Hackathon Project Phases:

Project Title:

Gemini Landmark Description App Enhancing Tourist Experiences with AI

Team Name:

GeoIntellects

Team Members:

- Anjana Bandam
- Varshitha Reddy Anugu
- Ruthika Rasala
- Chandana Vadathya

Phase-1: Brainstorming & Ideation

Objective:

Develop an AI-powered landmark description app using Gemini Flash to provide instant and detailed information about iconic landmarks.

Key Points:

1.Problem Statement:

- Tourists and curious individuals often struggle to quickly gather accurate and engaging information about landmarks they encounter.
- Traditional methods like guidebooks or manual internet searches can be time-consuming and language-restrictive.

2. Proposed Solution:

- An AI-powered app that uses Gemini Flash to generate real-time descriptions of landmarks based on uploaded images and user prompts.
- The app will provide historical significance, architectural details, and interesting facts about landmarks, with multilingual support and accessibility features.

3.Target Users:

- Tourists exploring new cities.
- Tour guides seeking quick, reliable information.
- History enthusiasts curious about global landmarks.

4.Expected Outcome:

- A functional AI landmark description app that delivers accurate, engaging, and accessible landmark insights in real time.

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

2. **Functional Requirements:**

- Upload landmark images for AI analysis.
- Enter text prompts for customized landmark descriptions.
- Display historical, architectural, and fun facts in a clear format.
- Provide multilingual support for a global audience.
- Ensure accessibility features like text-to-speech.

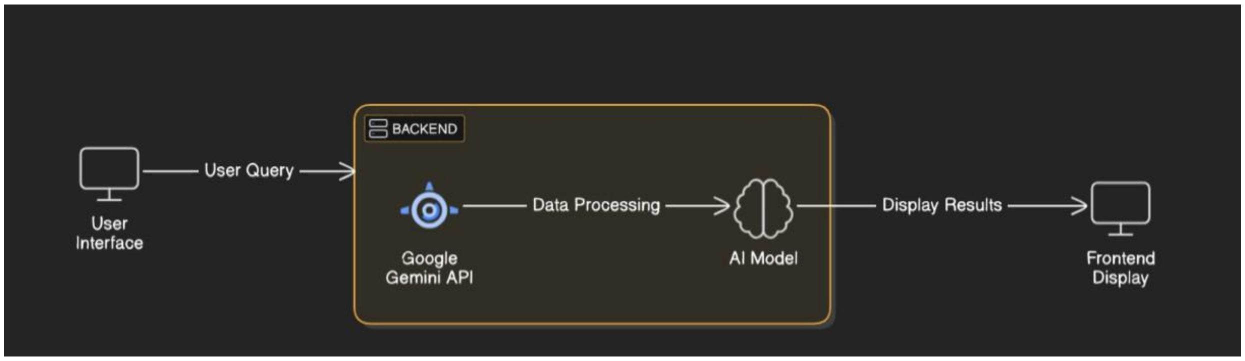
3. **Constraints & Challenges:**

- Ensuring real-time responses from Gemini API.
- Managing API rate limits.
- Designing a user-friendly, responsive UI

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.



Key Points:

1. **System Architecture:**

- User uploads a landmark image and/or inputs a text prompt.
- Query is processed using Google Gemini API.
- AI model generates landmark descriptions.
- The frontend displays details like historical significance, architecture, and facts.

2. **User Flow:**

- **Step 1:** User uploads an image or enters a text prompt.
- **Step 2:** Backend sends the request to the Gemini Flash API.
- **Step 3:** The AI processes data and returns descriptions.
- **Step 4:** The app presents AI-generated content in a user-friendly format

3. **UI/UX Considerations:**

- Clean, minimalist design for intuitive navigation.
- Filters for historical facts, architecture, and trivia.
- Options for light and dark modes.
- Accessibility-friendly fonts and color schemes.

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	End of Day 1	Anjana	Google API Key, Python setup	API connection established & working
Sprint 1	Frontend UI Development	□ Medium	2 hours	End of Day 1	Varshitha	API response format finalized	Basic UI with input fields
Sprint 2	Image Upload & AI Processing	● High	3 hours	Mid-Day 2	Ruthika	API & UI setup	Image processing & AI descriptions
Sprint 2	Error Handling & Debugging	● High	1.5 hours	Mid-Day 2	Chandana	API logs, UI inputs	Improved stability
Sprint 3	Testing & UI Enhancements	□ Medium	1.5 hours	Mid-Day 2	Team	Completed API and UI	Responsive design & bug fixes
Sprint 3	Final Presentation & Deployment	□ Low	1 hour	End of Day 2	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:** Streamlit
- **Backend:** Google Gemini Flash API
- **Programming Language:** Python

2. **Development Process:**

- Set up Gemini API key authentication.
- Implement image upload and AI response functionalities.
- Optimize API calls for faster data retrieval.
- Develop text-to-speech and multilingual support.

3. **Challenges & Fixes:**

- **Challenge:** API response delays.
  Fix: Implement caching for frequently queried landmarks.
- **Challenge:** Limited API call quotas.
  Fix: Optimize prompts to reduce redundant queries.

Phase-6: Functional & Performance Testing

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Upload landmark image & prompt	Accurate AI-generated landmark description	✓ Passed	Anjana
TC-002	Functional Testing	Query historical facts about a landmark	Relevant historical info displayed	✓ Passed	Varshitha
TC-003	Performance Testing	API response time < 500ms	Fast API response	⚠ Needs Optimization	Ruthika
TC-004	Bug Fixes	Fixed incorrect AI descriptions	Accurate data returned	✓ Fixed	Chandana
TC-005	UI Testing	Ensure mobile responsiveness	Works across devices	✗ Failed	Team

TC-006	Deployment Testing	Deploy using Streamlit Sharing	App accessible online	🚀 Deployed	Team

Final Submission

1. **Project Report** (based on this template)
2. **Demo Video** (3-5 minutes)
3. **GitHub/Code Repository Link**