
Hackathon Project Phases Template

Project Title:

Blog Generation Using LLAMA 2 and Streamlit

Team Name:

Steel Birds

Team Members:

- Neena Bhanu
- Neha
- Nishitha
- Sri Vidya

Phase-1: Brainstorming & Ideation

Objective:

Develop an AI-powered blog generation tool using LLaMA 2 to assist content creators, researchers, and professionals in quickly generating high-quality written content.

Key Points:

1. **Problem Statement:**
    - Content creation is time-consuming and requires extensive research.
    - Researchers and professionals often need structured blog posts tailored to specific audiences.

2. **Proposed Solution:**
    - A Streamlit-based web application powered by LLaMA 2.
    - Allows users to input topic, word count, and target audience to generate customized blogs.
3. **Target Users:**
    - Content creators who need automated blog drafts.
    - Researchers seeking structured and technical blog content.
    - Professionals require quick article generation for documentation.
4. **Expected Outcome:**
    - A functional application that generates structured, relevant, and high-quality blog posts based on user input.

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for the LLaMA 2-powered blog generation tool.

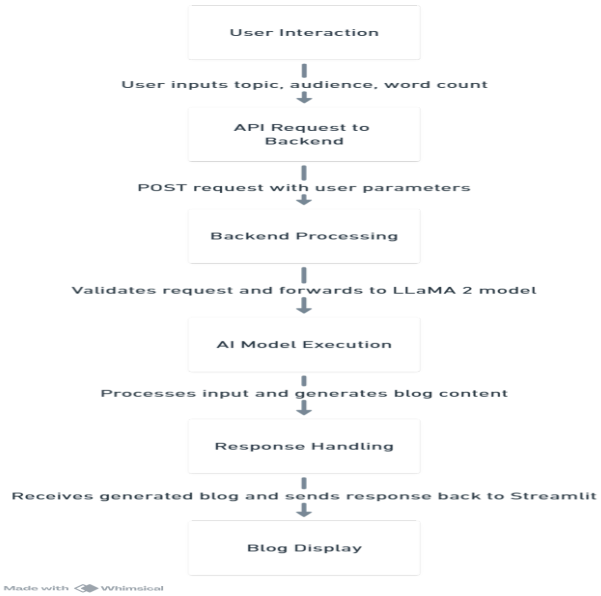
Key Points:

1. **Technical Requirements:**
    - **Programming Language:** Python
    - **Backend:** LLaMA 2 Model
    - **Frontend:** Streamlit Web Framework
    - **Database:** Not required initially
2. **Functional Requirements:**
    - Accept user input for topic, word count, and audience.
    - Generate blog content using LLaMA 2.
    - Display results in a structured format.
    - Option to export generated blogs as a PDF.
3. **Constraints & Challenges:**
    - Ensuring relevant and non-repetitive content generation.
    - Handling large API requests and optimizing response time.
    - Providing a user-friendly UI for seamless interaction.

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.



Key Points:

1. **System Architecture:**
   - User inputs topic, word count, and target audience.
   - LLaMA 2 processes the input and generates blog content.
   - The output is displayed on the frontend with an option to export as a PDF.
2. **User Flow:**
   - Step 1: User enters blog requirements.
   - Step 2: Application processes input using LLaMA 2.
   - Step 3: Generated blog is displayed for review and export.
3. **UI/UX Considerations:**
   - Minimalist design with clear input fields.
   - Readable blog formatting for better user experience.
   - Export functionality for convenience.

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	Sri Vidya	LLaMA 2 API, Python, Streamlit	API connection established & working
Sprint 1	Frontend UI Development	🟡 Medium	2 hours (Day 1)	End of Day 1	Nishitha	API response format finalized	Basic UI with input fields
Sprint 2	Blog Generation & Refinement	🔴 High	3 hours (Day 2)	Mid-Day 2	Neena	API response, UI elements ready	AI-generated blogs with structured content
Sprint 2	Error Handling & Debugging	🔴 High	1.5 hours (Day 2)	Mid-Day 2	Neha	API logs, UI inputs	Improved API stability
Sprint 3	Testing & UI Enhancements	🟡 Medium	1.5 hours (Day 2)	Mid-Day 2	Neena &Srividya	API response, UI layout completed	Responsive UI, better user experience
Sprint 3	Final Presentation & Deployment	🟢 Low	1 hour (Day 2)	End of Day 2	Neha & Nishitha	Working prototype	Demo-ready project

---

Phase-5: Project Development

Objective:

Implement core features of the blog generation tool.

Key Points:

1. **Technology Stack Used:**
    - Frontend: Streamlit
    - Backend: LLaMA 2
    - Programming Language: Python
2. **Development Process:**
    - Implement API key authentication and model integration.
    - Develop content generation logic with adjustable word limits.
    - Optimize blog structuring for coherence and readability.
3. **Challenges & Fixes:**
    - **Challenge:** Ensuring contextually accurate content.
        - Fix: Fine-tune prompts and filtering mechanisms.
    - **Challenge:** High response times for long blogs.
        - Fix: Implement efficient API request handling and caching.

Phase-6: Functional & Performance Testing

Objective:

Ensure that the LLaMA 2-powered blog generation tool works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Generate a 500-word blog on AI for researchers	Relevant and structured blog content	✅ Passed	Tester 1
TC-002	Functional Testing	Generate a 1000-word blog for general audience	Well-structured and readable blog	✅ Passed	Tester 2
TC-003	Performance Testing	API response time under 1s for 500-word blogs	Fast blog generation	⚠ Needs Optimization	Tester 3
TC-004	Bug Fixes & Improvements	Fixed repetitive content generation	Blog uniqueness improved	✅ Fixed	Developer
TC-005	Final	Ensure UI is	UI works	❌ Failed - UI	Tester 2

	Validation	responsive on all devices	smoothly	broken on mobile	
TC-006	Deployment Testing	Host the app using Streamlit Sharing	App is accessible online	🚀 Deployed	DevOps