Hackathon Project Phases for the Gemini Landmark Description App

---

Hackathon Project Phases

Project Title:

Gemini Landmark Description App

Team Name:

The Atlas Coders

Team Members:

- Mutyala Lakshmi Vishnu Sahiti
- Kurella Sai Siri Chandana
- Machiraju Pranathi
- Mittapally Charmika

Phase-1: Brainstorming & Ideation

Objective:

The **Gemini Landmark Description App** enhances tourist experiences by leveraging AI-powered image recognition and natural language processing to provide real-time, informative descriptions of landmarks. Users can capture or upload an image, and the app generates detailed insights, including historical, cultural, and geographical information.

Key Points:

1. **Problem Statement:**
    - The **Gemini Landmark Description App** enhances tourist experiences by providing AI-generated descriptions of iconic landmarks.
    - Users can upload an image and enter a prompt to receive detailed insights on historical significance, architecture, and interesting facts.

2. **Proposed Solution:**

    - The **Gemini Landmark Description App** utilizes AI-powered image recognition and natural language processing to provide real-time, accurate, and multilingual descriptions of landmarks.
    - By capturing or uploading an image, users can instantly receive historical, cultural, and geographical insights through text and audio formats.
3. **Target Users:**

    - Tourists and travelers exploring new places.
    - History and culture enthusiasts.
    - Travel bloggers and content creators.
    - Educational institutions and students.
4. **Expected Outcome:**

    - The **Gemini Landmark Description App** will enhance tourist experiences by providing instant, AI-generated landmark descriptions with historical and cultural insights.

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for the **Gemini Landmark Description App**

Key Points:

1. **Technical Requirements:**

    - Programming Language: **Python**
    - Backend: **Gemini 1.5 Vision API**
    - Frontend: **Gradio**
    - Database: **Not required initially (API-based queries)**
2. **Functional Requirements:**

    - **Image Upload & AI Recognition** – Users upload an image, and AI identifies the landmark.
    - **Detailed AI Descriptions** – Provides historical, architectural, and cultural insights.
    - **Multilingual & Accessibility Features** – Supports multiple languages and text-to-speech
    - **Search, Save & Bookmark** – Users can search landmarks and save descriptions.

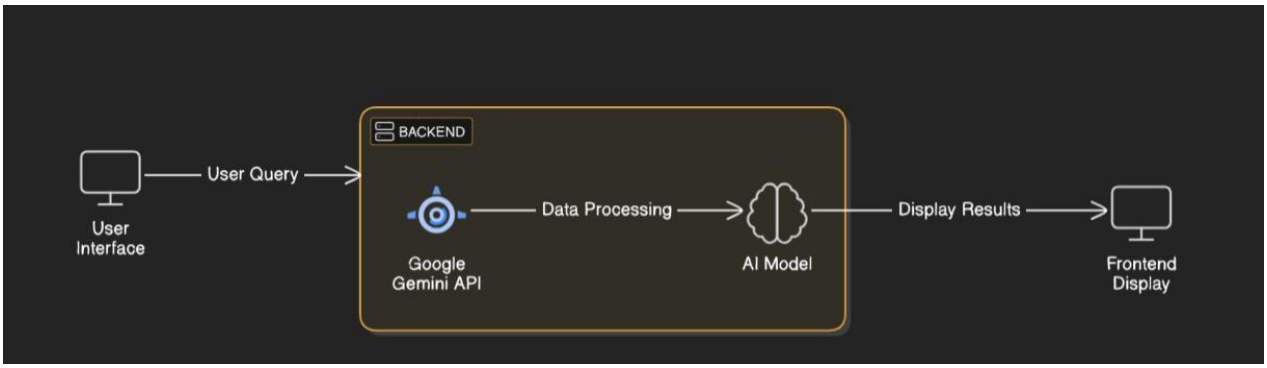
- **Sharing & Data Privacy** – Allows sharing descriptions while ensuring data security.

3. **Constraints & Challenges:**
    - **Accuracy & Recognition Issues** – AI may misidentify landmarks, especially lesser-known or partially visible ones.
    - **Real-Time Processing Limitations** – Generating instant responses with high accuracy requires significant computational power.
    - **Data Privacy & Security** – Ensuring user data protection while handling image uploads and AI-generated content.

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.



Key Points:

1. **System Architecture:**
    - **Layers** – Divided into Presentation, Application, and Data layers for modularity and scalability.
    - **Components** – Includes frontend, backend, database, and external integrations (APIs, cloud services).
    - **Communication** – Uses protocols like HTTP, WebSockets, or message queues for data exchange.
    - **Scalability & Security** – Designed for performance, redundancy, authentication, and data protection.
2. **User Flow:**

- **Entry Point** – User interacts with the system via a web/app interface or API request.
- **Processing** – Backend processes the request, applies logic, and retrieves/stores data.
- **Response & Feedback** – Processed data is sent back to the user with relevant actions or insights.

3. **UI/UX Considerations:**

- **User-Friendly Design** – Ensure intuitive navigation, clear layouts, and accessible elements.
- **Consistency & Aesthetics** – Maintain uniform branding, colors, and typography for a seamless experience.
- **Performance & Responsiveness** – Optimize loading speed and ensure compatibility across devices.

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	Pranathi	Google API Key, Python, Streamlit setup	API connection established & working
Sprint 1	Frontend UI Development	🟡 Medium	2 hours (Day 1)	End of Day 1	Sahiti	API response format finalized	Basic UI with input fields
Sprint 2	Landmark Search & Description Generation	🔴 High	3 hours (Day 2)	Mid-Day 2	Charmika	API response, UI elements ready	Search functionality with accurate descriptions
Sprint 2	Error Handling & Debugging	🔴 High	1.5 hours (Day 2)	Mid-Day 2	Siri Chandana	API logs, UI inputs	Improved API stability
Sprint 3	Testing & UI Enhancements	🟡 Medium	1.5 hours (Day 2)	Mid-Day 2	Sahiti	API response, UI layout completed	Responsive UI, better user experience
Sprint 3	Final Presentation & Deployment	🟢 Low	1 hour (Day 2)	End of Day 2	Entire Team	Working prototype	Demo-ready project

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 with input fields**.

Sprint 2 – Core Features & Debugging (Day 2)

(🔴 **High Priority)** Implement **search & comparison functionalities**.
(🔴 **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 **Gemini Landmark Description App.**

Key Points:

1. **Technology Stack Used:**
    - **Frontend:** Gradio
    - **Backend:** Gemini 1.5 Vision API
    - **Programming Language:** Python
2. **Development Process:**
    - Implement **API key authentication** and **Gemini API integration**.
    - Develop landmark description generation and tourist guidance logic.
    - Optimize search queries for accuracy and relevance in landmark details.
3. **Challenges & Fixes:**

- **Challenge:** Inaccurate or incomplete landmark descriptions.
  Fix: Fine-tune prompt engineering and leverage additional data sources.

- **Challenge:** High latency in generating responses.
  Fix: Implement caching and optimize API request handling.

Phase-6: Functional & Performance Testing

Objective:

Ensure that the Gemini Landmark Description App works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Upload an image of the Charminar	AI correctly identifies and describes the landmark	✅ Passed	Siri Chandana
TC-002	Functional Testing	Request description in English	AI provides an accurate English description	✅ Passed	Sahiti
TC-003	Performance Testing	API response time under 500ms	API should return results quickly.	⚠ Needs Optimization	Charmika
TC-004	Bug Fixes & Improvements	Fixed incorrect landmark identification	AI recognition accuracy should improve	✅ Fixed	Charmika
TC-005	Final Validation	Ensure app works properly	UI should be responsive	❌ Failed - UI broken on mobile	Sahiti
TC-006	Deployment Testing	Host the app on a cloud platform	App should be accessible online	🚀 Deployed	DevOps

---