# **GEN-AI Hackathon Project**

## **Project Title:**

**Gemini Landmark Description App: Enhancing Tourist Experiences with AI**

## **Team Name:**

Silver Eagles

## **Team Members:**

* P. Khyati
* M. Pranay Reddy
* C. Amshula
* P. Sai Charan
* M. Praveen

## **Phase-1: Brainstorming & Ideation**

### **Objective:**

Develop an AI-powered app using Gemini Flash to provide instant and detailed information about landmarks through image recognition, voice input, and data summarization.

### **Key Points:**

* **Problem Statement:**
  + Tourists and travelers often struggle to find quick and accurate information about landmarks they visit.
  + Language Barriers and the process of searching for historical, cultural, and contextual data manually can be time-consuming.
* **Proposed Solution:**
* **Comprehensive Landmark Insights** – Gain detailed information about landmarks through AI-based image analysis and historical context.
* **Seamless Navigation & Travel Assistance** – View the landmark on a map, get the best travel routes transportation options, and estimated costs.
* **Engaging Trivia & Cultural Etiquette** – Discover fun facts, unique trivia, and essential cultural etiquette tips for respectful interactions.
* **Smart Travel Planning** – Get seasonal travel advice, customized packing lists, and practical local tips for a smooth trip.
* **Nearby Attractions & Exploration** – Explore additional points of interest near the landmark for an enriched travel experience.
* **Multi-Language Support** – Use the built-in translator and customize language settings for seamless communication.
* **Real-Time Weather & Currency Updates** – Stay informed with live weather forecasts and currency exchange rates.
* **Safety & Emergency Assistance** – Access emergency contact numbers, embassy details, and essential safety guidelines for a secure journey.
* **Target Users:**
  + **Tourists:** Exploring cultural heritage sites and needing quick and accurate information.
  + **Travel Guides:** Accessing detailed landmark descriptions and historical insights.
  + **History Enthusiasts and Students:** Conducting research and learning about historical landmarks.
* **Expected Outcome:** 
  + A functional **AI-powered landmark information app** that delivers insights based on real-time data and user queries.

## **Phase-2: Requirement Analysis**

### **Objective:**

Define the technical and functional requirements for the Landmark Description App.

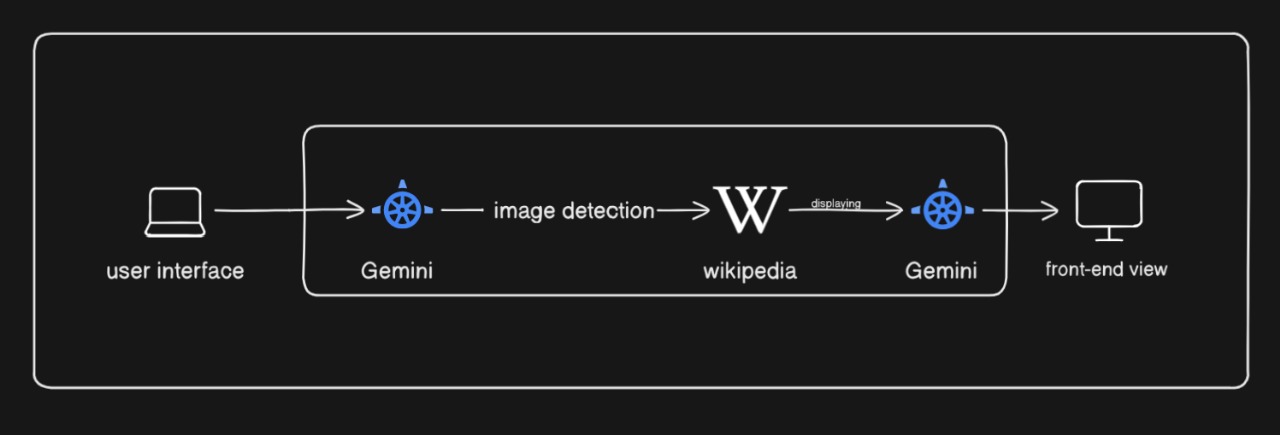
### **Key Points:**

1. **Technical Requirements:**
   * Programming Language: **Node JS**
   * Backend: **Google Gemini Flash API, Electron.JS**
   * Frontend: **HTML, CSS, React**
   * Database: **Not required initially (API-based queries)**
2. **Functional Requirements:**
   * Ability to analyze and recognize landmarks from uploaded images.
   * Generate AI-powered descriptions, including historical context and architectural features.
   * Provide multilingual support for a global audience.
   * Ensure accessibility features such as text-to-speech for visually impaired users.
3. **Constraints & Challenges:**
   * Managing real-time data for popular tourist destinations.
   * Ensuring the app provides accurate and reliable historical data.
   * Maintaining a user-friendly interface for diverse audiences.

## **Phase-3: Project Design**

### **Objective:**

Develop the architecture and user flow of the application.



### **Key Points:**

1. **System Architecture:**
   * **User Query:** The user inputs a landmark-related query via the User Interface.
   * **Image Processing:** The query is processed using the Google Gemini API for image analysis.
   * **Wikipedia Integration:** The processed data is sent to Wikipedia to retrieve detailed landmark descriptions.
   * **Frontend Display**: The app displays information such as historical significance, cultural relevance, and key facts about the landmark.
2. **User Flow:**
   * *Step 1:* User enters a query (e.g., "Describe the Eiffel Tower").
   * *Step 2:* The backend calls the Google Gemini API for image processing and landmark identification.
   * *Step 3:* The app retrieves detailed descriptions from Wikipedia.
   * *Step 4:* Displays the results in a clear and engaging format on the frontend.
3. **UI/UX Considerations:**
   * **Minimalist, user-friendly interface** for seamless navigation.
   * **Engaging Visuals**
   * **Easy-to-Read Format** for better user experience.

## **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 | Pranay | Gemini API Key, Node JS | API connection established and working |
| Sprint 1 | Frontend UI Development | 🟡 Medium | 2 hours (Day 1) | End of Day 1 | Khyati | API response format finalized | Basic UI with input fields |
| Sprint 2 | Image Processing & land mark recognition | 🔴 High | 3 hours (Day 2) | Mid-Day 2 | Amshula | API response, UI elements ready | Image recognition and AI description generation |
| Sprint 2 | Error Handling & Debugging | 🔴 High | 1.5 hours (Day 2) | Mid-Day 2 | SaiCharan | API logs, UI inputs | Improved API stability |
| Sprint 3 | Testing & UI Enhancements | 🟡 Medium | 1.5 hours (Day 2) | Mid-Day 2 | Praveen | 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 & Core Integration (Day 1)**

🔴 High Priority:

* Environment Setup: Create a project folder, initialize the environment, and install dependencies (e.g., React, Tailwind CSS, Vite).
* Integrate Google Gemini API: Establish a connection with the API for image processing and landmark recognition.
* Set Up Navigation System: Integrate a free maps API (e.g., Mapbox, OpenStreetMap) for route planning, travel modes, and cost estimation.

🟡 Medium Priority:

* **Build Basic UI:** Design input fields for image upload, source/destination for navigation, and other key features.
* Implement Weather API Integration: Add the weather feature with real-time updates using the existing API integration.

**Sprint 2 – Feature Implementation & Debugging**

🔴 High Priority:

* Image Description & Landmark Data Retrieval: Implement image processing and fetch relevant information from Wikipedia API and other data sources.
* Implementation of Core Features
* Geographic Location: Show landmarks on the map.
* Historical Context & Fun Facts: Provide engaging insights about the landmark.
* Translator & Text-to-Speech: Enable language customization and audio output.
* Navigation Features: Display routes, travel modes, times, costs, and recommend the best option.

🟡 Medium Priority:

* Testing Core Functionalities: Ensure the app can handle image uploads, search queries, and dynamic API responses effectively.
* Cultural Tips & Emergency Contacts: Display local dos and don'ts and provide emergency contact information for travelers.

**Sprint 3 – Testing, Refinements & Final Touches**

🔴 High Priority:

* Debugging & Error Handling: Fix API issues, validate input, and handle errors gracefully.
* Weather Integration Testing: Ensure real-time weather data is displayed correctly.
* UI/UX Enhancements: Refine the front-end design and improve user experience.

🟡 Medium Priority:

* Packing List Generator: Implement the feature to suggest packing items based on weather and activities.
* Final Testing & Bug Fixes: Conduct end-to-end testing for all functionalities, including navigation, image description, weather, and translations.

🟢 Low Priority:

* Demo Preparation: Create a presentation or demo video showcasing all features.
* Deployment: Ensure the app is ready for presentation and accessible to end users.

## **Phase-5: Project Development**

### **Objective:**

Implement core features of the Gemini Landmark Description App.

### **Key Points:**

1. **Technology Stack Used:**
   * **Frontend:** : HTML, CSS, React
   * **Backend:** Google Gemini Flash API, Electron.JS
   * **Programming Language:** Node JS
2. **Development Process:**
   * Implement **API key authentication** and **Gemini API integration**.
   * Develop **image recognition** and **AI-generated description logic.**
   * Optimize query processing for quick response times.
3. **Challenges & Fixes:**
   * **Challenge:** Delayed API response times.  
      **Fix:** Implement **caching** to store frequently queried results.
   * **Challenge:** Limited API calls per minute.  
      **Fix:** Optimize queries to fetch **only necessary data**.

## **Phase-6: Functional & Performance Testing**

### **Objective:**

Ensure that the Landmark Description App works as expected.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Category** | **Test Scenario** | **Expected Outcome** | **Status** | **Tester** |
| TC-001 | Functional Testing | Upload image of the Eiffel Tower | Accurate historical and architectural details should be displayed | ✅ Passed | Pranay |
| TC-002 | Functional Testing | Use text-to-speech feature | The AI-generated description should be read aloud clearly | ✅ Passed | Khyati |
| TC-003 | Performance Testing | API response time under 500ms | API should return results quickly. | ⚠ Needs Optimization | Amshula |
| TC-004 | Bug Fixes & Improvements | Fixed incorrect API responses. | Data accuracy should be improved. | ✅ Fixed | SaiCharan |
| TC-005 | Final Validation | Ensure UI is responsive across devices. | UI should work on mobile & desktop. | ❌ Failed - UI broken on mobile | Praveen |
| TC-006 | Deployment Testing | Host the app using Node JS Sharing | App should be accessible online. | 🚀 Deployed | Entire  Team |

## **Final Submission**

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