**Hackathon Project Phases Template**  for the **Studymate App** project.

**Hackathon Project Phases Template**

**Project Title:**

**StudyMateV – A Multimodal AI-Powered Study Assistant**

**Team Name:**

Think sync

**Team Members:**

* Madhumitha Palla
* Yasaswini Malladi
* Kadali Kirthika

**Phase-1: Brainstorming & Ideation**

**Objective:**

Develop a multimodal AI-powered study assistant that answers questions based on various file types like PDF, PPT, images, text, and audio using Hugging Face and IBM Watson models.

**Key Points:**

**1.Problem Statement:**

* Students often struggle to extract and understand key information from study materials in different formats, making learning less efficient.

**2.Proposed Solution:**

* An interactive web app that uses AI to extract content from PDFs, PPTs, images, and audio files, then answers user queries using language models like Falcon-3B (Hugging Face) and IBM Granite (Watsonx).

**3. Target Users:**

* Students, educators, and self-learners needing quick, reliable answers from uploaded study content..

**4. Expected Outcome:**

* A functional web-based app that supports multimodal input and delivers accurate, AI-generated answers to improve study efficiency.

**Phase-2: Requirement Analysis**

**Objective:**

Define the technical and functional requirements for the StudyMateV app.

.

**Key Points:**

1. **Technical Requirements:**

* Programming Language: Python
* Frontend: Streamlit
* Backend: Hugging Face Transformers (Falcon-3B) and IBM Watsonx Granite (optional) via LangChain
* Database/Storage: FAISS for vector storage of embeddings
* Libraries Used: PyMuPDF, python-pptx, pytesseract, Whisper, LangChain, SentenceTransformers

2.**Functional Requirements:**

 Upload and process files: PDFs, PPTs, images, and audio

 Convert input into text using OCR (images), Whisper (audio), etc.

 Generate embeddings and store them in FAISS

 Answer natural language queries based on the content

 Provide accurate responses using Falcon-3B and/or IBM Watson models

**3. Constraints & Challenges:**

* Accurate extraction of text from different file formats
* Model selection based on API availability and latency
* Managing large file sizes and multiple formats efficiently
* API limits (for IBM Watsonx or Hugging Face endpoints

**Phase-3: Project Design**

**Objective:**

Develop the architecture and user flow of the application.



**Key Points:**

1. **System Architecture:**

*  **User Uploads File** (PDF, PPT, Image, or Audio) via Streamlit UI
*  **Content Extraction Layer**:
* PDF: PyMuPDF
* PPT: python-pptx
* Image: pytesseract (OCR)
* Audio: Whisper
*  **Text Embedding & Storage**:
* SentenceTransformers for embeddings
* FAISS for vector storage
*  **Question Input** (User enters a query)
*  **Answer Retrieval Layer**:
* LangChain routes the query
* Hugging Face (Falcon-3B) / IBM Watsonx Granite generates answer
*  **Frontend Display**: Streamlit shows the final answer

1. **User Flow:**

* User opens the app and uploads a study file
* App processes and extracts text content
* User enters a question based on the uploaded file
* Backend retrieves relevant text using FAISS
* AI model generates and returns the answer
* Streamlit displays the response to the user

1. **UI/UX Considerations:**

* Clean, minimal interface for ease of use
* File type icons and upload progress indicators
* Search box with suggestions and auto-complete
* Support for light/dark mode and mobile responsiveness

**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 | OCR integration  (py Tessaract) | 🔴 High | 6 hours (Day 1) | End of Day 1 | Yasaswini Malladi | Google API Key, Python, Streamlit setup | API connection established & working |
| Sprint 1 | Speech to text setup | 🟡 Medium | 2 hours (Day 1) | End of Day 1 | Madhumitha Palla | API response format finalized | Basic UI with input fields |
| Sprint 2 | UI Development  (streamlit) | 🔴 High | 3 hours (Day 2) | Mid-Day 2 | Kadali Kirthika | API response, UI elements ready | Search functionality with filters |
| Sprint 2 | Question answer model | 🔴 High | 1.5 hours (Day 2) | Mid-Day 2 | Madhumita and Yasaswini | API logs, UI inputs | Improved API stability |
| Sprint 3 | Testing and Bug fixing | 🟡 Medium | 1.5 hours (Day 2) | Mid-Day 2 | Kadali Kirtika | 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 development environment and install all necessary dependencies, including Python modules and Streamlit framework.

🔴 **High Priority**: Integrate the Google Gemini API for question answering and document analysis functionality.

**🟠 Medium Priority**: Design and build a basic user interface using Streamlit, with input fields for text, image, and voice-based queries.

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

🔴 High Priority: Implement core features such as search, comparison, and answer generation using Gemini API outputs.

🔴 High Priority: Debug issues in API integration, handle edge cases, and fix query parsing or response format inconsistencies.

**✅ \*\*Sprint 3 – Testing, Enhancements & Submission**

**(🟡 Medium Priority)** Testing and Bugging.  
 **(🟢 Low Priority)** Final **demo preparation & deployment**.

**Phase-5: Project Development**

**Objective:**

Implement the core functionality of the StudyMateV multimodal AI assistant.

**Key Points:**

1. **Technology Stack Used:**
   * Frontend: Streamlit
   * Backend: Python with LangChain
   * AI Models: Hugging Face Falcon-3B, IBM Watsonx Granite
   * Libraries: PyMuPDF (PDFs), python-pptx (PPTs), pytesseract (images), OpenAI Whisper (audio)
   * Database: FAISS (for storing vector embeddings**)**
2. **Development Process:**

 Create file handlers for each input type (PDF, PPT, Image, Audio).

 Use Whisper and OCR to transcribe audio and images.

 Convert extracted content into text and create vector embeddings.

 Store embeddings in FAISS and retrieve relevant chunks based on user query.

 Use LangChain to route the query to Falcon-3B or IBM Watsonx models.

 Display the answer in the Streamlit interface.

1. **Challenges & Fixes:**
   * **Challenge:** Managing large file sizes and multimodal content.
     + **Fix:** Chunking and embedding strategy for efficient retrieval.
   * **Challenge:** Switching between models depending on performance.
     + **Fix:** Model selection logic added with fallback to the best response.

**Phase-6: Functional & Performance Testing**

**Objective:**

Ensure StudyMateV performs accurately across all supported file types and delivers timely, reliable answers.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Category** | **Test Scenario** | **Expected Outcome** | **Status** | **Tester** |
| TC-001 | Functional Testing | Upload a PDF and ask a question from it | Correct answer from the PDF content is returned | ✅ Passed | Madhumita Palla |
| TC-002 | Functional Testing | Upload an image with text and ask a related question | Accurate OCR text extraction and relevant answer shown | ✅ Passed | Yasaswini Malladi |
| TC-003 | Functional Testing | Upload an audio lecture and ask a follow-up question | Audio transcribed correctly and context used for answer | ✅ Passed | Kadali Kirthika |
| TC-004 | Functional Testing | Upload a PowerPoint and ask a specific slide-based question | Content from PPT slide correctly referenced in the answer | ✅ Passed | Madhumita Palla |
| TC-005 | Performance Testing | Response time for question after upload | Answer delivered within 2–3 seconds | ⚠ Needs Tuning | Yasaswini Malladi |
| TC-006 | Compatibility Testing | Test app on mobile browser | Responsive layout and full functionality | ❌ UI Broken | Kadali Kirthika |
| TC-007 | Integration Testing | Switch between Hugging Face and IBM Watsonx Granite models | Seamless model routing with accurate responses | ✅ Passed | Madhumita Palla |

**Final Submission**

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