# **Gen AI Hackathon**

## **Project Title:**

**StudBud: AI Study Planner**

## **Team Name:**

Learning Legends

## **Team Members:**

* K.Chandra Shekar Reddy
* B.Harsha Vardhan Reddy
* S.Harsha Vardhan Reddy
* B.Jeshwanth

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

### **Objective:**

Develop an AI-powered study planner tool using BERT to help students create personalized study schedules based on their goals, strengths, weaknesses, and preferences.

### **Key Points:**

**Problem Statement:**

**- Many students struggle with organizing their study schedules effectively.**

**- They face challenges in balancing study time across different subjects, identifying strengths and weaknesses, and staying motivated.**

**Proposed Solution:**

- An AI-powered application using BERT to provide personalized study plans.

- The app offers recommendations on what to study, when to study, and how to study, based on user inputs.

**Target Users:**

**- Students who need help organizing their study schedules.**

**- Teachers and tutors who want to provide personalized study plans for their students.**

**- Parents looking to support their children's academic progress.**

**Expected Outcome:**

**- A functional AI-powered study planner that provides personalized study schedules based on real-time data and user inputs.**

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

### **Objective:**

Define the technical and functional requirements for the StudBud App.

### **Key Points:**

**Technical Requirements:  
- Programming Language: Python**

**- Backend: BERT model for text analysis**

**- Frontend: Streamlit Web Framework**

**- Database: Not required initially (API-based queries)**

**Functional Requirements:**

**- Programming Language: Python**

**- Backend: BERT model for text analysis**

**- Frontend: Streamlit Web Framework**

**- Database: Not required initially (API-based queries)**

**Constraints & Challenges:**

**- Ensuring real-time updates from the BERT model.**

**- 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 study-related query via UI.**

**- Query is processed using BERT model.**

**- AI model fetches and processes the data.**

**- The frontend displays personalized study plans.**

1. **User Flow:**

**- Step 1: User enters a query (e.g., "How can I improve my math skills?").**

**- Step 2: The backend calls the BERT model to analyze the query.**

**- Step 3: The app processes the data and displays personalized study plans in an easy-to-read format.**

1. **UI/UX Considerations:  
   - Minimalist, user-friendly interface for seamless navigation.**

**- Filters for subjects, study methods, and preferences.**

**- 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 & Model Integration** | **🔴 High** | **6 hours (Day 1)** | **End of Day 1** | **B.Jeshwanth** | **Python Streamlit**  **setup,BERT model** | **Environment set up & model integrated |** |
| **Sprint 1** | **Frontend UI Development** | **🟡 Medium** | **2 hours (Day 1)** | **End of Day 1** | **Chandra shekhar** | **Basic UI design** | **Basic UI with input fields** |
| **Sprint 2** | **Study Plan Generation Logic** | **🔴 High** | **3 hours (Day 2)** | **Mid-Day 2** | **B.Harsha** | **User inputs, BERT model** | **Functional study plan generation** |
| **Sprint 2** | **Error Handling & Debugging** | **🔴 High** | **1.5 hours (Day 2)** | **Mid-Day 2** | **B.Jeshwanth&S.Harsha** | **User inputs, model responses** | **Improved stability & error handling** |
| **Sprint 3** | **Testing & UI Enhancements** | **🟡 Medium** | **1.5 hours (Day 2)** | **Mid-Day 2** | **B.Jeshwanth&K.Chandra shekhar** | **User feedback, UI layout** | **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.  
 (🟡 Medium Priority) Build a frontend UI development.**

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

**(🔴 High Priority) Study Plan Generation Logic  
 (🔴 High Priority) Error Handling & Debugging**

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

**(🟡 Medium Priority) Testing & UI Enhancements   
 (🟢 Low Priority) Final demo preparation & deployment.**

## **Phase-5: Project Development**

### **Objective:**

**Implement core features of the StudBud App.**

### **Key Points:**

1. **Technology Stack Used:**
   * **Frontend: Streamlit**
   * **Backend: BERT model**
   * **Programming Language: Python**
2. **Development Process:**
   * **Implement API key authentication and BERT model integration.**
   * **Develop study plan logic.**
   * **Optimize queries for performance and relevance.**
3. **Challenges & Fixes:**
   * **Challenge: Delayed model 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 StudBud 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**

**Code of studbud project**

**# Install necessary libraries**

**!pip install torch transformers**

**# Import necessary libraries**

**from transformers import BertTokenizer, BertForSequenceClassification**

**import torch**

**# Initialize the BERT model and tokenizer**

**def initialize\_bert\_model():**

**tokenizer = BertTokenizer.from\_pretrained('bert-base-uncased')**

**model = BertForSequenceClassification.from\_pretrained('bert-base-uncased', num\_labels=8)**

**return tokenizer, model**

**# Generate a study plan based on user input**

**def generate\_study\_plan(study\_goal, subject, strengths, weaknesses, available\_time, learning\_method, tokenizer, model):**

**"""**

**Generate a study plan based on the user's input.**

**Args:**

**study\_goal (str): User's study goal.**

**subject (str): Subject to focus on.**

**strengths (str): User's strengths.**

**weaknesses (str): User's weaknesses.**

**available\_time (str): Available study time.**

**learning\_method (str): Preferred learning method.**

**tokenizer: Pre-trained BERT tokenizer.**

**model: Pre-trained BERT model.**

**Returns:**

**str: Generated study plan.**

**"""**

**prompt = (f"Study goal: {study\_goal}\n"**

**f"Subject: {subject}\n"**

**f"Strengths: {strengths}\n"**

**f"Weaknesses: {weaknesses}\n"**

**f"Available time: {available\_time}\n"**

**f"Learning method: {learning\_method}")**

**# Tokenize the input prompt**

**inputs = tokenizer(prompt, return\_tensors="pt")**

**# Get the model output**

**outputs = model(\*\*inputs)**

**logits = outputs.logits**

**max\_index = torch.argmax(logits).item()**

**# Map the model's output to a study plan (example mapping)**

**study\_plans = [**

**"Focus on core concepts and practice quizzes.",**

**"Divide topics into smaller chunks and allocate time accordingly.",**

**"Emphasize weak areas and review frequently.",**

**"Adopt visual learning techniques like mind maps.",**

**"Spend more time on practice exams and problem-solving.",**

**"Combine group study sessions with self-paced learning.",**

**"Allocate consistent study hours daily with breaks.",**

**"Experiment with different methods to find what works best."**

**]**

**# Return the generated study plan**

**return study\_plans[max\_index]**

**# Initialize the model**

**print("Initializing BERT model...")**

**tokenizer, model = initialize\_bert\_model()**

**print("Model initialized.")**

**# Input data from the user**

**study\_goal = input("Enter your study goal: ")**

**subject = input("Enter the subject to focus on: ")**

**strengths = input("Enter your strengths: ")**

**weaknesses = input("Enter your weaknesses: ")**

**available\_time = input("Enter your available study time: ")**

**learning\_method = input("Enter your preferred learning method: ")**

**# Generate the study plan**

**print("\nGenerating your personalized study plan...")**

**study\_plan = generate\_study\_plan(**

**study\_goal, subject, strengths, weaknesses, available\_time, learning\_method, tokenizer, model**

**)**

**print("\nYour personalized study plan:")**

**print(study\_plan)**