 – Testing, Enhancements & Submission (Day 1)

-  **Medium Priority:** Test API responses, refine generating processing, & optimize blog.
-  **Low Priority:** Prepare final data visualizations, generate insights, & deploy the project for presentation.

Phase-5: Project Development

Objective:

Implement the core features of Generating Blogs including natural language query processing, data analysis, and real-time insights using Gemini Ai Flash 2.0 API.

Key Points:

1. Technology Stack Used:

- **Frontend:** Streamlit framework to create an intuitive and interactive web interface for blog generation.
- **Backend:** Gemini ai flash 2.0 model to process user inputs and generate high-quality, contextually relevant blog content.
- **Programming Language:** Python, for backend development, AI model integration, and handling user requests.

2. Development Process:

- **Model Integration:** Implement and fine-tune the Gemini ai flash 2.0 model to generate high-quality, contextually relevant blog content based on user inputs.
- **User Input Processing:** Develop logic to handle user inputs such as topic, word count, and target audience, ensuring precise content generation.
- **Content Optimization:** Enhance blog coherence, readability, and structure by refining text generation algorithms and customization features.
- **Editing & Exporting:** Implement editing tools for users to modify content and enable export options in formats like text and PDF.
- **Search & History Management:** Allow users to search, retrieve, and manage previously generated blog content for future use.
- **Deployment & Security:** Ensure secure deployment on cloud platforms with authentication and data protection mechanisms.

3. Challenges & Fixes:

- **Challenge:** Ensuring high-quality and contextually relevant blog content.
 - Fix: Fine-tune the Gemini ai flash 2.0 model with prompt optimization and content validation techniques.
- **Challenge:** Managing response time and handling multiple user requests efficiently.
 - Fix: Optimize backend processing, implement caching, and deploy on scalable cloud infrastructure.
- **Challenge:** Maintaining UI responsiveness while generating long-form content.
 - Fix: Use asynchronous processing and optimize Streamlit UI updates for a smooth user experience.

Phase-6: Functional & Performance Testing

Objective:

Ensure that the Blog Generation App works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Generate a blog post on "AI Trends in 2025"	The system should produce a coherent, relevant blog post.	✅ Passed	Tester 1
TC-002	Functional Testing	Specify a word count of 300 words for the generated blog	The generated blog should be close to or exactly 300 words.	✅ Passed	Tester 2
TC-003	Performance Testing	API response time under 1 second	The model should return generated content quickly.	⚠️ Needs Optimization	Tester 3
TC-004	Bug Fixes & Improvements	Address repeated content in multiple paragraph	The generated text should be unique and free of repetition	✅ Fixed	Developer
TC-005	Final Validation	Ensure UI is responsive on mobile and desktop	The Streamlit interface should function properly on all devices.	❌ Failed - UI issues on mobile	Tester 2
TC-006	Deployment Testing	Host the app using Streamlit Sharing	The app should be accessible online for public use.	🚀 Deployed	DevOps

Final Submission

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