**Detailed Design- CS Tutor AI**

1. **Project Scope and Requirements:**

## **Introduction - Get to Know Our Project**

Our project is an AI Assistant designed for Computer Science students.   
The project is a user friendly web application - includes backend system and interactive UI.  
The application will serve as a comprehensive tool to enhance learning and understanding of course material.  
The user will be able to upload PDF files to the system, that will contain his summaries. In response, the system will ask from the user what is his specific request- note summarization, questions generations and checking or specific query response.

## **Project Scope**

**Target Users:** Computer Science students at university level, computer science lecturers, other curious people.

#### **1. Note Summarization**

* **Function:** Upload class notes and receive concise summaries.
* **Input:** Students can upload notes in PDF format
* **Process:** extract text from uploaded documents into a vector database and use LLMs to condense information into key points and summaries.

#### **2. Question Generation and Checking**

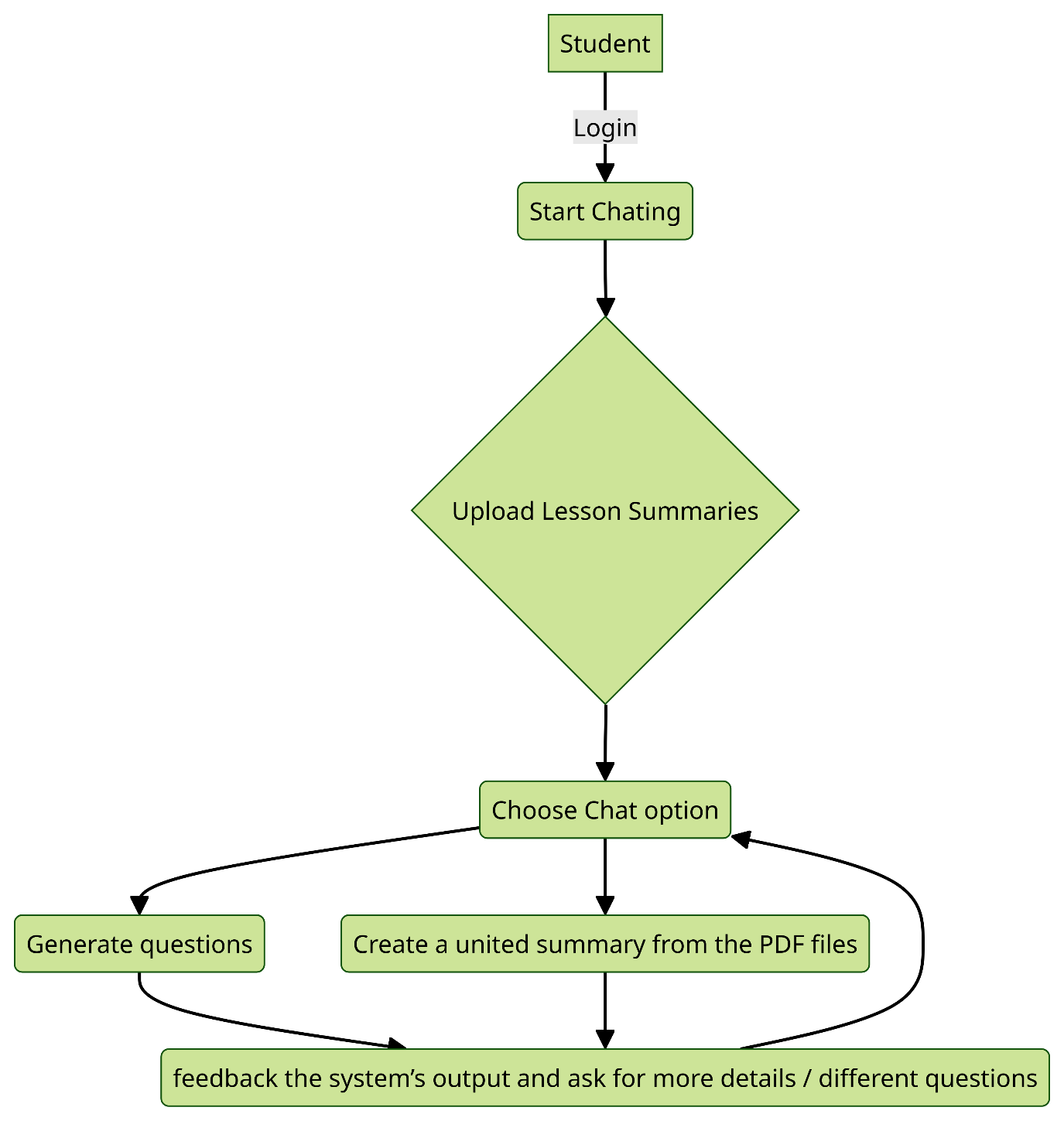
* **Function:** Create self-assessment questions in different levels from the uploaded PDF files and provide answer checking.
* **Input:** Same upload process as for note summarization.
* **Process:** generate multiple-choice or short-answer questions based on the content, provide a mechanism to check answers against model solutions.

#### **3. Query Response and Reference Provision**

* **Function:** Answer specific queries related to course material and provide references from uploaded notes.
* **Input:** Text input for queries, uploaded PDF files
* **Process:** Analyze the query and search uploaded notes for relevant content, present information and references directly from notes

**Functional Requirements:**

1. The user will be able to register to the system. While registering, the system will ask from the user the following details: Name, email, university name, studies subject
2. The registered users details will be saved into a database.
3. The user will be able to upload up to X pdf files, containing summaries related to the student’s studies subject. The files will have a size limitation. (final limitations decisions will be taken on later steps).
4. The system will be able to convert PDF files to text files. The files will go through pre-processing data process and preparing for indexing of the data.
5. When the user will interact with the chat, he will have two possible request types:
   1. Create a united summary from the PDF files
   2. Generate questions according to the amount requested by the user, in different levels, about the studies metirials and provide matching answers to self checking
6. The user will be able to feedback the system’s output and ask for more details / different questions / etc.
7. The system’s output will be saved and could be viewed as “history” and will be saved into a database.
8. All users’ summarizations and questions created by the system will be saved to a database.



**2. Modular Architecture:**

תמונה שמכילה טקסט, תרשים, צילום מסך, קו

התיאור נוצר באופן אוטומטי

**3. Programming Languages and Tools & Common Utilities and Libraries:**

**Platform:** Web Application.

**The programming languages that will be used in our project are:**

**Frontend:** React.js  
**Backend:** Python

**The technological tools we will be using in our project are:**  
**AI Orchestration Tier:** LangChain/ LLAMA\_index, Canopy. The AI Orchestration Tier manages and coordinates different parts of an AI system, like data processing and model deployment. It ensures everything works smoothly together, allocates resources, monitors performance, and automates tasks.  
**Language Models (LLMs):** we will use one of the following known APIs - OpenAI, Anthropic, Mistral.  
**Databases:** FireBase by Google.  
**Authorization Flow:** FireBase’s authorization feature  
**Monitoring & Evaluation:** langSmith , RAGAS  
**Vector DB:** pineCone

All decisions regarding the technology stack may change based on new releases and technologies.

Other than what mentioned here, we will use additional Python libraries according to our local demands.

|  |  |  |
| --- | --- | --- |
| **Task** | **Dates** | **Description** |
| Literature Review + Competitor Analysis | Submitted - Feb 13, 2024 |  |
| Functional and Non-Functional Requirements | Submitted - Feb 27, 2024 |  |
| Detailed Design Document | Submitted - Mar 5, 2024 |  |
| Meeting with Mentor | Mar 3, 2024 | Discuss project progress and receive guidance. Review timeline and adjust tasks if necessary. |
| Preparation of Presentation Slides | Mar 6 - Mar 8, 2024 | Create slides to communicate project objectives, scope, features, technologies, progress, etc. |
| Learning React.js | Mar 8 - Mar 19, 2024 | Team members learn React.js for frontend development. |
| Backend Development with Python | Mar 20 - Apr 2, 2024 | Develop backend functionalities like note summarization, question generation, and query response. |
| Integration of AI Models | Apr 3 - Apr 16, 2024 | Integrate AI models for tasks mentioned above (LLMs: OpenAI, Anthropic, Mistral). |
| Frontend Development with React.js | Apr 17 - Apr 30, 2024 | Develop user-friendly frontend for students to interact with the AI tools. |
| Testing and Debugging | May 1 - May 14, 2024 | Identify and fix bugs or issues. Ensure functionalities work as intended. |
| Documentation | May 15 - May 28, 2024 | Creating documentation |
| Final Checks and Submission | Deadline: June 2, 2024 | Perform final checks and submit the completed project. |

**4. TimeLine:**