

Hackathon Project Phases Template for the **Study plan** project.

Hackathon Project Phases Template

Project Title: Study plan

Team Name:

Team Techies

Team Members:

- Ram Charan
 - Ganesh
 - Madhu
-

Phase-1: Brainstorming & Ideation

Objective:

Develop a structured learning approach for mastering React development, focusing on building dynamic and scalable web applications. This study will cover core concepts, state management, routing, API integration, and performance optimization to enhance practical implementation skills.

Key Points:

1. **Problem Statement:**

Students and professionals looking for a structured and distraction-free study environment.

Individuals preparing for competitive exams, certifications, or academic assessments.

Teams or study groups working on research, projects, or collaborative learning.

2. **Proposed Solution:**

An AI-powered application that creates personalized study plans, tracks progress, and provides real-time study recommendations.

The app offers productivity tips, time management strategies, and adaptive learning insights based on user preferences

3. **Target Users:**

- Students and learners seeking detailed course information and study plan comparisons.
- Individuals needing timely academic tips and personalized strategies for exam preparation.
- Professionals and lifelong learners committed to sustainable, efficient study practices with digital resources.

4. Expected Outcome:

- A fully functional AI-powered study plan application that delivers personalized learning recommendations and insights based on real-time performance data and user queries.

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for the Study Plan App.

Key Points:

1. Technical Requirements:

- **Programming Language:** python
- **Backend:** Node.js with Express, integrated with the Google Gemini Flash API for AI-driven recommendations and real-time data processing.
- **Frontend:** React for a modern, responsive user interface.
- **Database:** MongoDB for persistent user data storage (initial development may leverage API-based queries if database integration isn't immediately required).

1. Functional Requirements:

Fetch Study Materials:

Ability to retrieve detailed course and study plan information using the Gemini Flash API.

Intuitive Display:

Present course details, user reviews, and comparative analyses of different study plans in a user-friendly interface.

Real-Time Study Recommendations:

Offer adaptive study tips and productivity advice based on users' current performance and upcoming academic deadlines.

Advanced Search Capabilities:

Allow users to search for courses and study resources based on subject matter, difficulty level, and personalized learning preferences

2. Constraints & Challenges:

• Ensuring Real-time Updates:

Guaranteeing that study plan recommendations and performance insights are updated instantly using the Gemini Flash API.

- **Handling API Rate Limits:**

Efficiently managing API rate limits and optimizing calls to maintain seamless data flow without interruptions.

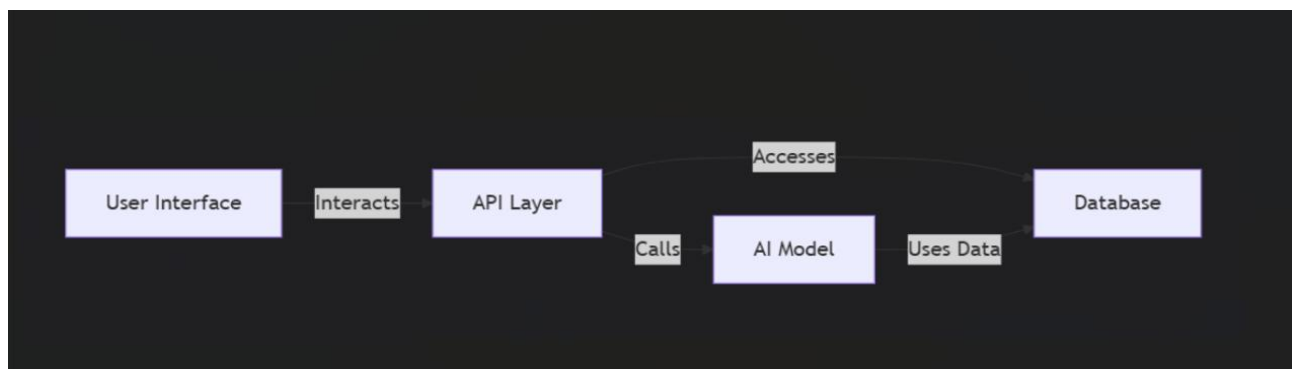
- **Optimizing UI Experience:**

Delivering a smooth, responsive, and engaging user interface with React that can handle dynamic content updates gracefully.

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.



Key Points:

1. System Architecture:

- **User Interaction:**
The user enters a study-related query via the UI.
- **Data Processing:**
The query is processed using the Google Gemini API.
- **AI Integration:**
The AI model fetches and processes study-related data.
- **Data Presentation:**
The frontend displays course details, study plans, and personalized recommendations.

2. User Flow:

Step 1: User enters a query (e.g., "Best online courses for data science").

- **Step 2:** The backend calls the Gemini Flash API to retrieve relevant study or course data.
- **Step 3:** The app processes the data and displays the results in an easy-to-read format..

3. UI/UX Considerations:

- **Interface Design:**

A minimalist, user-friendly interface that ensures seamless navigation.

- **Filtering Options:**

Filters for subjects, difficulty levels, and course ratings to help users refine their search.







- **Display Modes:**

Both dark and light modes are available for enhanced user experience.

Phase-4: Project Planning (Agile Methodologies)

Objective:

Break down development tasks for efficient completion in creating a **Study Plan** application.

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	Ram Charan	Google Gemini Flash API Key, Python, React setup	API connection established & working
Sprint 1	Frontend UI Development	 Medium	2 hours (Day 1)	End of Day 1	Ganesh	API response format finalized	Basic UI with input fields
Sprint 2	Vehicle Search & Comparison	 High	3 hours (Day 2)	Mid-Day 2	Madhu	API response, UI elements ready	Search functionality with filters
Sprint 2	Error Handling & Debugging	 High	1.5 hours (Day 2)	Mid-Day 2	Ram Charan	API logs, UI inputs	Improved API stability
Sprint 3	Testing & UI Enhancements	 Medium	1.5 hours (Day 2)	Mid-Day 2	Ganesh	API response, UI layout completed	Responsive UI, better user experience
Sprint 3	Final Presentation & Deployment	 Low	1 hour (Day 2)	End of Day 2	Ram Charan	Working prototype	Demo-ready project

Sprint Planning with Priorities

1. Sprint 1 – Setup & Integration (Day 1)

- (● High Priority) Set up the environment & install dependencies.
 - (● High Priority) Integrate Google Gemini Flash API.
 - (◐ Medium Priority) Build a basic UI with input fields.
2. Sprint 2 – Core Features & Debugging (Day 2)
- (● High Priority) Implement study resource search & comparison functionalities.
 - (● High Priority) Debug API issues & handle errors in queries.
3. 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 **Study Plan**.

Key Points:

1. **Technology Stack Used:**

- Frontend: React
- Backend: Google Gemini Flash API integration (via Node.js/Express)
- Programming Language: Python

2. **Development Process:**

API Authentication & Integration:

Implement API key authentication and integrate the Google Gemini Flash API to fetch personalized study resources and course data.

Feature Development:

Develop core logic for generating study plan recommendations and enabling course comparisons based on user queries.

Search Optimization:

Optimize search queries to enhance performance and ensure the relevance of the retrieved study materials

3. **Challenges & Fixes:**

● **Challenge:** Delayed API response times when fetching study resources.
Fix: Implement caching mechanisms to store frequently queried results for faster retrieval.

● **Challenge:** Limited API calls per minute.
Fix: Optimize queries to fetch only the necessary study resource data, minimizing redundant API calls

Phase-6: Functional & Performance Testing

Objective:

Ensure that the Study Plan App works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Query "Best online courses for Data Science"	Relevant study resources and course recommendations are displayed.	☑ Passed	Ram Charan
TC-002	Functional Testing	Query "Time management tips for exam preparation"	Adaptive study tips should be provided.	☑ Passed	Ganesh
TC-003	Performance Testing	API response time under 500ms	API should return results quickly.	⚠ Needs Optimization	Madhu
TC-004	Bug Fixes & Improvements	Fixed incorrect API responses for study recommendations.	Data accuracy should be improved.	☑ Fixed	Ram Charan
TC-005	Final Validation	Ensure UI is responsive across devices.	UI should work on mobile & desktop.	✗ Failed - UI broken on mobile	Ganesh
TC-006	Deployment Testing	Host the app using Vercel (or appropriate React hosting tool)	App should be accessible online.	🚀 Deployed	Madhu

Final Submission

1. **Project Report Based on the templates**

2. **Demo Video (3-5 Minutes)**
3. **GitHub/Code Repository Link**
4. **Presentation**