**Hackathon Project Phases**

## **Project Title:**

**StudBud: AI Study Planner using Hugging face**

## **Team Name:**

CODE MINDS

## **Team Members:**

* Shaik Ashraf Ahmed
* Prathamesh Kulkarni
* Shaik Amaan
* Renati Harshitha
* Samala Sai Shivani

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

### **Objective:**

To develop **AI-powered Study Planner** using **Hugging face** to assist students in creating personalized study plans based on **subject difficulty**, **study hours available**, and **deadlines**, we need to create an intelligent, efficient, and user-friendly system.

### **Key Points:**

1. **Problem Statement:**
   * Many students face difficulties when it comes to creating an effective study schedule, especially when trying to balance available study hours with the looming deadlines.
2. **Proposed Solution:**

* An AI-powered webapp using **Hugging face** to provide **effective study plan with required study hours.**
* The model offers Study schedule based on the difficult of the subject.

1. **Target Users:**
   * **Students** looking for personalized study planner.
   * **Students** needing study schedule based on the time available.
2. **Expected Outcome:**
   * A functional **AI-powered Study planner Website** that provides outcome based on user requirements.

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

### **Objective:**

Define the technical and functional requirements for the StudBud:AI Study Planner.

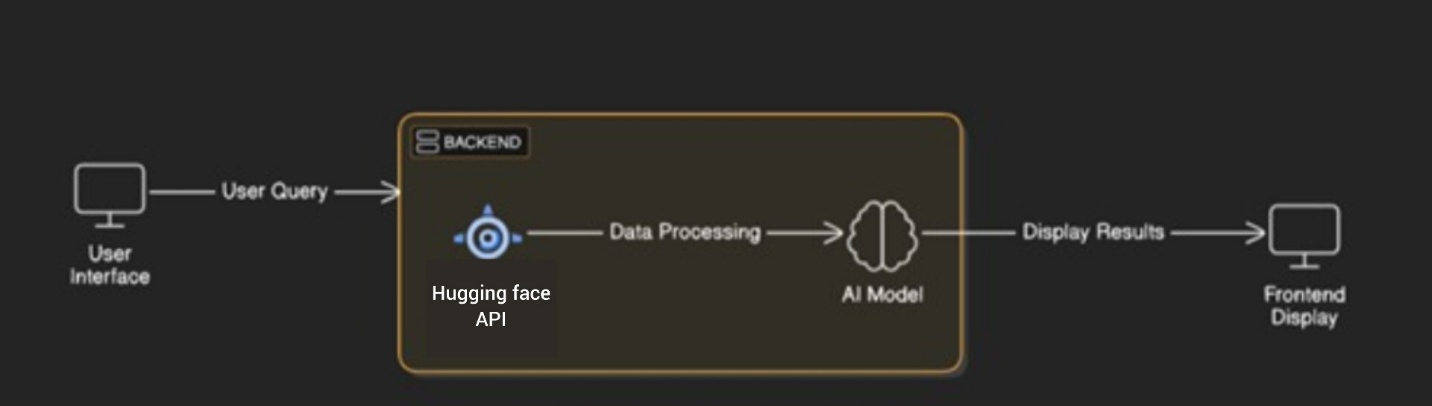
### **Key Points:**

1. **Technical Requirements:**
   * Programming Language: **Python**
   * Backend: **Hugging face API**
   * Frontend: **Html slicing**
   * Database: **API-based queries**
2. **Functional Requirements:**
   * Ability to generate study plan using Hugging face API.
   * Display customized study schedule in an intuitive UI.
   * Enable students to design their study plan according to their needs.
3. **Constraints & Challenges:**
   * Ensuring real-time updates from **Hugging face** **API**.
   * Handling **API rate limits** and optimizing API calls.
   * Providing a **smooth UI experience** with Html slice.

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

### **Objective:**

Develop the architecture and user flow of the .



### **Key Points:**

1. **System Architecture:**
   * User inputs their information to create schedule via UI.
   * Information is processed using **Hugging face API**.
   * AI model fetches and processes the data.
   * The frontend presents the customized study plan.
2. **User Flow:**
   * Step 1: User enters the information (e.g., “Study hours available,Subject difficulty,Deadline").
   * Step 2: The backend **calls the Hugging face API** to create a tailored study schedule.
   * 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.

## 

## **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 | Shaik Ashraf  Ahmed | Hugging face API Key, Python, Html slicing | API connection established & working |
| Sprint 1 | Frontend UI Development | 🟡 Medium | 2 hours (Day 1) | End of Day 1 | Shaik Amaan | API response format finalized | Basic UI with input fields |
| Sprint 2 | Input student requirement & study plan generation | 🔴 High | 3 hours (Day 2) | Mid-Day 2 | Samala Sai Shivani | API response, UI elements ready | Search functionality |
| Sprint 2 | Error Handling & Debugging | 🔴 High | 1.5 hours (Day 2) | Mid-Day 2 | Prathamesh Kulkarni | API logs, UI inputs | Improved API stability |
| Sprint 3 | Testing & UI Enhancements | 🟡 Medium | 1.5 hours (Day 2) | Mid-Day 2 | Renati Harshitha | 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 **Hugging face API**.  
 **(🟡 Medium Priority)** Build a **basic UI with input fields**.

### **Sprint 2 – Core Features & Debugging (Day 2)**

**(🔴 High Priority)** Implement **input & generating 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 StudBud: AI Study Planner webApp.

### **Key Points:**

1. **Technology Stack Used:**
   * **Frontend:** Html slicing
   * **Backend:** Hugging face API
   * **Programming Language:** Python
2. **Development Process:**
   * Implement **API key authentication** and **Hugging face API integration**.
   * Develop **personalized study schedule based on user preferences**.
   * Enhance **input data for efficiency**.
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 | "Study schedule for 10 days with two hours per day, physics subject" | Effective Study plan should be displayed. | ✅ Passed | Samala Sai Shivani |
| TC-002 | Performance Testing | API response time under 500ms | API should return results quickly. | ⚠ Needs Optimization | Shaik Ashraf  Ahmed |
| TC-003 | Bug Fixes & Improvements | Fixed incorrect API responses. | Data accuracy should be improved. | ✅ Fixed | Shaik Amaan |
| TC-005 | Final Validation | Ensure UI is responsive. | UI should work on desktop. | ❌ Failed - UI broken on mobile | Renati  Harshitha |
| TC-006 | Deployment Testing | Host the app using html slice | App should be accessible online. | 🚀 Deployed | DevOps |