**Hackathon Project Phases Template**  for the **Gemini Landmark Description App** project.

**Hackathon Project Phases Template**

**Project Title:**

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

**Team Name:**

NGPA TEAM

**Team Members:**

* A.NIVYA SRI
* B.PRAGATHI VAISHNAVI
* A.GREESHMA SAI
* I.ANANYA

**Phase-1: Brainstorming & Ideation**

**Objective:**

**The Gemini Landmark Description App aims to enhance tourist experiences by providing personalized, AI-driven descriptions of landmarks. It offers real-time, context-aware information to enrich engagement and accessibility. The app also tailors content to individual preferences, creating unique, educational, and immersive experiences for each user.**

**Key Points:**

1. **Problem Statement:**

* Tourists often lack personalized and engaging information about landmarks, relying on generic, outdated, or inaccessible content that fails to enrich their experience.
* Traditional guides and static apps don't adapt to individual preferences, real-time contexts, or diverse languages, limiting the depth and accessibility of cultural or historical understanding.

1. **Proposed Solution:**

* An AI-powered app that provides real-time, immersive, and personalized landmark descriptions tailored to user preferences, interests, and languages
  + Enhanced experiences with interactive features like augmented reality overlays, voice narration, and dynamic recommendations for nearby attractions or activities.

1. **Target Users:**
   * Tech -savvy tourists seeking immersive and personalized travel experiences to deepen their connection with landmarks and cultural sites.
   * Independent travelers, families, and small groups who prefer accessible, real-time information over traditional tour guides.
2. **Expected Outcome:**

* Enhanced tourist satisfaction through personalized and engaging landmark experiences, fostering deeper cultural appreciation and higher app adoption rates globally.

**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: **Hugging Face ,Google cloud API**
   * Frontend: **Streamlit Web Framework**
   * Database: **Not required initially (API-based queries)**
2. **Functional Requirements:**

* User Personalization: Provide tailored landmark descriptions based on user preferences, interests, and past interactions.
* Real-Time Landmark Identification: Automatically detect nearby landmarks using geolocation and AR-based object recognition.
* Multilingual Support: Offer content in multiple languages with seamless translation**.**

1. **Constraints & Challenges:**

* Scalability: Ensuring the app can handle a large user base, diverse landmarks, and multilingual data processing without performance bottlenecks.
* Data Privacy: Balancing personalized experiences with strict adherence to user data protection and privacy regulations (e.g., GDPR).
* Cross-Platform Compatibility: Maintaining consistent functionality and performance across different operating systems, devices, and screen sizes.

**Phase-3: Project Design**

**Objective:**

Develop the architecture and user flow of the application.



**Key Points:**

1. **System Architecture:**
   * User enters landmark image query via UI.
   * Query is processed using **Google Cloud API**.
   * AI model fetches and processes the data.
   * The frontend displays **description according to given prompt.**
2. **User Flow:**
   * Step 1: User enters a query.
   * Step 2: The backend **calls the Gemini Cloud API** to retrieve image data.
   * Step 3: The app processes the data and **displays results** in an easy-to-read format.
3. **UI/UX Considerations:**
   * **Minimalist, user-friendly interface** for seamless navigation.
   * **Filters for price, mileage, and features**.
   * **Dark & light mode** 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 | Nivya Sri | Google API Key, Python, Streamlit setup | API connection established & working |
| Sprint 1 | Frontend UI Development | 🟡 Medium | 2 hours (Day 1) | End of Day 1 | Greeshma sai | API response format finalized | Basic UI with input fields |
| Sprint 2 | Upload image and giving prompt | 🔴 High | 3 hours (Day 2) | Mid-Day 2 | Pragathi Vaishnavi | API response, UI elements ready | Search functionality with filters |
| Sprint 2 | Error Handling & Debugging | 🔴 High | 1.5 hours (Day 2) | Mid-Day 2 | Nivya Sri, Ananya | API logs, UI inputs | Improved API stability |
| Sprint 3 | Testing & UI Enhancements | 🟡 Medium | 1.5 hours (Day 2) | Mid-Day 2 | Greeshma sai | 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 Cloud API**.  
 **(🟡 Medium Priority)** Build a **basic UI with input fields**.

**Sprint 2 – Core Features & Debugging (Day 2)**

**(🔴 High Priority)** Implement  **uploading of image and giving prompt.**  
 **(🔴 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:** Hugging Face, Google cloud API
   * **Programming Language:** Python
2. **Development Process:**
   * Implement **API key authentication** and **hugging face API integration**.
   * Develop **uploading a image and analysing description.**
   * Optimize **search queries for performance and relevance**.
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 **Gemini Landmark Description** App works as expected.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Category** | **Test Scenario** | **Expected Outcome** | **Status** | **Tester** |
| TC-001 | Functional Testing | Query "Eiffel tower picture" | Display history and architecture. | ✅ Passed | Nivya Sri |
| TC-002 | Functional Testing | Query "Falling tower of Pisa" | Display history and architecture. | ✅ Passed | An |
| TC-003 | Performance Testing | API response time under 500ms | API should return results quickly. | ⚠ Needs Optimization | Greeshma Sai |
| TC-004 | Bug Fixes & Improvements | Fixed incorrect API responses. | Database should be improved. | ❌Fixed | Ananya |
| TC-005 | Final Validation | Ensure UI is responsive across devices. | UI should work on mobile & desktop. | ❌ Failed - UI broken on mobile | Pragathi Vaishnavi |
| TC-006 | Deployment Testing | Host the app using Streamlit Sharing | App should be accessible online. | 🚀 Deployed | Ananya |

**Final Submission**

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