Hackathon Project Phases Template

Project Title:

Smart Study planner Website

Team Name:

Tech Avengers

Team Members:

- K. Srinivasa Pranav
- L. Ganesh
- M. Hemanth Reddy
- K. Harshitha Reddy
- H. Mridhini

Phase-1: Brainstorming & Ideation

Objective:

- A smart study planner website that optimizes time management, tracks progress, and enhances productivity through personalized schedules and reminders.
- It should integrate AI-driven recommendations, collaboration features, and adaptive

learning strategies for efficient studying.

Key Points:

1. **Problem Statement:**

   Students struggle with organizing their study schedules effectively, leading to poor time management and inconsistent learning. A smart **powered Study Planner** can help by providing **personalized schedules, quizzes, reminders, and performance tracking** to enhance productivity and retention

2. **Proposed Solution:**

   Develop an **AI-powered Study Planner** that integrates **smart scheduling, quizzes, reminders, and performance tracking**, helping students manage their time efficiently and improve learning outcomes through personalized recommendations.

3. Target Users:

Students & Exam Aspirants – School, college, and competitive exam candidates needing structured study plans and reminders.

Self-Learners & Educators – Individuals taking online courses and teachers managing study schedules for students.

4. Expected Outcome:

Users can efficiently plan, track, and complete their study schedules with automated reminders and AI-driven suggestions.

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for the Study planner Web.

Key Points:

1. **Technical Requirements:**

- Programming Language: **Python,HTML,CSS,JS**
- Backend: **Google Gemini Flash API**
- Frontend: **AI Tools**

2. **Functional Requirements:**

- **User Management & Authentication –** Allow users to sign up, log in, and manage profiles with authentication via email or social media.

- **Study Schedule & Task Management –** Enable users to create, edit, and organize study plans with deadlines, priorities, and reminders.

- **Progress Tracking & Analytics –** Provide visual reports, study time tracking, and performance insights to monitor progress.

- **Notifications & Reminders –** Send automated alerts via email, SMS, or push notifications for upcoming tasks and deadlines.

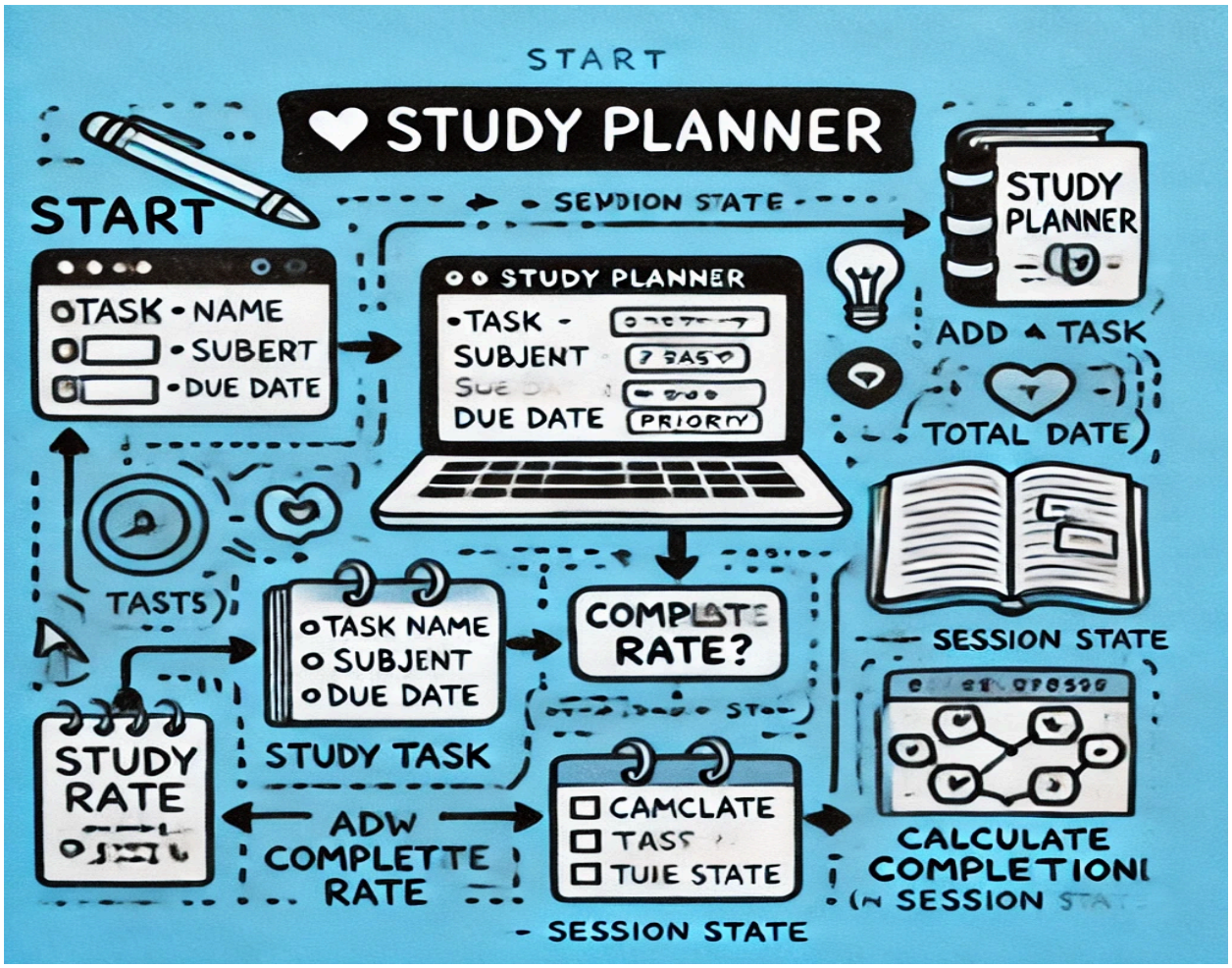
3. **Constraints & Challenges:**

    - Ensuring real-time updates from **Gemini API** .
    - Handling **API rate limits** and optimizing API calls.

Phase-3: Project Design

Objective:

this is an image to display the flow of work in the project


START
STUDY PLANNER
START
SEVDION STATE
STUDY PLANNER
TASK • NAME
• SUBERT
• DUE DATE
STUDY PLANNER
• TASK
SUBJENT
DUE DATE
PRIORTY
ADD TASK
TOTAL DATE
TASTS
TASK NAME
SUBJENT
DUE DATE
COMPLETE RATE?
SESSION STATE
STUDY RATE
STUDY TASK
ADW COMPLETTE RATE
CAMCLATE
TASS
TUIE STATE
CALCULATE COMPLETIONI
SESSION STATE

Key Points:

1. **System Architecture:**
    - User enters vehicle-related query via.
    - Query is processed using **Google Gemini API** .
    - AI model fetches and processes the data.
    - The frontend displays **username, calendar, schedule and quiz** .
2. **User Flow:**
    - Step 1: User enters a subject (e.g., "Timetable for effective study").
    - 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:**
    a. **Minimalist, user-friendly interface** for seamless navigation.

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	K. Srinivasa Pranav	Google API Key, Python, Streamlit setup	API connection established & working
Sprint 1	Frontend UI Development	🟡 Medium	2 hours (Day 1)	End of Day 1	L .Ganesh	API response format finalized	Basic UI with input fields
Sprint 2	Key features	🔴 High	3 hours (Day 2)	Mid-Day 2	K. Harshitha Reddy	API response, UI elements ready	Search functionality with filters
Sprint 2	Error Handling & Debugging	🔴 High	1 hour (Day 2)	Mid-Day 2	M. Hemanth Reddy	API logs, UI inputs	Improved API stability

Sprint 3	Testing & UI Enhancements	🟡 Medium	1 hour (Day 2)	Mid-Day 2	H. Mridhini	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:**

    a. **Frontend:** Streamlit
    - **Backend:** Google Gemini Flash API
    - **Programming Language:** Python

2. **Development Process:**

   a. 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:**

   a. **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 study planner web works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Query "remainder is set at the user relevant time "	should give remainders at the time set by user and display the task that	✅ Passed	H.Mridhini
TC-002	Functional Testing	Query "effective study tips "	According to the subject and the performance of the user suggestions must be displayed	✅ Passed	L. Ganesh M. Hemanth Reddy
TC-003	Performance Testing	API response time under 500ms	API should return results quickly.	⚠ Needs Optimization	K. Srinivasa Pranav
TC-004	Bug Fixes & Improvements	Fixed incorrect API responses.	Data accuracy should be improved.	✅ Fixed	K. Harshitha Reddy
TC-006	Deployment Testing	Host the web using Streamlit Sharing	Web should be accessible online.	🚀 Deployed	All Team

Final Submission

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