**Software Requirement Specification**



**Project by Muhammad Abu Huraira**

2025

Table of Contents

[1. Introduction 5](#_Toc196004928)

[2. Vision Document 5](#_Toc196004929)

[2.1. Problem Statement: 5](#_Toc196004930)

[2.2. Business Opportunities 6](#_Toc196004931)

[2.3. Objectives 6](#_Toc196004932)

[2.4. Scope 6](#_Toc196004933)

[2.5. Constraints 6](#_Toc196004934)

[2.9. Stakeholder and User Description 6](#_Toc196004935)

[2.9.1. Market Demographics 6](#_Toc196004936)

[2.9.2. User Environment 6](#_Toc196004937)

[2.9.3. Stakeholder Profiles 6](#_Toc196004938)

[2.9.4. Stakeholder Summary 7](#_Toc196004939)

[3. System Requirements Specification 8](#_Toc196004940)

[3.1. System Features 8](#_Toc196004941)

[3.2. Functional Requirements 8](#_Toc196004942)

[3.2.1. File Upload and Storage 8](#_Toc196004943)

[3.2.2. Subject Folder Management 8](#_Toc196004944)

[3.2.3. AI-Powered Search 8](#_Toc196004945)

[3.2.4. Audio Transcription 8](#_Toc196004946)

[3.2.5. Audio Summarization 8](#_Toc196004947)

[3.2.6. AI Tutor Assistance 8](#_Toc196004948)

[3.2.7. Flashcard Generation 8](#_Toc196004949)

[3.2.8. To-Do List Management 9](#_Toc196004950)

[3.2.9. Calendar Integration 9](#_Toc196004951)

[3.2.10. CGPA Calculator 9](#_Toc196004952)

[3.2.11. Note-Taking Feature 9](#_Toc196004953)

[3.2.12. Real-Time Synchronization 9](#_Toc196004954)

[3.3. Non-Functional Requirements 9](#_Toc196004955)

[3.3.1. User-Friendly Interface 9](#_Toc196004956)

[3.3.2. Data Security 9](#_Toc196004957)

[3.3.3. Performance 9](#_Toc196004958)

[3.3.4. Privacy 9](#_Toc196004959)

[3.3.5. Compatibility 9](#_Toc196004960)

[3.3.6. Scalability 9](#_Toc196004961)

[3.3.7. Reliability 9](#_Toc196004962)

[3.3.8. Availability 10](#_Toc196004963)

[4.Use case 10](#_Toc196004964)

[4.1 Use case Model 10](#_Toc196004965)

[4.2 Use case specification 11](#_Toc196004966)

[Use Case Table 1: File Upload and Storage 11](#_Toc196004967)

[Use Case Table 2: Subject Folder Management 12](#_Toc196004968)

[Use Case Table 3: AI-Powered Search 13](#_Toc196004969)

[Use Case Table 4: Audio Transcription 14](#_Toc196004970)

[Use Case Table 5: Audio Summarization 15](#_Toc196004971)

[Use Case Table 6: AI Tutor Assistance 15](#_Toc196004972)

[Use Case Table 7: Flashcard Generation 16](#_Toc196004973)

[Use Case Table 8: To-Do List Management 17](#_Toc196004974)

[Use Case Table 9: Calendar Integration 18](#_Toc196004975)

[Use Case Table 10: CGPA Calculator 19](#_Toc196004976)

[Use Case Table 11: Note-Taking Feature 19](#_Toc196004977)

[Use Case Table 12: Real-Time Synchronization 20](#_Toc196004978)

[5.Diagrams 21](#_Toc196004979)

[5.1 System Sequence Diagram 21](#_Toc196004980)

[5.2 Domain Diagram 22](#_Toc196004981)

[5.3 Class diagram 23](#_Toc196004982)

[5.4 Data flow diagram 24](#_Toc196004983)

[Level 0 24](#_Toc196004984)

[Level 1 24](#_Toc196004985)

[Level 2 25](#_Toc196004986)

[5.5 Activity Diagram 26](#_Toc196004987)

[5.6 State Machine diagram 27](#_Toc196004988)

[5.7 Sequence diagram 28](#_Toc196004989)

[5.8 ER Diagram 29](#_Toc196004990)

Tables Content

[Figure 1 Summary 5](#_Toc196130375)

[Figure 2 supervisor team 6](#_Toc196130376)

[Figure 3 Development Team 7](#_Toc196130377)

[Figure 4 End User 7](#_Toc196130378)

[Figure 5 Admin 7](#_Toc196130379)

[Figure 6 Stakeholder Summary 7](#_Toc196130380)

# 

# 1. Introduction

This documentation is about **Notlyfe – One Go Student Application**. It’s an system to help students keep their study things like notes and recordings in one place. We’re making it with Flutter so it works on phones and the web. We use Supabase to save stuff and Grok AI to make it smart.

# 2. Vision Document

## 2.1. Problem Statement:

The problem statement outlines the challenges students face with disorganized study materials and inefficient study tools, which Notlyfe aims to address. The table format ensures clarity and structure.

Figure Summary

|  |  |
| --- | --- |
| ***Category*** | **Description** |
| ***Problem*** | - Study materials (notes, PDFs, audio recordings) are scattered across devices and platforms.  - Student may find this time-consuming to review these files spread across.  - Existing tools (e.g., Google Drive) lack student-specific features like AI assistance or CGPA tracking. |
| ***Affects*** | - Students at high school and university levels, struggling with academic organization and productivity. |
| ***Impact*** | - Wasted time searching for materials and lectures of different devices.  - Increased stress and time wastage from disorganization .  - Reduced study efficiency and academic performance. |
| ***Solution*** | One system that can:  - Organizes files, notes, and tasks in one place using Supabase.  - Uses Grok AI for search, transcription, summarization, and tutoring.  - Includes tools like a calendar, to-do list, and CGPA calculator built with Flutter. |

## 2.2. Business Opportunities

Notlyfe will helps students and can work with schools or study companies. It’s a new way to make studying easy with smart tools and cloud saving.

## 2.3. Objectives

* Make an app to upload content online and find study files easily.
* Add Grok AI to search, turn audio to text, and teach stuff.
* Put in tools like a calendar and grade checker with Flutter.
* Use Supabase to keep files safe and ready on all devices.

## 2.4. Scope

The **Notlyfe** scope involves a Flutter app for students to manage files, folders, notes, tasks, and schedules, with AI features (search, transcription, tutor, flashcards) using Supabase and Grok AI, excluding collaboration, third-party integrations, offline mode, and admin tools, targeting completion in 6 months.

## 2.5. Constraints

Notlyfe will be an app for phones and the web. It helps students save files, use AI, and stay organized. We start with Android, iPhone, and web.

## 2.9. Stakeholder and User Description

### 2.9.1. Market Demographics

Notlyfe is for students across globe, ages 16 to 25, from school or college who want help with studies.

### 2.9.2. User Environment

Students use Notlyfe on phones or web with internet. It works on Android, iPhone, or browsers.

### 2.9.3. Stakeholder Profiles

#### 2.9.3.1. Supervisor Team

Figure supervisor team

|  |  |
| --- | --- |
| ***Representatives*** | **Supervisor, Helper Supervisor** |
| ***Description*** | Watch over the project. |
| ***Type*** | People who know tech. |
| ***Responsibility*** | - Help the team.  - Check work.  - Make sure it’s good. |
| ***Success Criteria*** | All intended feature must work. |
| ***Involvement*** | Look at plans and updates. |
| ***Comments/Issues*** | None |

#### 2.9.3.2. Development Team

Figure Development Team

|  |  |
| --- | --- |
| **Representatives** | **Coders** |
| **Description** | Build Notlyfe. |
| **Type** | Tech people. |
| **Responsibility** | - Make the app.  - Plan tech stuff.  - Add features. |
| **Success Criteria** | Everything works fine. |
| **Involvement** | Code, write papers, test. |
| **Comments/Issues** | None |

#### 2.9.3.3. End Users

Figure End User

|  |  |
| --- | --- |
| **Representatives** | **Students** |
| **Description** | People who use Notlyfe. |
| **Type** | Outside people. |
| **Responsibility** | Use the app. |
| **Involvement** | None |
| **Comments/Issues** | None |

#### 2.9.3.4. Admin

Figure Admin

|  |  |
| --- | --- |
| **Representatives** | **Admin** |
| **Description** | Look after the app. |
| **Type** | Inside person. |
| **Responsibility** | Check users and fix problems. |
| **Involvement** | None |
| **Comments/Issues** | None |

### 2.9.4. Stakeholder Summary

Figure Stakeholder Summary

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Responsibility** |
| **Development Team** | Make Notlyfe with Flutter and AI. | Build app, plan tech. |
| **Supervisor Team** | Watch the project. | Help team, check work. |
| **End Users** | Students who use it. | Use app for study. |
| **Admin** | Keep app running. | Watch users, fix issues. |

# 3. System Requirements Specification

## 3.1. System Features

Notlyfe has these main things:

* **Files**: Save and sort study stuff like PDFs and audio.
* **AI**: Find stuff, turn audio to text, and help with study.
* **Tools**: Calendar, clock, to-do list, and grade checker.
* **Sync**: Keep everything the same on all devices.

## 3.2. Functional Requirements

### FR 3.2.1 File Upload and Storage

* The system shall let users put PDFs, PPTX, and audio in Supabase Storage.
* The system must keep files in special boxes for each user.

### FR 3.2.2. Subject Folder Management

* The system shall let users make folders for subjects in Supabase.
* The system must put files in these folders so they’re easy to find.

### FR 3.2.3. AI-Powered Search

* The system shall use Grok AI to find stuff when users ask (like “show me math notes”).
* The system must look at file names and text inside to find things.

### FR 3.2.4. Audio Transcription

* The system shall turn audio files into words with Grok AI.
* The system must save these words in Supabase in tabe.

### FR 3.2.5. Audio Summarization

* The system shall make short versions of audio words with Grok AI.

### FR 3.2.6. AI Tutor Assistance

* The system shall use Grok AI to explain things from files when users ask.
* The system must give answers that help with study.

### FR 3.2.7. Flashcard Generation

* The system shall make study cards from files with Grok AI.
* The system must let users keep and look at these cards.

### FR 3.2.8. To-Do List Management

* The system shall have a list for tasks in Supabase.
* The system must let users add, change, or finish tasks afterward.

### FR 3.2.9. Calendar Integration

* The system shall show a calendar with flutter\_calendar for tasks.
* The system must show all reminders in calender in Supabase.

### FR 3.2.10. CGPA Calculator

* The system shall let users put in grades to find their CGPA.

### FR 3.2.11. Note-Taking Feature

* The system shall let users write notes and save them in Supabase.
* The system must let users make notes bold or italic.

### FR 3.2.12. Real-Time Synchronization

* The system shall keep files and stuff the same on all devices with Supabase.
* The system must update everything in 5 seconds.

## 3.3. Non-Functional Requirements

### NFR 3.3.1. User-Friendly Interface

* The system must be easy to use with a simple Flutter screen.

### NFR 3.3.2. Data Security

* The system shall lock up data in Supabase so no one steals it.

### NFR 3.3.3. Performance

* The system must work fast and answer in 2 seconds.

### NFR 3.3.4. Privacy

* The system shall keep user info safe with Supabase login.

### NFR 3.3.5. Compatibility

* The system must work on Android, iPhone, and web like Chrome.

### NFR 3.3.6. Scalability

* The system shall handle 5,000 users at once with no slow-down.

### NFR 3.3.7. Reliability

* The system must work 99% of the time without breaking.

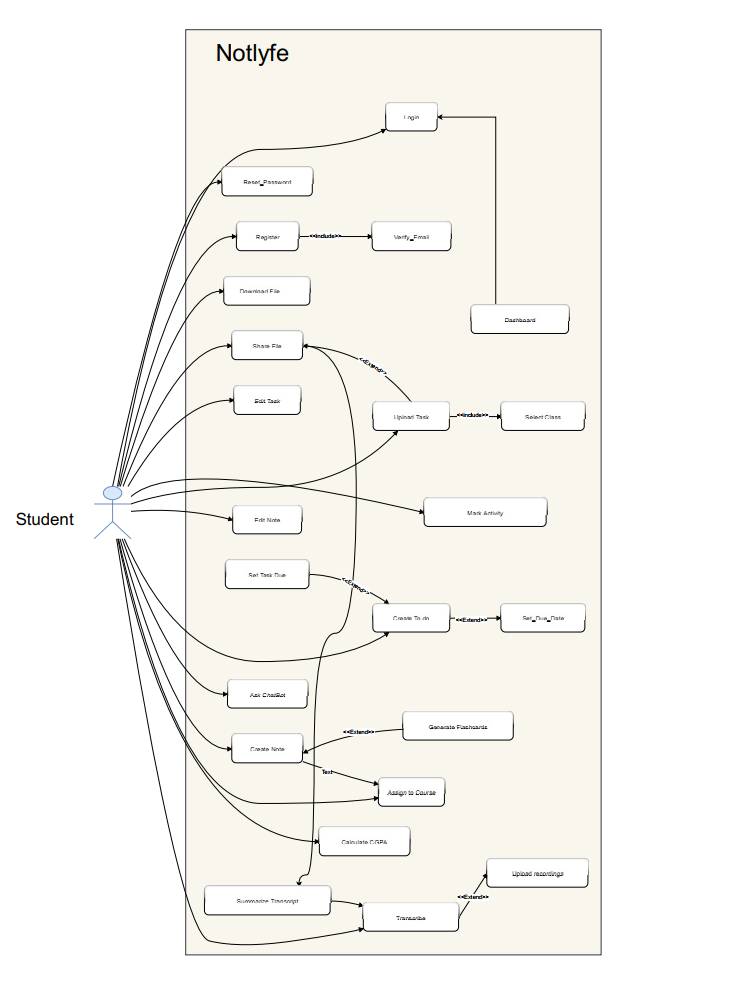
### NFR 3.3.8. Availability

* The system shall be ready to use all day, every day.

# 4.Use case

Use cases represent the potential are where the developing software will be utilized. Mention below diagram illustrate the potential use of the software.

## 4.1 Use case Model



## 4.2 Use case specification

### Use Case Table 1: File Upload and Storage

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Designation** | FR-01 |
| **Name** | File Upload and Storage |
| **Type** | Primary |
| **Source** | M. Bilal (domain expert for educational systems) |
| **Person responsible** | Muhammad Abu Huraira |
| **Description** | The system allows users to upload and store files (e.g., PDFs, PPTX, audio) in the cloud. |
| **Trigger event** | User selects a file to upload. |
| **Actors** | Student, Notlyfe System, Supabase Storage |
| **Pre-conditions** | User is logged in and has internet access. |
| **Post-conditions** | File is uploaded and stored in Supabase Storage. |
| **Result** | User have saved the file and can access from any device. |
| **Main scenario** | 1. User selects a file to upload.  2. System uploads the file to Supabase Storage.  3. System saves file metadata in the database. |
| **Alternative scenarios** | 4a. If upload fails due to network issues, system retries or notifies the user. |
| **Exception scenarios** | Trigger: File exceeds size limit.  Action: System notifies the user to upload a smaller file. |
| **Frequency of Use** | High |

### Use Case Table 2: Subject Folder Management

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Designation** | FR-02 |
| **Name** | Subject Folder Management |
| **Type** | Primary |
| **Source** | Muhammad Abu Huraira (domain expert for educational systems) |
| **Person responsible** | Muhammad Abu Huraira |
| **Description** | The system allows users to create and manage subject folders for organizing files. |
| **Trigger event** | User creates a new folder or organizes files. |
| **Actors** | Student, Notlyfe System, Supabase Database |
| **Pre-conditions** | User is logged in. |
| **Post-conditions** | Folders are created or updated in the Supabase database. |
| **Result** | User can organize files into subject folders. |
| **Main scenario** | 1. User creates a new folder or selects an existing one  2. System saves folder information in the database.  3. User moves files into the folder. |
| **Alternative scenarios** | 4a. If folder name already exists, system prompts for a new name. |
| **Exception scenarios** | Trigger: Database error during folder creation.  Action: System retries or notifies the user. |
| **Frequency of Use** | Medium |

### Use Case Table 3: AI-Powered Search

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Designation** | FR-03 |
| **Name** | AI-Powered Search |
| **Type** | Primary |
| **Source** | Muhammad Abu Huraira (domain expert for educational systems) |
| **Person responsible** | Muhammad Abu Huraira |
| **Description** | The system provides AI-powered search to find files and notes using textual prompts. |
| **Trigger event** | User enters a search query. |
| **Actors** | Student, Notlyfe System, Grok AI |
| **Pre-conditions** | User is logged in and has files or notes stored. |
| **Post-conditions** | Search results are displayed. |
| **Result** | User finds relevant files or notes quickly. |
| **Main scenario** | 1. User enters a search query.  2. System sends query to Grok AI.  3. Grok AI processes file the folder type and search the file.  4. System displays the results. |
| **Alternative scenarios** | 4a. If no results are found, system suggests refining the query. |
| **Exception scenarios** | Trigger: AI service is unavailable. Or no internet  Action: System falls back to keyword search. |
| **Frequency of Use** | High |

### Use Case Table 4: Audio Transcription

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Designation** | FR-04 |
| **Name** | Audio Transcription |
| **Type** | Primary |
| **Source** | Muhammad Abu Huraira (domain expert for educational systems) |
| **Person responsible** | Muhammad Abu Huraira |
| **Description** | The system transcribes uploaded audio files into text using Grok AI. |
| **Trigger event** | User uploads an audio file for transcription. |
| **Actors** | Student, Notlyfe System, Grok AI, Supabase Storage |
| **Pre-conditions** | User is logged in and has an audio file. |
| **Post-conditions** | Transcription is generated and stored in Supabase. |
| **Result** | User can read the transcription of the audio. |
| **Main scenario** | 1. User uploads an audio file.  2. System sends the file to Grok AI for transcription.  3. Grok AI returns the transcription.  4. System stores the transcription in the database. |
| **Alternative scenarios** | 4a. If transcription fails, system notifies the user. |
| **Exception scenarios** | Trigger: Audio file is corrupted.  Action: System rejects the file and asks for a valid one. |
| **Frequency of Use** | Medium |

### Use Case Table 5: Audio Summarization

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Designation** | FR-05 |
| **Name** | Audio Summarization |
| **Type** | Secondary |
| **Source** | Muhammad Abu Huraira (domain expert for educational systems) |
| **Person responsible** | Muhammad Abu Huraira |
| **Description** | The system generates summaries of transcribed audio using Grok AI. |
| **Trigger event** | User requests summarization after transcription. |
| **Actors** | Student, Notlyfe System, Grok AI |
| **Pre-conditions** | Transcription is available in the system. |
| **Post-conditions** | Summary is generated and stored in Supabase. |
| **Result** | User gets a concise summary of the audio content. |
| **Main scenario** | 1. User selects a transcription to summarize.  2. System sends the transcription to Grok AI.  3. Grok AI generates a summary.  4. System stores the summary. |
| **Alternative scenarios** | 4a. If summarization is not needed, user can skip. |
| **Exception scenarios** | Trigger: Summarization service is down.  Action: System notifies the user to try later. |
| **Frequency of Use** | Low to Medium |

### Use Case Table 6: AI Tutor Assistance

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Designation** | FR-06 |
| **Name** | AI Tutor Assistance |
| **Type** | Primary |
| **Source** | Shayan Zawar (domain expert for educational systems) |
| **Person responsible** | Shayan Zawar |
| **Description** | The system provides AI-powered tutoring to explain concepts from study materials. |
| **Trigger event** | User asks a question or requests help with a topic. |
| **Actors** | Student, Notlyfe System, Grok AI |
| **Pre-conditions** | User is logged in and has study materials uploaded. |
| **Post-conditions** | AI provides answers or explanations to the user. |
| **Result** | User gains a better understanding of the material. |
| **Main scenario** | 1. User asks a question.  2. System sends the question to Grok AI.  3. Grok AI generates an answer.  4. System displays the answer. |
| **Alternative scenarios** | 4a. If the question is unclear, AI asks for clarification. |
| **Exception scenarios** | Trigger: AI cannot answer the question.  Action: System suggests rephrasing or checking notes. |
| **Frequency of Use** | Medium |

### Use Case Table 7: Flashcard Generation

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Designation** | FR-07 |
| **Name** | Flashcard Generation |
| **Type** | Secondary |
| **Source** | Shayan Zawar (domain expert for educational systems) |
| **Person responsible** | Shayan Zawar |
| **Description** | The system generates flashcards from notes or transcripts using Grok AI. |
| **Trigger event** | User requests flashcards from a note or transcript. |
| **Actors** | Student, Notlyfe System, Grok AI |
| **Pre-conditions** | Notes or transcripts are available in the system. |
| **Post-conditions** | Flashcards are created and stored in Supabase. |
| **Result** | User can review key points using flashcards. |
| **Main scenario** | 1. User selects a note or transcript.  2. System sends it to Grok AI.  3. Grok AI generates flashcards.  4. System saves the flashcards. |
| **Alternative scenarios** | 4a. If content is too short, system notifies the user. |
| **Exception scenarios** | Trigger: AI fails to generate flashcards.  Action: System asks the user to try again later. |
| **Frequency of Use** | Low to Medium |

### Use Case Table 8: To-Do List Management

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Designation** | FR-08 |
| **Name** | To-Do List Management |
| **Type** | Primary |
| **Source** | Shayan Zawar (domain expert for educational systems) |
| **Person responsible** | Shayan Zawar |
| **Description** | The system allows users to create, edit, and manage their to-do lists. |
| **Trigger event** | User adds, edits, or completes a task. |
| **Actors** | Student, Notlyfe System, Supabase Database |
| **Pre-conditions** | User is logged in. |
| **Post-conditions** | Task list is updated in the Supabase database. |
| **Result** | User can track their tasks effectively. |
| **Main scenario** | 1. User adds a new task.  2. System saves the task in the database.  3. User can edit or mark tasks as complete. |
| **Alternative scenarios** | 4a. If task is overdue, system highlights it. |
| **Exception scenarios** | Trigger: Database error during task save.  Action: System retries or notifies the user. |
| **Frequency of Use** | High |

### Use Case Table 9: Calendar Integration

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Designation** | FR-09 |
| **Name** | Calendar Integration |
| **Type** | Secondary |
| **Source** | Shayan Zawar (domain expert for educational systems) |
| **Person responsible** | Shayan Zawar |
| **Description** | The system integrates a calendar for scheduling tasks and events using Flutter packages. |
| **Trigger event** | User views or adds events to the calendar. |
| **Actors** | Student, Notlyfe System, Flutter Calendar Package |
| **Pre-conditions** | User is logged in. |
| **Post-conditions** | Calendar events are saved and displayed in the UI. |
| **Result** | User can manage their schedule effectively. |
| **Main scenario** | 1. User opens the calendar.  2. User adds a new event or views existing ones.  3. System syncs events with the database. |
| **Alternative scenarios** | 4a. If events conflict, system warns the user. |
| **Exception scenarios** | Trigger: Calendar sync fails.  Action: System retries or shows an error. |
| **Frequency of Use** | Medium |

### Use Case Table 10: CGPA Calculator

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Designation** | FR-10 |
| **Name** | CGPA Calculator |
| **Type** | Secondary |
| **Source** | Shayan Zawar (domain expert for educational systems) |
| **Person responsible** | Shayan Zawar |
| **Description** | The system calculates the user’s CGPA based on input grades. |
| **Trigger event** | User enters grades and requests CGPA calculation. |
| **Actors** | Student, Notlyfe System |
| **Pre-conditions** | User is logged in. |
| **Post-conditions** | CGPA is calculated and displayed in the Flutter UI. |
| **Result** | User knows their current CGPA. |
| **Main scenario** | 1. User enters grades for each course.  2. System calculates CGPA using a predefined formula.  3. System displays the result. |
| **Alternative scenarios** | 4a. If grades are incomplete, system prompts for missing data. |
| **Exception scenarios** | Trigger: Calculation error due to invalid input.  Action: System asks for correct grades. |
| **Frequency of Use** | Low |

### Use Case Table 11: Note-Taking Feature

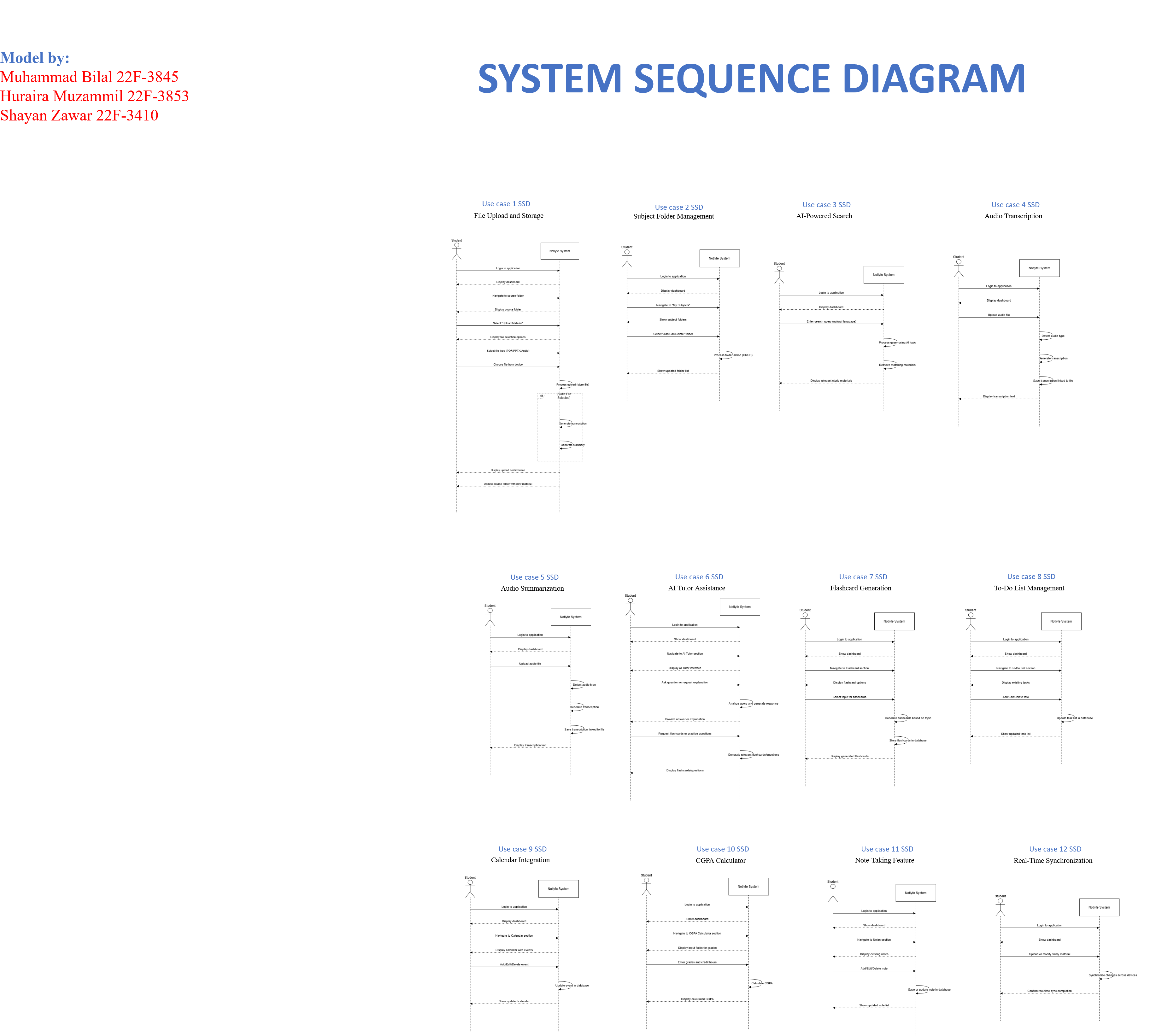
|  |  |
| --- | --- |
| **Field** | **Details** |
| **Designation** | FR-11 |
| **Name** | Note-Taking Feature |
| **Type** | Primary |
| **Source** | M. Bilal (domain expert for educational systems) |
| **Person responsible** | M. Bilal |
| **Description** | The system allows users to create, edit, and organize notes. |
| **Trigger event** | User creates or edits a note. |
| **Actors** | Student, Notlyfe System, Supabase Database |
| **Pre-conditions** | User is logged in. |
| **Post-conditions** | Notes are saved and organized in Supabase. |
| **Result** | User can access their notes from any device. |
| **Main scenario** | 1. User creates a new note.  2. User writes and formats the note.  3. System saves the note in the database. |
| **Alternative scenarios** | 4a. If note is empty, system prompts to add content. |
| **Exception scenarios** | Trigger: Save fails due to network issue.  Action: System retries or saves locally. |
| **Frequency of Use** | High |

### Use Case Table 12: Real-Time Synchronization

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Designation** | FR-12 |
| **Name** | Real-Time Synchronization |
| **Type** | Primary |
| **Source** | M. Bilal (domain expert for educational systems) |
| **Person responsible** | M. Bilal |
| **Description** | The system synchronizes data (files, notes, tasks) across devices in real-time using Supabase. |
| **Trigger event** | Data is updated on one device. |
| **Actors** | Student, Notlyfe System, Supabase Database |
| **Pre-conditions** | User is logged in and has internet access. |
| **Post-conditions** | Data is consistent across all devices. |
| **Result** | User can switch devices without losing data. |
| **Main scenario** | 1. User updates data on one device.  2. System syncs the update to Supabase.  3. Other devices receive the update. |
| **Alternative scenarios** | 4a. If sync is delayed, system queues updates. |
| **Exception scenarios** | Trigger: Sync conflict occurs.  Action: System resolves conflict or notifies user. |
| **Frequency of Use** | High |

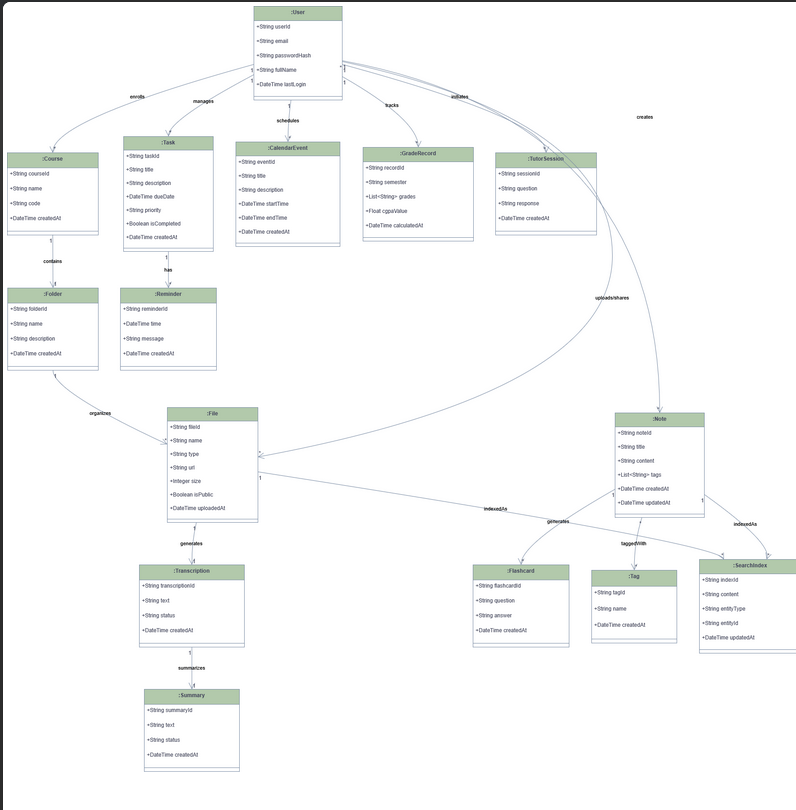
# 5.Diagrams

## 5.1 System Sequence Diagram

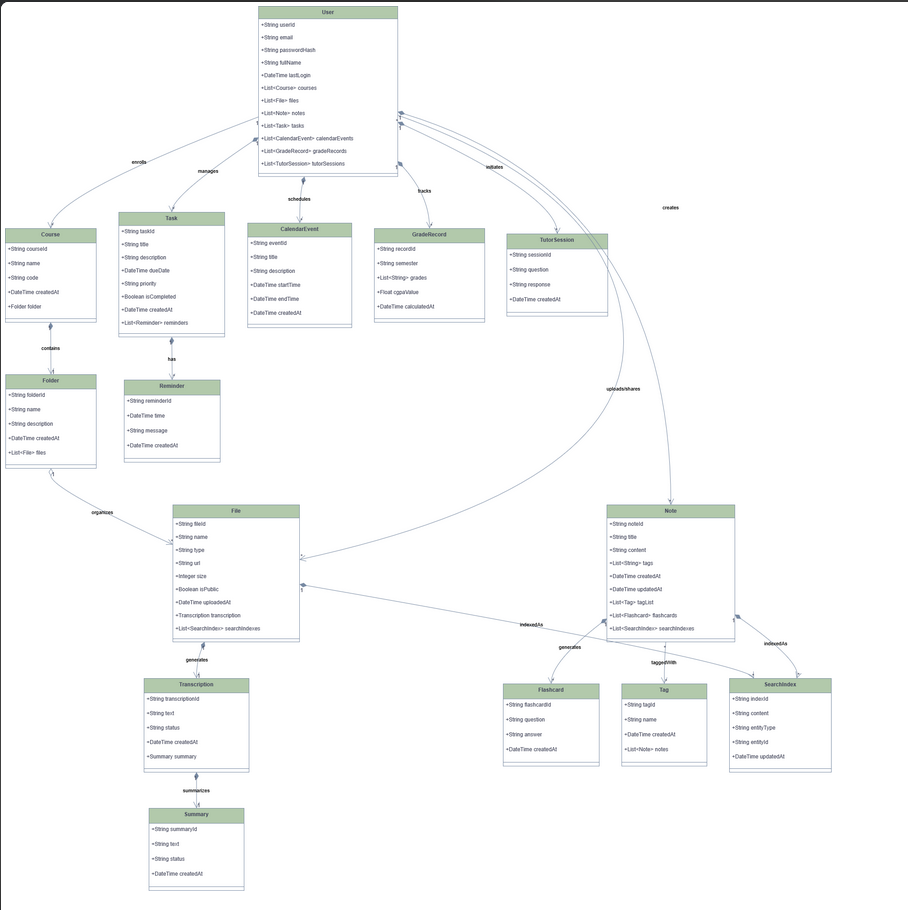


# 

## 5.2 Domain Diagram

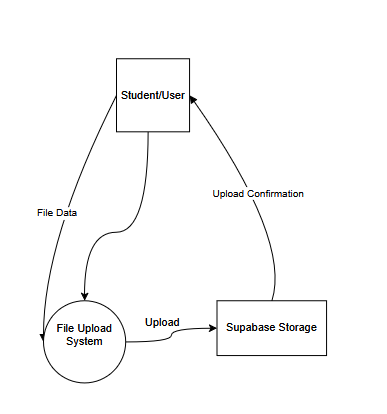


## 5.3 Class diagram

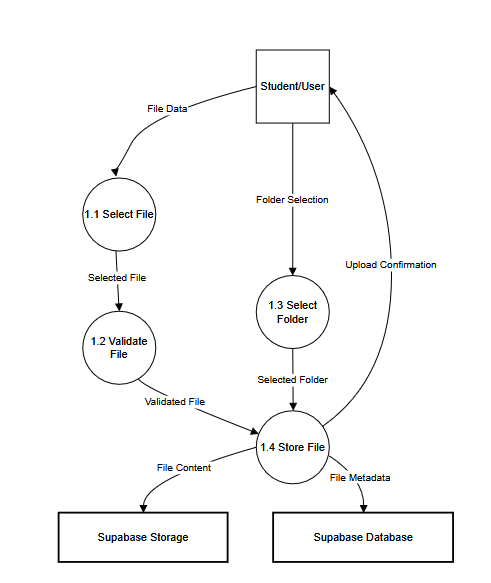


## 5.4 Data flow diagram Youndon Notation

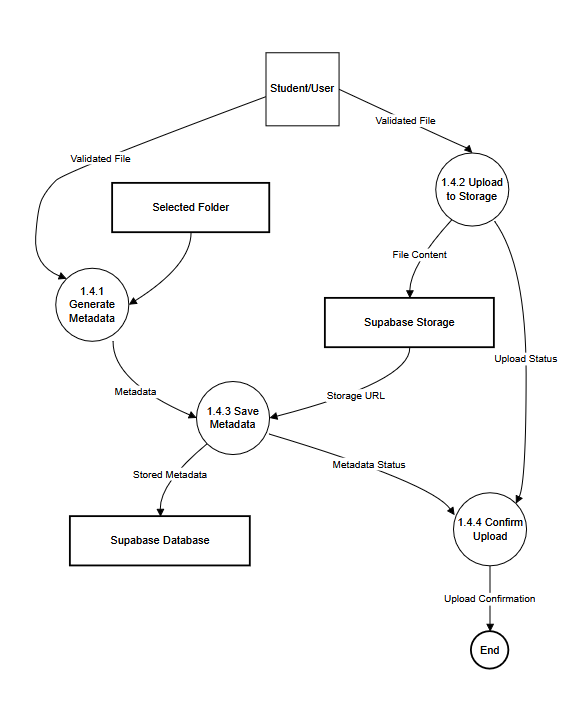
### Level 0



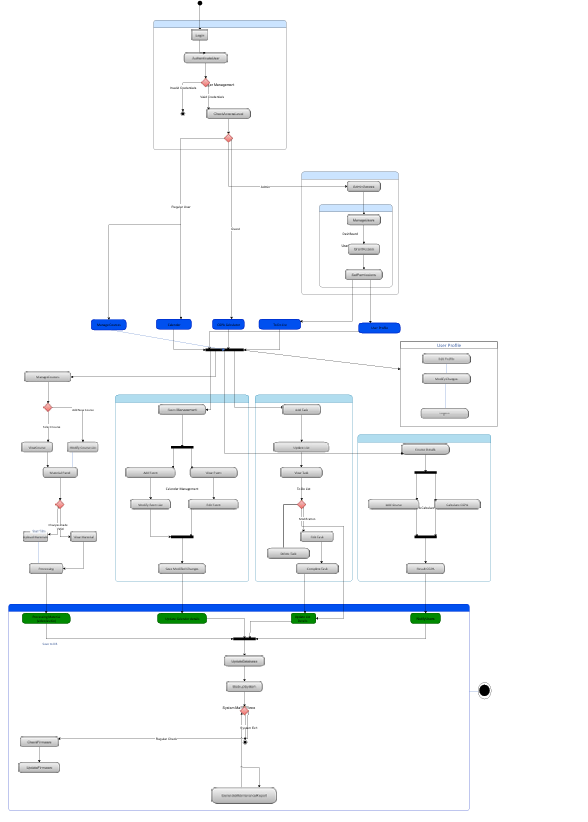
### Level 1



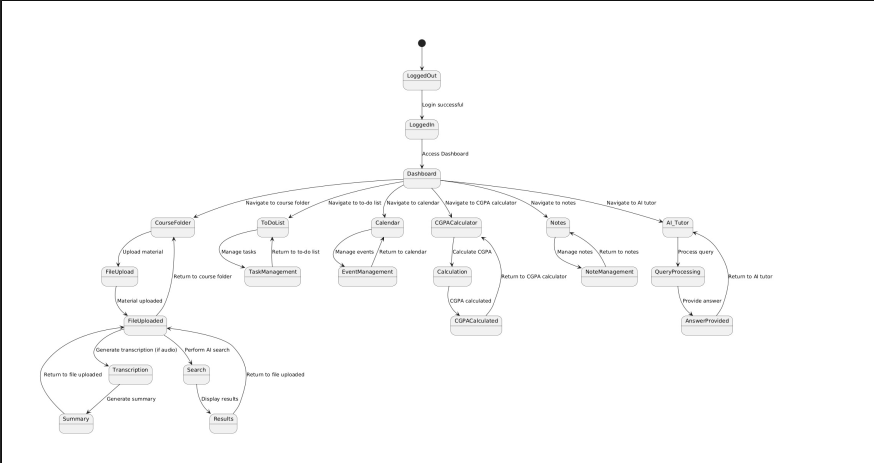
### Level 2



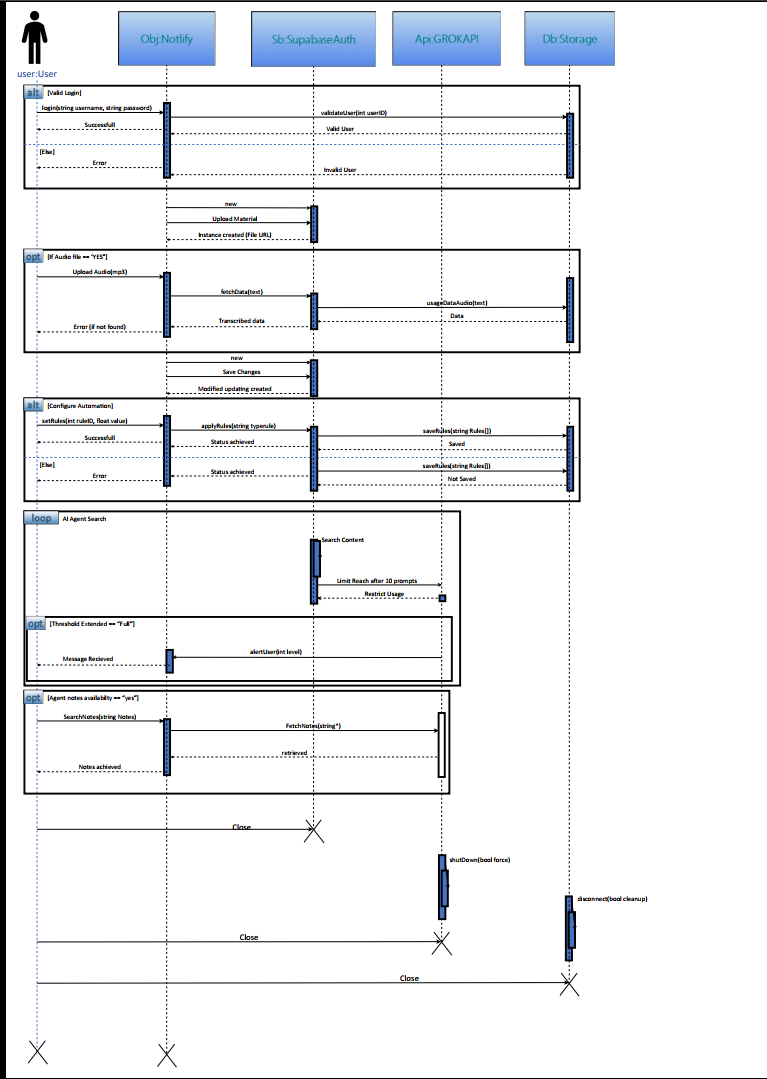
## 5.5 Activity Diagram



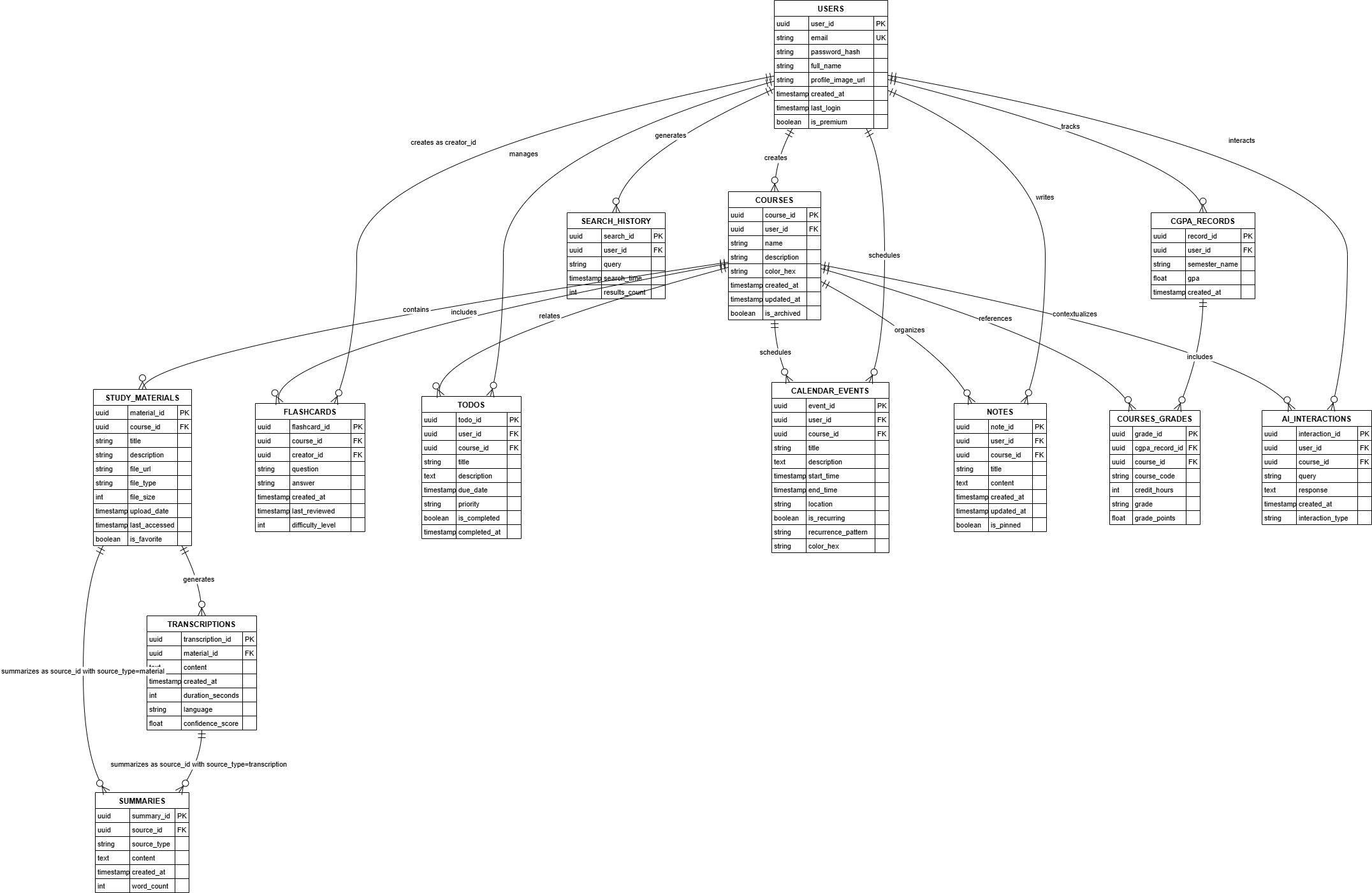
## 5.6 State Machine diagram



## 5.7 Sequence diagram



## 5.8 ER Diagram



End of file