Hackathon Project Phases Template

Project Title:

StudBud: AI Study Planner

Team Name:

WINGS OF GARUDA

Team Members:

- Pinjala Siva Teja
- Thipparthi Swasthik
- Venkata Ramana Panigrahi
- Reddimasu Meghana
- Sadupati Trisha

Phase-1: Brainstorming & Ideation

Objective

StudBud is an AI-powered study planner that helps students optimize their study schedules, manage deadlines, and track progress. It uses AI to generate personalized study plans based on syllabus complexity, time availability, and learning patterns. The platform offers smart reminders, progress analytics, and adaptive scheduling to improve study efficiency.

Key Points:

1. **Problem Statement:**

   StudBud: AI Study Planner

1. **Proposed Solution:**
   - Technology Stack: Built using Next.js, JavaScript, Node.js, Tailwind CSS, HTML, and Firebase, with UI/UX designed in Figma.
   - Personalized Study Planning:Uses AI to generate study schedules based on syllabus complexity, time availability, and learning patterns.
   - Smart Features:Includes adaptive scheduling, smart reminders, and real-time progress tracking to enhance study efficiency.

- Chatbot Integration: An interactive AI-powered chatbot assists users in planning and managing their studies.
- Responsive UI:Designed with Next.js, Tailwind CSS, and HTML for a smooth and user-friendly experience across devices.

1. **Target Users:**

   Students (School & College): Those looking to manage study schedules and track progress

   Self-Learners & Online Course Takers: People learning through platforms like Coursera, Udemy.

1. **Expected Outcome:**

StudBud will provide an AI-powered study planner with personalized schedules, adaptive learning, and smart reminders. It features a chatbot for assistance and real-time progress tracking. The goal is to enhance study efficiency and help students achieve their academic goals.

Phase-2: Requirement Analysis

Objective:

1. Technical Requirements:

Frontend:Developed using Next.js, HTML, Tailwind CSS, and JavaScript for a responsive and interactive UI.

Backend: Built with Node.js and Express.js to handle study plan generation, chatbot responses, and user data management.

Database: Uses Firebase/MongoDB for storing user profiles, study schedules, and progress tracking.

AI Integration: Implements Machine Learning (ML) and Natural Language Processing (NLP) for personalized study plans and chatbot interactions.

Authentication & Security: User authentication via Firebase Auth or OAuth, with data encryption and secure storage.

Hosting & Deployment: Deployed on Vercel (Next.js) or AWS/GCP for scalability and reliability.

APIs & Third-Party Services: Integrates Google Calendar API for scheduling and reminders, and external EdTech APIs for course recommendations.

2. Functional Requirements:

User Registration & Login: Secure sign-up/login with options for Google or email authentication.

Personalized Study Plans: AI-generated schedules based on syllabus complexity, time availability, and learning patterns.

Smart Reminders & Notifications: Automated alerts for upcoming deadlines and study sessions.

Chatbot Assistance: AI-powered chatbot to provide study tips, schedule adjustments, and course recommendations.

Progress Tracking & Analytics: Visual dashboard displaying study progress, time spent, and goal completion.

Customizable Study Preferences: Users can set subjects, difficulty levels, and preferred study hours.

Collaboration Features: Option to create or join study groups for shared planning.

Dark Mode & UI Customization: User-friendly interface with theme customization for better accessibility.

Key Points:

1. **Technical Requirements:**

   - Programming Language: **Java Script**
   - Backend: **Node.js, Express.js**
   - Frontend: **Next.js, HTML, Tailwind CSS**
   - Database: **Firebase, MongoDB**

2. **Functional Requirements:**

   - User Registration & Login: Secure sign-up/login with options for Google or email authentication.
   - Personalized Study Plans: AI-generated schedules based on syllabus complexity, time availability, and learning patterns.
   - Smart Reminders & Notifications:Automated alerts for upcoming deadlines and study sessions.
   - Chatbot Assistance:AI-powered chatbot to provide study tips, schedule adjustments, and course recommendations.
   - Progress Tracking & Analytics:Visual dashboard displaying study progress, time spent, and goal completion.
   - Customizable Study Preferences: Users can set subjects, difficulty levels, and preferred study hours.
   - Ability to Fetch Vehicle Details: Uses **Gemini Flash API** to retrieve real-time vehicle specifications.
   - Vehicle Information & Comparisons: Displays specifications, reviews, and comparisons in an intuitive UI.
   - Real-Time Maintenance Tips: Provides seasonal vehicle maintenance suggestions for optimal performance.
   - Eco-Friendly Vehicle Search: Allows users to find vehicles based on emissions and available incentives.

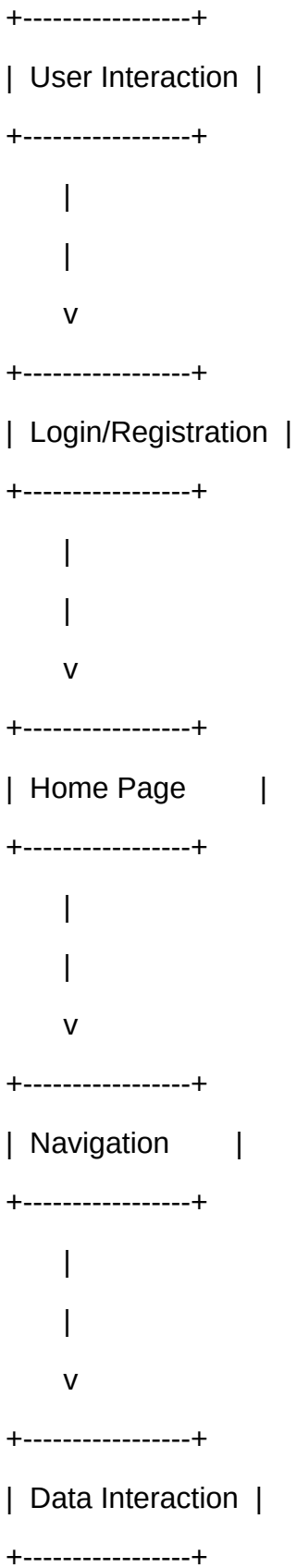
3. **Constraints & Challenges:**

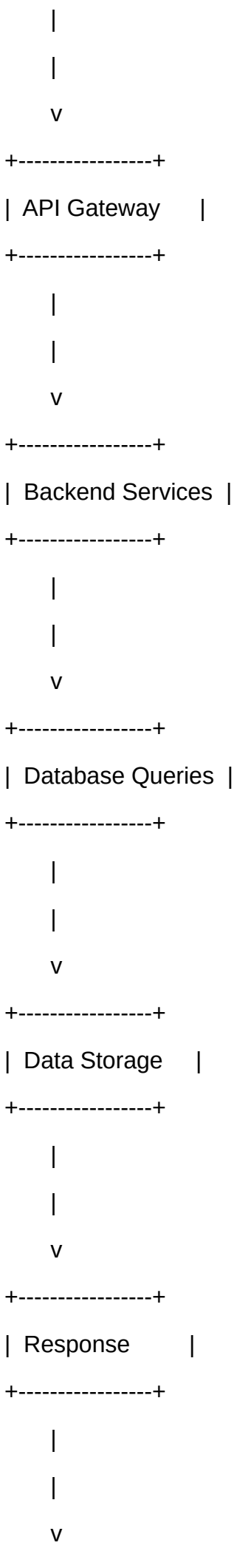
   - Ensuring real-time updates from **Gemini API**.
   - Handling **API rate limits** and optimizing API calls.
   - Providing a **smooth UI experience.**

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.




API Gateway
Backend Services
Database Queries
Data Storage
Response

```
+-----------------+
|  Logout/Error    |
+-----------------+
```

Key Points:

1. **System Architecture:**

2. **User Flow:**

Step 1: User opens **StudBud** via http://localhost:3001.
Step 2: User logs in or registers using **Google authentication or Email/Password**.
Step 3: Dashboard displays the **personalized study plan** based on user preferences.
Step 4: User interacts with the **AI chatbot** to get **study tips, adjust schedules.**
Step 5: System sends **smart reminders and notifications** for upcoming study sessions.
Step 6: User **tracks progress** through analytics and study completion status.
Step 7: AI **updates and refines the study plan** based on user progress and time availability.
Step 8: User can **log out or continue** studying as per their schedule.

3.**UI/UX Considerations:**

- **Dark Mode & Customization –** Users can switch themes and adjust font sizes for better readability.
- **Chatbot Optimization–** AI should provide quick, structured responses with suggested replies.
- **Mobile-Friendly & Fast–** Responsive design with fast loading times for a smooth experience.
- **Smart Notifications –** Subtle reminders and calendar sync to keep users on track.
- **Security & Privacy –** Secure authentication and user data protection features.

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 & AI Model Integration	🔴 High	6 hours (Day 1)	End of Day 1	Member 1	Python, TensorFlow/PyTorch, FastAPI setup	AI model integrated and functional
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 ready
Sprint 2	Study Plan Generator Implementation	🔴 High	3 hours (Day 2)	Mid-Day 2	Member 1 & 2	AI model, API connection established	AI-powered study plans generated
Sprint 2	Smart Insights & Progress Tracking	🔴 High	1.5 hours (Day 2)	Mid-Day 2	Member 1 & 4	Study plan generator functional	AI analytics & recommendations working
Sprint 3	Adaptive Scheduling & Notifications	🟡 Medium	1.5 hours (Day 2)	Mid-Day 2	Member 2 & 3	Study plan generator completed	Dynamic schedules & reminders enabled
Sprint 3	Final Testing & Deployment	🟢 Low	1 hour (Day 2)	End of Day 2	Entire Team	Fully functional application	Demo-ready project for presentation

Sprint Planning with Priorities

Sprint 1 – Setup & Integration (Day 1)

- (🔴 High Priority) Set up the environment & install dependencies.
- (🔴 High Priority) Integrate AI model with FastAPI backend.
- (🟡 Medium Priority) Build a basic UI with input fields.

Sprint 2 – Core Features & Debugging (Day 2)

- (🔴 High Priority) Implement AI-powered study plan generator.
- (🔴 High Priority) Develop smart insights & progress tracking.
- (🔴 High Priority) Debug AI model responses & optimize scheduling.

Sprint 3 – Testing, Enhancements & Submission (Day 2)

- (🟡 Medium Priority) Test AI-generated study plans, refine UI, & fix UI bugs.
- (🟢 Low Priority) Final demo preparation & deployment.

Phase-5: Project Development

Objective:

Implement core features of the StudBud AI Study Planner.

Key Points:

Technology Stack Used:

- **Frontend:** Next.js (Web)
- **Backend:** Gemini API Key

Development Process:

- Implement AI-driven study plan generation and adaptive scheduling.
- Develop progress tracking and smart insights functionality.
- Integrate with EdTech platforms (Google Calendar, Notion, Coursera, Udemy, etc.).
- Implement authentication (OAuth, Firebase Auth) and user onboarding.

Challenges & Fixes:

- **Challenge:**AI-generated schedules not aligning with user preferences.
  - **Fix:** Improve personalization by refining ML algorithms and user feedback loops.
- **Challenge:**API response time lag.
  - **Fix:** Optimize data caching and reduce redundant queries.
- **Challenge:**Handling multiple integrations efficiently.
  - **Fix:** Use asynchronous processing to improve performance and responsiveness.

Phase-6: Functional & Performance Testing

Objective:

Ensure that the StudBud: AI Study Planner works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Query "Generate a study plan for math exam"	AI should create a personalized study plan.	✅Passed	Tester 1
TC-002	Functional Testing	Query "Suggest best learning resources for physics"	AI should recommend relevant online resources.	✅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 progress tracking calculations	Data accuracy should be improved.	✅Fixed	Develop er
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 on a cloud platform	App should be accessible online.	🚀 Deployed	DevOps