

# Hackathon Project Phases Template

## Project Title:

AutoSage App Using Gemini Flash

## Team Name:

(Provide your team's name)

## Team Members:

- Member 1
  - Member 2
  - Member 3
  - Member 4
- 

## Phase-1: Brainstorming & Ideation

### Objective:

Develop an AI-powered vehicle expert tool using Gemini Flash to help users compare and analyze vehicle specifications, reviews, and eco-friendly options.

### Key Points:

#### 1. Problem Statement:

- Many users struggle to find reliable, up-to-date information about two-wheelers and four-wheelers before making a purchase decision.
- Users also need guidance on vehicle maintenance and eco-friendly vehicle choices.

## 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:

- 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:

### 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:

- Ability to **fetch vehicle details** using Gemini Flash API.
- Display **specifications, reviews, and comparisons** in an intuitive UI.
- Provide **real-time vehicle maintenance tips** based on seasons.
- Allow users to **search eco-friendly vehicles** based on emissions and incentives.

### 3. Constraints & Challenges:

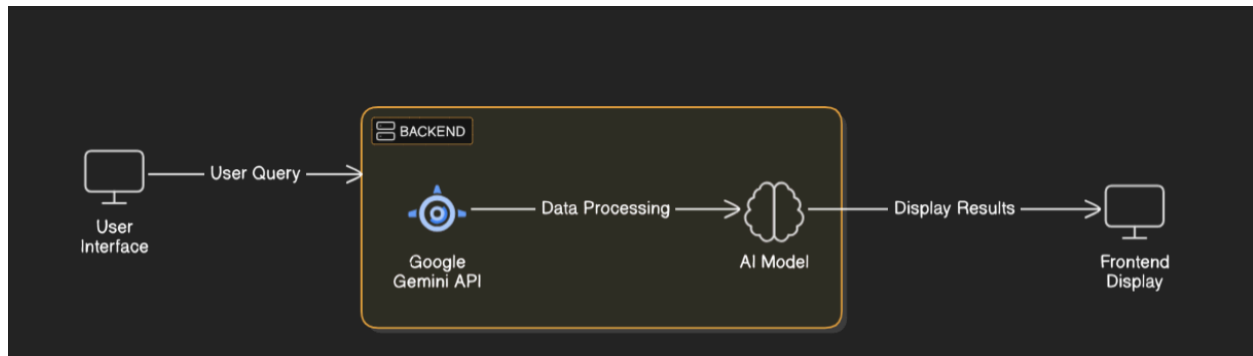
- Ensuring real-time updates from **Gemini API**.
- Handling **API rate limits** and optimizing API calls.
- Providing a **smooth UI experience** with Streamlit.

---

## Phase-3: Project Design

### Objective:

Develop the architecture and user flow of the application.



### Key Points:

#### 1. System Architecture:

- User enters vehicle-related query via UI.
- Query is processed using **Google Gemini API**.
- AI model fetches and processes the data.
- The frontend displays **vehicle details, reviews, and comparisons**.

#### 2. User Flow:

- Step 1: User enters a query (e.g., "Best motorcycles under ₹1 lakh").
- Step 2: The backend **calls the Gemini Flash API** to retrieve vehicle 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	Member 1	Google API Key, Python, Streamlit setup	API connection established & working
Sprint 1	Frontend UI Development	● Medium	2 hours (Day 1)	End of Day 1	Member 2	API response format finalized	Basic UI with input fields
Sprint 2	Vehicle Search & Comparison	● High	3 hours (Day 2)	Mid-Day 2	Member 1 & 2	API response, UI elements ready	Search functionality with filters
Sprint 2	Error Handling & Debugging	● High	1.5 hours (Day 2)	Mid-Day 2	Member 1&4	API logs, UI inputs	Improved API stability
Sprint 3	Testing & UI Enhancements	● Medium	1.5 hours (Day 2)	Mid-Day 2	Member 2& 3	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:** Streamlit
  - **Backend:** Google Gemini Flash API
  - **Programming Language:** Python
- 2. **Development Process:**
  - Implement **API key authentication** and **Gemini API integration**.
  - Develop **vehicle comparison 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**