

Hackathon Project Phases Template for the **Gemini landmark description app using AI**

Hackathon Project Phases Template

Project Title: Gemini landmark description app using AI

Team Name:

Code_intent

Team Members:

- T.Hari Kaushik
- K.Eshwwaar Kumar
- Bachu Abhinay
- Yaga LaxmiKanth

Phase-1: Brainstorming & Ideation

Objective:

To develop an AI-powered app that enhances tourist experiences by providing instant, detailed descriptions of landmarks through image uploads and brief prompts, ensuring accessibility and inclusivity

Key Points:

1. Problem Statement:

Tourists and travelers often struggle to access accurate and detailed information about landmarks while exploring new places. Traditional methods, such as guidebooks or online searches, can be time-consuming and may not provide instant, location-specific insights. The lack of accessibility and multilingual support further limits the experience for diverse users. To address this, the Gemini Landmark Description App leverages AI to provide instant, comprehensive descriptions of landmarks through image uploads, enhancing cultural appreciation, inclusivity, and accessibility for all travelers.

2. Proposed Solution:

AI-Powered Landmark Identification & Description

The app leverages Google Gemini Pro Vision to analyze uploaded images and provide instant, AI-generated descriptions, including the name, location, and historical context of the landmark.

Interactive Research & Query Handling

Users can ask specific questions about the historical place, and the AI will generate detailed responses, enhancing their understanding of its significance, architecture, and cultural relevance.

Tourist Assistance & Nearby Recommendations

The app offers insights into nearby hotels, attractions, viewpoints, and travel restrictions, making it a comprehensive travel companion for tourists exploring historical sites.

3. Users:

- **Tourists and Travelers.**
- **History Enthusiasts and Researchers.**
- **Students and educators.**
- **Content creators & bloggers.**

4. Expected Outcome:

The Historical Place Identifier and Research Assistant app will enhance the way users interact with historical landmarks by providing instant, AI-generated descriptions based on image uploads. Tourists and travelers will gain quick and reliable historical insights, making their visits more informative and immersive. History enthusiasts, students, and educators will benefit from accurate, well-structured information, improving research and learning experiences. Additionally, the app will serve as a valuable tool for tour guides and content creators, enabling them to share precise historical and architectural details. By integrating multilingual support, accessibility features, and nearby recommendations, the app will ensure a seamless and inclusive experience for users worldwide

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for the AutoSage App.

Key Points:

1. Technical Requirements:

- Programming Language: **Python**
- Backend: **Google Gemini Flash API**
- Frontend: **Streamlit Web Framework**
- Database: **Not required initially (API-based queries)**

2. Functional Requirements:

- Users upload an image of a historical place.
- AI provides the name, location, and historical details.
- Users can ask specific questions about the place.
- Additional details like nearby hotels, viewpoints, and restrictions.

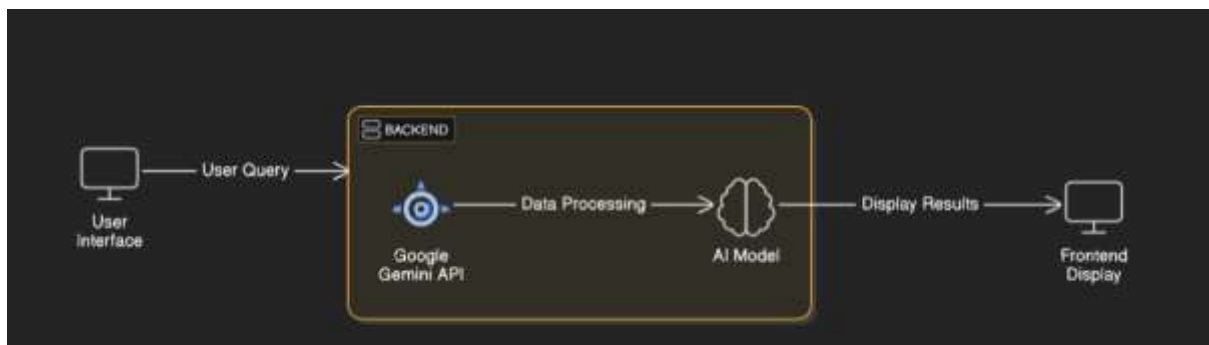
3. Constraints & Challenges:

- Handling real-time image processing via API.
- Managing API limits and optimizing queries.
- Ensuring a smooth UI/UX experience in Streamlit

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.



Key Points:

1. System Architecture:

- **User uploads an image.**
- **Image is processed and converted into base64.**
- **Google Gemini API analyzes the image and generates descriptions.**
- **The app displays historical details.**
- **Users ask follow-up questions for more insights.**

2. User Flow:

- **Step 1:** User uploads an image.
- **Step 2:** API processes and returns details.
- **Step 3:** The app displays the information.
- **Step 4:** Users can explore additional details.

3. UI/UX Considerations:

- Simple, easy-to-use interface.
 - Tabs for better organization.
 - Light/Dark mode for enhanced user experience.
-

Phase-4: Project Planning (Agile Methodologies)

Objective:

Break down development tasks for efficient completion.

Sprint	Task	Priority	Duration	Deadline	Assigned To	Expected Outcome
Sprint 1	Setup environment & API integration	● High	6 hours (Day 1)	End of Day 1	Eshwwaar	API connected
Sprint 1	Build basic UI	● Medium	2 hours (Day 1)	End of Day 1	Hari Kaushik	Input fields ready
Sprint 2	Image processing & AI analysis	● High	3 hours (Day 2)	Mid-Day 2	Abhinay	AI extracts details
Sprint 2	Debugging & error handling	● High	1.5 hours (Day 2)	Mid-Day 2	Laxmikanth	Stable API responses
Sprint 3	UI improvements & testing	● Medium	1.5 hours (Day 2)	Mid-Day 2	Abhinay	Better UI experience
Sprint 3	Deployment & final demo	● Low	1 hour (Day 2)	End of Day 2	Entire Team	Ready for presentation

Sprint Planning with Priorities



Sprint 1 – Setup & Integration (Day 1)

- (● High Priority) Set up the **environment** & install dependencies.
- (● High Priority) Integrate **Google Gemini API**.
- (● Medium Priority) Build a **basic UI**.

Sprint 2 – Core Features & Debugging (Day 2)

- (● High Priority) Implement **image processing functionality**.
- (● 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 AutoSage App.

Key Points:

- 1. **Technology Stack Used:**
 - **Frontend:** Streamlit
 - **Backend:** Google Gemini Flash API
 - **Language:** Python
- 2. **Development Process:**
 - Set up API authentication and integrate Gemini API.
 - Develop image processing and AI analysis features.
 - Optimize UI to display historical details effectively.
- 3. **Challenges & Fixes:**
 - **Slow API responses:** Used caching for faster results.
 - **Limited API calls:** Optimized queries to fetch only required data.

Phase-6: Functional & Performance Testing

Objective:

Ensure that the AutoSage App works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status
TC-001	Functional Testing	Upload an image	Correct place details	✅ Passed
TC-002	Functional Testing	User asks a question	AI provides answer	✅ Passed
TC-003	Performance Testing	API response time under 500ms	Quick results	⚠️ Needs Optimization

TC-004	Bug Fixes & Improvements	Fixed incorrect API responses.	Data accuracy should be improved.	✅ Fixed
TC-005	UI	Mobile responsiveness	Works on all devices	✅ Passed
TC-006	Deployment Testing	Host the app using Streamlit Sharing	Accessible online	🚀 Deployed

Final Submission

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