**Hackathon Project Phases Template** for the **AutoSage App** project.

# **Hackathon Project Phases Template**

## **Project Title: Gemini Landmark Description App Enhancing Tourists Experiences with AI**

## **Team Name: MAP AI**



## **Team Members:**

* RAJESH G (23RA1A05A8/CSE)
* SAIKUMAR G (23RA1A05C3/CSE)
* SAI KRISHNA G (23RA1A05C2/CSE)
* SANDESH M (23RA1A6777/CSD)

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

### **Objective:**

The Gemini Landmark Description App aims to enhance tourist experiences by leveraging AI-powered descriptions to provide immersive, informative, and personalized insights into historical landmarks, cultural sites, and attractions worldwide.

### **Key Points:**

1. **Problem Statement:**
   * AI-Powered Landmark Insights – Users can upload an image and input a prompt to receive AI-generated descriptions.
   * Comprehensive Information – Covers historical significance, architectural details, and interesting facts.
   * Enhancing Tourist Experiences – Ideal for travelers, tour guides, and history enthusiasts.
   * Global Cultural Connection – Provides knowledge about landmarks worldwide.
   * Multilingual Support – Enables access to descriptions in multiple languages.
   * Accessibility Features – Ensures inclusivity for all users, including those with disabilities.
   * Instant & Convenient – Quick access to detailed information anytime, anywhere.
2. **Proposed Solution:**
   * An AI-powered application using **Gemini Flash** to provide **real-time vehicle specifications, reviews, and comparisons.**
   * The app offers **maintenance tips** and **eco-friendly vehicle insights** based on user preferences.
3. **Target Users:**
   * **Vehicle buyers** looking for specifications and comparisons.
   * **Vehicle owners** needing seasonal maintenance tips.
   * **Eco-conscious consumers** searching for hybrid and electric vehicle options.
4. **Expected Outcome:**
5. * A functional **AI-powered vehicle information app** that provides insights based on real-time data and user queries.

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

### **Objective:**

Define the technical and functional requirements for the AutoSage App.

### **Key Points:**

|  |  |
| --- | --- |
| * **User Requirements** | * - Target Audience: Tourists, locals, varying tech literacy |

|  |  |
| --- | --- |
|  | * - Easy to Use: Intuitive design, minimal learning curve |

|  |  |
| --- | --- |
|  | * - Personalization: Tailored content based on preferences (language, interests) |

|  |  |
| --- | --- |
|  | * - Offline Functionality: Download maps and descriptions in advance |

|  |  |
| --- | --- |
|  | * - Multi-language Support: To cater to international tourists |

|  |  |
| --- | --- |
| * **Functional Requirements** | * - AI-Powered Landmark Descriptions: Real-time, dynamic descriptions based on user input or images |

|  |  |
| --- | --- |
|  | * - Location-based Services: GPS integration to suggest nearby landmarks |

|  |  |
| --- | --- |
|  | * - Interactive Maps: Detailed maps with landmark info, AR integration for immersive experience |

|  |  |
| --- | --- |
|  | * - Voice Assistance: Voice-controlled interaction for hands-free usage |

|  |  |
| --- | --- |
|  | * - Augmented Reality: Overlay information on landmarks via phone camera |

|  |  |
| --- | --- |
|  | * - Tour Recommendations: Personalized suggestions based on preferences |

|  |  |
| --- | --- |
|  | * - Itinerary Management: Create, modify, share custom itineraries |

|  |  |
| --- | --- |
|  | * - Feedback and Ratings: Allow users to rate and review landmarks |

|  |  |
| --- | --- |
| * **Non-Functional Requirements** | * - Scalability: Ability to add more landmarks, regions, and features |

|  |  |
| --- | --- |
|  | * - Performance: Quick load times and efficient data processing |

|  |  |
| --- | --- |
|  | * - Security: Protect user data (location, preferences) |

|  |  |
| --- | --- |
|  | * - Reliability: App stability with minimal downtime |

|  |  |
| --- | --- |
|  | * - Accessibility: Features for users with disabilities (voice control, contrast modes) |

|  |  |
| --- | --- |
| * **System Requirements** | * - Platform Support: Android and iOS availability |

|  |  |
| --- | --- |
|  | * - Third-Party Integrations: Maps, weather, transport info, etc. |

|  |  |
| --- | --- |
|  | * - AI Models: Use AI for landmark recognition and context-aware descriptions |

|  |  |
| --- | --- |
|  | * - Backend Services: Cloud backend for data storage and processing |

|  |  |
| --- | --- |
| * **Data Requirements** | * - Landmark Data: Accurate, up-to-date information on landmarks (history, images, significance) |

|  |  |
| --- | --- |
|  | * - User Data: Preferences, travel history, and feedback |

|  |  |
| --- | --- |
|  | * - Real-time Data: Weather, crowd info, live events |

|  |  |
| --- | --- |
| * **AI-Specific Requirements** | * - Training Data: Relevant datasets for AI model training (images, user interactions, landmark info) |

|  |  |
| --- | --- |
|  | * - AI Interpretation: AI to interpret text and voice queries about landmarks |

|  |  |
| --- | --- |
|  | * - Continuous Learning: AI improvement based on user interaction |

|  |  |
| --- | --- |
| * **Legal & Ethical Considerations** | Copyright Compliance: Respect copyright laws for content usage |

|  |  |
| --- | --- |
|  | * - Data Privacy: Compliance with regulations like GDPR |

|  |  |
| --- | --- |
|  | * - AI Bias: Address and mitigate biases in AI models |

|  |  |
| --- | --- |
| * **Monetization Strategy** | * - In-App Purchases: Premium features, ad-free options, exclusive content |

|  |  |
| --- | --- |
|  | * - Advertising: Relevant ads for local businesses (hotels, restaurants) |

|  |  |
| --- | --- |
|  | * - Partnerships: Collaborations with tourism boards or local attractions |

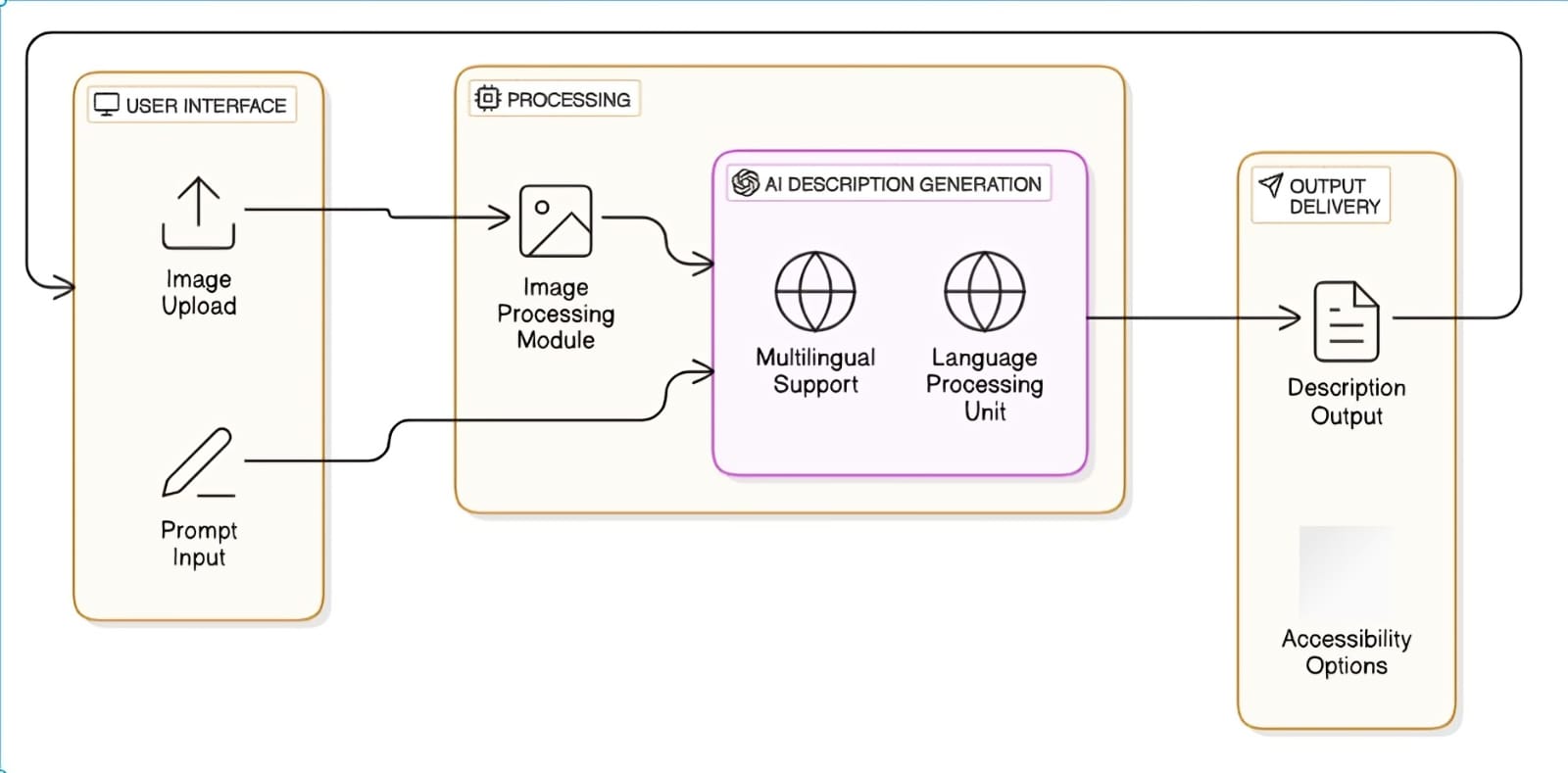
|  |  |
| --- | --- |
| * **Testing & Validation** | * - User Acceptance Testing (UAT): Real-world testing with tourists for feedback |

|  |  |
| --- | --- |
|  | * - Continuous Updates: Ongoing maintenance and updates to improve app functionality and features |

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

### **Objective:**

Develop the architecture and user flow of the application.



### **Key Points:**

1. **System Architecture:**
   * User enters input as image
   * Query is processed using **Google collab**.
   * AI model fetches and processes the data.
   * The frontend displays **location map,sign up and description space.**
2. **User Flow:**
   * Step 1: User uploads an image (e.g., “any place or destination”).
   * Step 2: The backend **calls the google collab** to retrieve landmark description.
   * 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 | Rajesh | Google API Key, Python, Streamlit setup | API connection established & working |
| Sprint 1 | Frontend UI Development | Medium | 1 hour (Day 1) | End of Day 1 | sandesh | API response format finalized | Basic UI with input fields |
| Sprint 2 | Search & Comparison | 🔴 High | 3 hours (Day 2) | Mid-Day 2 | Sai kumar | API response, UI elements ready | Search functionality with filters |
| Sprint 2 | Error Handling & Debugging | 🔴 High | 4 hours (Day 2) | Mid-Day 2 | Sai krishna and sai kumar | API logs, UI inputs | Improved API stability |
| Sprint 3 | Testing & UI Enhancements | 🟡 Medium | 1.5 hours (Day 2) | Mid-Day 2 | Sai krishna | 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 AutoSage App.

### **Key Points:**

1. **Technology Stack Used:**
   * **Frontend:** 10Web
   * **Backend:** google collab
   * **Programming Language:** Python
2. **Development Process:**
   * Implement **API key authentication** and **Gemini API integration**.
   * Develop **landmark locaters and maintenance tips logic**.
   * 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 AutoSage App works as expected.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Category** | **Test Scenario** | **Expected Outcome** | **Status** | **Tester** |
| TC-001 | Functional Testing | Query "Best budget cars under ₹10 lakh" | Relevant budget cars should be displayed. | ✅ Passed | Tester 1 |
| TC-002 | Functional Testing | Query "Motorcycle maintenance tips for winter" | Seasonal tips should be provided. | ✅ Passed | Tester 2 |
| TC-003 | Performance Testing | API response time under 500ms | API should return results quickly. | ⚠ Needs Optimization | Tester 3 |
| TC-004 | Bug Fixes & Improvements | Fixed incorrect API responses. | Data accuracy should be improved. | ✅ Fixed | Developer |
| TC-005 | Final Validation | Ensure UI is responsive across devices. | UI should work on mobile & desktop. | ❌ Failed - UI broken on mobile | Tester 2 |
| TC-006 | Deployment Testing | Host the app using Streamlit Sharing | App should be accessible online. | 🚀 Deployed | DevOps |

## **Final Submission**

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