Hackathon Project Phases Template for the **ProVisionAI** project.

Hackathon Project Phases Template

Project Title:

ProVisionAI: Unleashing the Power of Gemini Vision for Image Annotation

Team Name:

NEX AI

Team Members:

- Chakilela Srividhya
- Fariha Naaz
- Aramati Poojani Reddy

Phase-1: Brainstorming & Ideation

Objective:

Develop an AI-powered Image Captioning & Insight Generation Tool* that extracts meaningful insights from images and generates descriptive captions using BLIP and Google Gemini AI.

Key Points:

1. **Problem Statement:**

- Users need automated insights and captions for images.
- Manual image analysis is time-consuming and lacks accuracy.
- Existing tools don’t provide detailed contextual understanding of images.

2. **Proposed Solution:**

- An AI-powered web application where users can upload images to receive:
- Descriptive Captions (BLIP Model).
- Deeper Insights (Google Gemini AI).
- Insights may include objects, context, themes, and relevant interpretations.

3. **Target Users:**

- Content Creators & Marketers – Need AI-generated descriptions & insights for better engagement.
- Researchers & Educators – Require in-depth image analysis for academic and learning purposes.
- General Users – Want AI-generated insights and captions for various applications.

4. **Expected Outcome:**

   A functional AI-powered system that delivers accurate, detailed, and context-aware image captions & insights.

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for the AI-Powered Image Captioning & Insight System* to ensure smooth image processing and analysis.

Key Points:

1. **Technical Requirements:**

- Language: Python
- Backend: Google Gemini Pro Vision API, Hugging Face, OpenAI API
- Frontend: Streamlit

2. **Functional Requirements:**

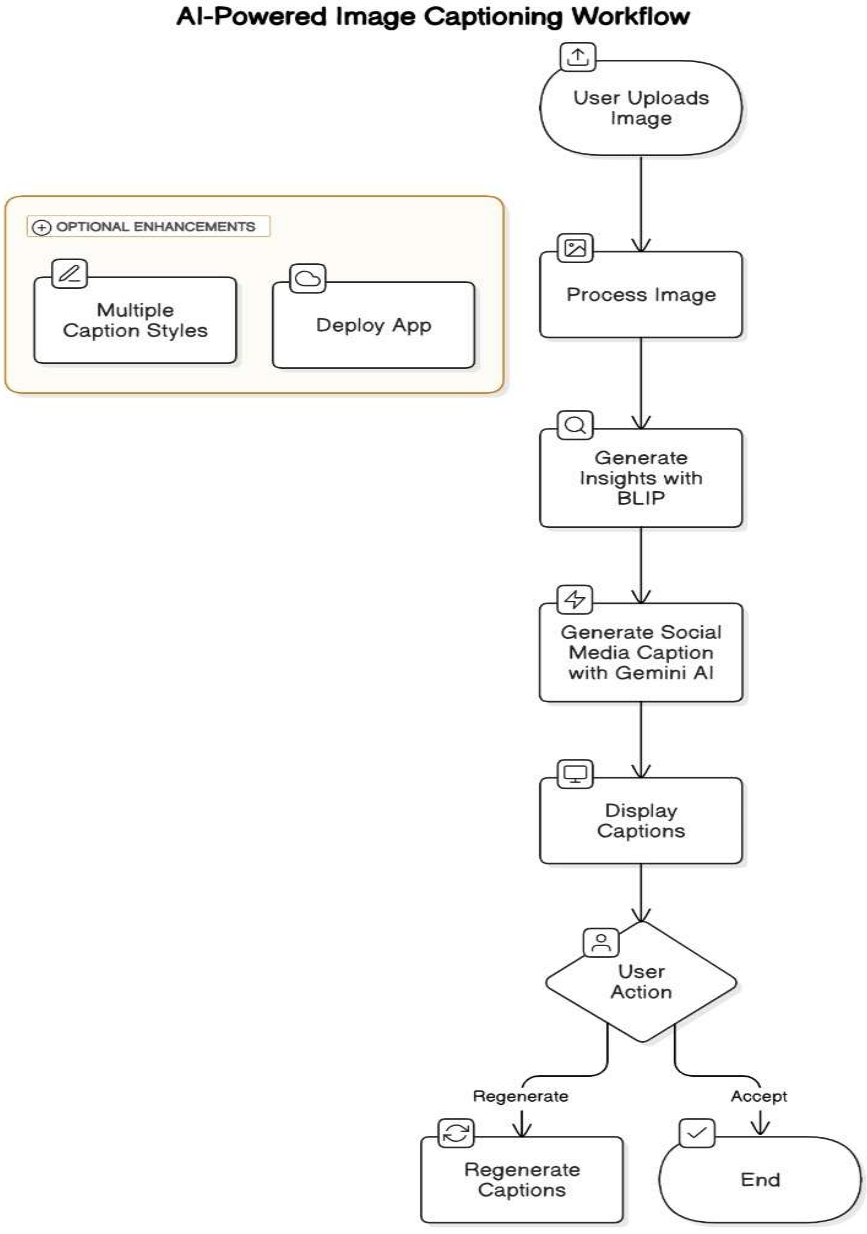
- Image Upload Support: Users can upload images
- AI-Powered Captioning & Insights
- BLIP Model generates a basic caption.
- Google Gemini AI provides deeper insights (object detection, theme analysis, contextual understanding).
- Regeneration Feature: Users can request new captions & insights.

3. **Constraints & Challenges:**
    - API Rate Limits: Optimizing queries to Google Gemini, Hugging Face to avoid restrictions.
    - Latency Issues: Ensuring fast image processing for a seamless experience.
    - Model Selection: Choosing the best pre-trained models for each category.
    - Scalability: Ensuring the system can handle multiple user requests simultaneously

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.



Key Points:

1. **System Architecture:**

    - User uploads an image* via the Streamlit UI.
    - Image is processed* by an AI model such as BLIP or Gemini AI.
    - AI extracts insights* from the image, generating relevant metadata.
    - Social media captions are generated* using Gemini AI.
    - Captions are displayed* to the user for review.
    - User can take action* by either accepting or regenerating captions.
    - Optional enhancements* include multiple caption styles and app deployment.

2. **User Flow:**
   Step 1: User uploads an image.
   Step 2: The image is processed by AI.
   Step 3: AI extracts insights using BLIP.
   Step 4: AI generates a social media caption with Gemini AI.
   Step 5: Captions are displayed to the user.
   Step 6: User decides to accept or regenerate captions.
   Step 7: If regenerated, new captions are generated. If accepted, the process endsUI/UX

3. **Considerations:**

    - Minimalist, user-friendly design for easy navigation.
    - Interactive caption review process allowing user modifications.
    - Optional caption styles to enhance output diversity.
    - Seamless API processing ensuring quick response times.
    - Optional app deployment for broader 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	Entire team	Google API Key, Hugging Face, OpenAI API, Python, Streamlit	API connection established & working
Sprint 1	Basic UI Development(Category selection and image upload)	🟡 Medium	2 hours (Day 1)	End of Day 1	Member 2	Streamlit setup	UI with upload and category selection ready
Sprint 2	Implement AI Processing (Category-wise Analysis)	🔴 High	3 hours (Day 2)	Mid-Day 2	Member 1 & 3	Integrated APIs	AI Processing for selected categories completed
Sprint 2	Debug API Isues & Optmize API calls	🔴 High	1.5 hours (Day 2)	Mid-Day 2	Member 2 & 1	API logs, UI inputs	API response issues fixed, optimized calls
Sprint 3	Test API Accuracy, Refine UI & fix Bugs	🟡 Medium	1.5 hours (Day 2)	Mid-Day 2	Entire team	API response, UI layout completed	Working prototype with accurate outputs
Sprint 3	Final Demo Preparation & Deployment	🟢 Low	1 hour (Day 2)	End of Day 2	Entire Team	Finalized UI & AI Processing	Demo-ready project

Sprint Planning with Priorities

Sprint 1 – Setup & Integration (Day 1)

🔴 **High Priority –** Set up the environment (Google Colab, VS Code, API keys).
🔴 **High Priority –** Integrate Google Gemini Vision Pro, Hugging Face, and OpenAI APIs.
🟡 **Medium Priority –** Build a basic UI with category selection & image upload.

Sprint 2 – Core Features & Debugging (Day 2)

🔴 **High Priority –** Implement AI processing (category-wise analysis).
🔴 **High Priority –** Debug API response issues, optimize API calls.
Sprint 3 – Testing, Enhancements & Submission (Day 2)

🟡 **Medium Priority –** Test API accuracy, refine UI, and fix UI bugs.
🟢 **Low Priority –** Final demo preparation & deployment

Phase-5: Project Development

Objective:

Implement core features of the provision AI.

Key Points:

1. **Technology Stack Used:**
    - **Frontend:** Streamlit
    - **Backend:** Google Gemini Flash API
    - **Programming Language:** Python
2. **Development Process:**
    - Integrate APIs: Set up authentication for Google Gemini Vision Pro, Hugging Face, and OpenAI APIs.
    - Develop Image Processing Logic: Implement category-based analysis (Historical, Medical, E-commerce, etc.).
    - Enable Reverse Image Search: Retrieve similar images and contextual data.
    - Build Streamlit UI: Design an intuitive interface for result visualization.
    - Optimize Performance: Enhance API request handling for faster responses.
3. **Challenges & Fixes:**
    - API Rate Limits and openAI issues:

- Implement batch processing to minimize redundant API calls and use caching for frequently queried results. Also, switch to a fallback model (local processing) when API limits are reached.

- Handling Diverse Image Categories:
- Develop a modular pipeline that dynamically routes images to the correct AI model based on category selection (or auto-detection), ensuring smooth processing across different challenges.

Phase-6: Functional & Performance Testing

Objective:

Ensure that the ProVision AI works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	budget friendly	Relevant data should be displayed	✅ Passed	Tester 1
TC-003	Performance Testing	API response time under 500ms	API should return results quickly.	⚠️ Needs Enhancements	Tester 3
TC-004	Bug Fixes & Improvements	Fixed incorrect API responses.	Data accuracy should be improved.	✅ Fixed	Developer tool
TC-005	Final Validation	Ensure UI is responsive across devices.	UI should work on all devices	Working properly	Tester 2
TC-006	Deployment Testing	Deploy using Streamlit Sharing	should be accessible online.	Need to Deployed	AWS

Final Submission

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