Hackathon Project Phases Template for the **AI BLOG GENERATOR ASSISTANT** project.

Hackathon Project Phases Template

Project Title:

Blog Generation Using LLaMA 2 and Streamlit

Team Name:

Team EXTRINOS

Team Members:

- Yanala Sanjay Reddy
- Banala Varun Reddy
- R Venkata Sai Raghuram

Phase-1: Brainstorming & Ideation

Objective:

Develop an AI-powered blog generator using LLaMA 2 and Streamlit to assist content creators, researchers, and businesses in generating high-quality blog posts efficiently.

Key Points:

1. **Problem Statement:**
    - Content creators often struggle with writer's block and time constraints.
    - Businesses and researchers need high-quality, structured blog posts tailored to specific audiences.
    - Existing AI-generated content lacks accuracy, coherence, and audience-specific customization.
2. **Proposed Solution:**

- A web-based application utilizing LLaMA 2 for generating structured blog posts based on user input.
- Users can specify the topic, word count, and target audience.
- Optionally integrates SterlINT for content validation and refinement.

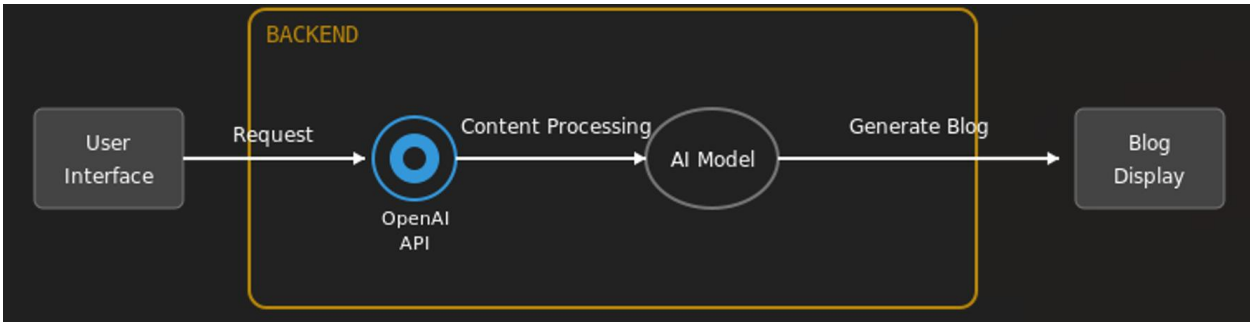
3. **Target Users:**
   - Bloggers and content creators.
   - Digital marketers and SEO specialists.
   - Researchers and educators.
   - Businesses needing automated content generation

4. **Expected Outcome:**
   - A functional AI-powered blog generator that delivers high-quality, customizable, and audience-specific blog posts.

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for the Blog Generator App.



Key Points:

1. **Technical Requirements:**

   - Programming Language: **Python**
   - Backend: **FastAPI**
   - Frontend: **Streamlit Web Framework**
   - Database: **Optional (for saving user-generated content)**
   - Model: **LLaMA 2 (Meta's Large Language Model)**
   - **Optional: SterlINT for content validation**

2. **Functional Requirements:**

- Generate high-quality blogs based on user input.
- Customize tone, style, and target audience.
- Display generated content in an easy-to-read format.
- Validate and refine content (if SterlINT is integrated)..

3. **Constraints & Challenges:**
   - Ensuring coherence and factual accuracy in generated content.
   - Handling API rate limits and optimizing LLaMA 2 responses.
   - Providing a seamless user experience via Streamlit.

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.

Key Points:

1. **System Architecture:**
   - User inputs topic, audience, and word count.
   - LLaMA 2 processes the request and generates content.
   - (Optional) SterlINT validates the generated text.
   - Streamlit displays the blog post.

2. **User Flow:**
   - Step 1: User enters blog topic and preferences.
   - Step 2: Backend processes the request using LLaMA 2.
   - Step 3: AI-generated blog post is displayed in Streamlit UI.
   - Step 4: User can copy or save the generated content.

3. **UI/UX Considerations:**
   - **Clean, user-friendly interface.**
   - **Adjustable settings for customization.**
   - **Dark & light mode for enhanced readability.**.

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	Raghuram	LLaMA 2 Model Setup	AI Model Ready
Sprint 1	Frontend UI Development	Medium	2 hours (Day 1)	End of Day 1	Varun	Model API Defined	Basic UI Ready
Sprint 2	Blog Content Generation	High	3 hours (Day 2)	Mid-Day 2	Sanjay	Backend API	Content Generation Working
Sprint 2	Content Validation (SterlINT)	High	1.5 hours (Day 2)	Mid-Day 2	Varun	AI Model Response	Quality Validation Implemented
Sprint 3	Testing & UI Enhancements	Medium	1.5 hours (Day 2)	Mid-Day 2	Sanjay & Raghuram	Functional Features Ready	UI Improved
Sprint 3	Final Presentation & Deployment	Low	1 hour (Day 2)	End of Day 2	Entire Team	Working prototype	Demo-ready project

Sprint Planning with PrioritiesSprint 1 – Setup & Integration (Day 1)

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

(High Priority) Integrate LLaMA 2 API.

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

Sprint 2 – Core Features & Debugging (Day 2)

(High Priority) Implement blog content generation.

(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 Blog Generator.

Key Points:

1. **Technology Stack Used:**
    - **Frontend:** Streamlit
    - **Backend:** FastAPI
    - **Programming Language:** Python
    - **AI Model**: LLaMA 2
    - **Optional**: SterlINT for validation
2. **Development Process:**
    - Implement API for LLaMA 2 integration.
    - Develop content generation logic.
    - Optimize UI/UX for an interactive experience.
3. **Challenges & Fixes:**
    - **Challenge: Slow AI response times.**
    - **Fix**: Implement caching and optimized prompts**.**
    - **Challenge: Ensuring quality in generated blogs.**
    - **Fix:** Use fine-tuning techniques and validation tools.

Phase-6: Functional & Performance Testing

Objective:

Ensure that the AutoSage App works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Generate a blog on "AI Trends"	AI-generated content appears	✅ Passed	Raghuram
TC-002	Functional Testing	Generate content for "Researchers"	Technical blog is generated	✅ Passed	Sanjay

TC-003	Performance Testing	Response time under 500ms	API should return results quickly.	⚠ Needs Optimization	Sanjay
TC-004	UI Testing	Ensure UI is user-friendly	Smooth navigation	✅ Fixed	Varun
TC-005	Final Validation	Mobile responsiveness	UI should work on mobile & desktop.	❌ Failed - UI broken on mobile	Raghuram
TC-006	Deployment Testing	Host the app using Streamlit Sharing	App should be accessible online.	Deployed	DevOps

---

Final Submission

1. **Project Report Based on the templates**
2. **GitHub/Code Repository Link**
3. **Presentation.**