Software Requirements Specification for Course Buddy

Team #5, Overwatch League
Jingyao, Qin
Qianni, Wang
Qiang, Gao
Chenwei, Song
Shuting, Shi

November 3, 2023

Contents

Revision History

Date	Version	Notes
Oct 11, 2023	Revision 0	First draft of SRS
Nov 1, 2023	Revision 0.1	Security update

1 Purpose of the Project

1.1 User Business

The users would be students in high school to college institutes all over the world dealing with multiple courses.

1.2 Goals of the Project

We aim to develop a tool to help students integrate multiple course information with their personal schedules to generate a customized study plan. The plan should dynamically self-adjust according to the user's preference and study efficiency so that students can finish their deliverables before the deadline at their own pace with minimal pressure.

2 Stakeholders

2.1 Client

N/A

2.2 Customer

The primary customers are students across various levels of education, from high school to college institutions. These students are in need of a comprehensive solution to manage their overwhelming schoolwork, deadlines, and tests.

2.3 Other Stakeholders

- **Educational Institutions**: Schools, colleges, and universities that may use this app as part of their academic toolkit.
- **Parents**: Concerned about their child's academic performance and well-being.
- **Educational Researchers**: Those interested in studying the effects of task management and its correlation with academic success.

2.4 Hands-On Users of the Project

Students: Using the application to manage their studies and academic tasks, connecting through the app's social network component for collaborative study sessions.

2.5 User Participation

- **Initial User Involvement** Users will be initially involved in providing insights regarding their needs and preferences through surveys and interviews.
- **User Acceptance Testing** Selected users will be involved in the beta testing phase to gather feedback and identify any issues or areas of improvement before the final release.
- **Continuous Feedback** Once the app is launched, user participation will continue through feedback mechanisms built into the app, allowing continuous improvement of features and user experience.
- **Community Engagement** Users can participate in community forums to discuss features, share tips, and support each other in using the app effectively

2.6 Maintenance Users and Service Technicians

- Maintenance users and service technicians are responsible for the maintenance and optimization of the application, ensuring all features and components function as intended.
- They will monitor application performance, resolve any arising issues or bugs, and implement necessary updates and enhancements.
- Service technicians will manage the training pipeline for machine learning algorithms, ensuring the models are accurate, reliable, and up-to-date.
- Regular maintenance are scheduled to ensure the application's consistent performance. Critical issues are addressed immediately to minimize any disruption to users.

3 Mandated Constraints

3.1 Solution Constraints

The development of a fully functional product is required to be finished by February 5 when the Revision 0 Demonstration is scheduled.

3.2 Implementation Environment of the Current System

The project implementation would be done through *VS Code* in the beginning and converted into *Github Codespace* once set up gets finished to ensure consistent compiling performance among group members.

3.3 Partner or Collaborative Applications

Our system would import event data from and export generated study plans to popular calendar applications like *Google Calendar* and *Outlook Calendar*.

3.4 Off-the-Shelf Software

3.4.1 StudySchedule.org

StudySchedule is a free scheduling software dedicated to generating customized daily schedules for students to study for MCAT. They would ask the user to set up an account, pick study material from their MCAT resource library, and take a questionnaire on time constraints and pace preference. Students could view their progress and make adjustments as they wish.

3.4.2 Taskade AI Genrator

Taskade is a powerful team project management tool. The component *Taskade AI* is capable of generating tasks for given project topics and creating checklists, project plans, and calendar schedules. *Taskade AI* is also capable of summarising *PDF* files.

3.5 Anticipated Workplace Environment

Our web-based app is anticipated to be compatible with mainstream browsers including *Chrome, Safari, Microsoft Edge*, and *Firefox* on the latest version of *Windows, Linux* and *macOS*.

3.6 Schedule Constraints

Each member of our team would devote 8 hours per week to work on the project making a total of 40 hours per week.

3.7 Budget Constraints

The monetary expenditure for the entire project could not exceed $750.

3.8 Enterprise Constraints

N/A

4 Naming Conventions and Terminology

4.1 Glossary of All Terms, Including Acronyms, Used by Stakeholders involved in the Project

- **UI:** User Interface - the space where interactions between humans and machines occur.
- **ML:** Machine Learning - a field of artificial intelligence that uses statistical techniques to give computer systems the ability to "learn" from data.
- **Pipeline (in ML):** A series of automated processes that allow for the streamlining of data from ingestion to processing, transformation, training, and evaluation in machine learning models.
- **API:** Application Programming Interface - a set of tools and definitions used to implement software applications.

- **HTTP:** Hypertext Transfer Protocol - an application protocol used for transferring hypermedia documents, such as HTML. It is the foundation of any data exchange on the Web.
- **Database:** Centralized collection of data, which can be stored, accessed, and managed easily.
- **Pomodoro Timer:** A time management method that uses a timer to break down work into intervals, traditionally 25 minutes in length, separated by short breaks.

4.2 Table of Units

Throughout this document, SI (Système International d'Unités) is employed as the unit system. In addition to the basic units, several derived units are used as described below. For each unit, the symbol is given followed by a description of the unit and the SI name.

symbol	unit	SI
`s`	time	second
`h`	time	hour

4.3 Symbolic Parameters

parameter	value	unit	description
MIN_EDUCATION	high school	N/A	the minimum level of education
MIN_UNDERSTAND%	95	N/A	the minimum percentage of testers who can understand among all testers
MAX_TRIAL_TIME	1200	s	the maximum allowed trial time
MIN_TESTER_NUM	20	N/A	the minimum number of testers needed
MAX_BAD_GRAMMAR	0	N/A	the maximum occurrence of grammar mistakes allowed
MAX_OFFENSIVE	0	N/A	the maximum occurrence of offensive messages allowed
MAX_COLOR_AMBIGUOUS	0	N/A	the maximum occurrence of indistinguishable color combinations allowed
MIN_OPERABLE%	95	N/A	the minimum percentage of system being operable
MIN_API_SUCCESS%	95	N/A	the minimum percentage of successful API calls
MIN_REGRESSION_PASS%	100	N/A	the minimum percentage of successful API calls
MAX_RESPONSE_TIME	24	h	The maximum issue resolve response time
MIN_LANGUAGE	5	N/A	The minimum number of languages that can be translated
MAX_SUPPORT_STEP	5	N/A	The maximum steps needed for asking for support
NUMBER_Of_MEMBERS	5	N/A	The number of group members working on this project
WORK_HOURS	40	h	The total number of hours spent on this project per week

5 Relevant Facts And Assumptions

5.1 Relevant Facts

Manually reading through multiple course outlines, inputting all the deliverable information into the calendar, and crunch time has always been an inefficient part of academic life. Students wish to have a seamless tool integrating deadline management into their everyday lives without risking overdue penalties.

5.2 Business Rules

N/A

5.3 Assumptions

- Course outlines are available as *PDF* files.
- Users are using *Windows, Linux* and *macOS* operating systems.
- Users have access to browsers including *Chrome, Safari, Microsoft Edge*, and *Firefox*.
- Users have access to the internet.
- Users have basic technical skills including typing, downloading, and uploading files.
- Users are using a mainstream calendar application: *Google Calendar, Outlook Calendar, Calendar*.
- Users have a relatively stable weekly schedule.

6 The Scope of the Work

6.1 The Current Situation

Currently, students across various educational levels face significant challenges in managing their academic tasks and schedules effectively due to the overwhelming schoolwork. Teachers and educational institutions also seek

innovative solutions to foster management skills among students and enhance their learning experiences. The existing solutions are often not comprehensive, lacking intelligent task prioritization, progress visualization, and effective time management tools.

6.2 The Context of the Work

The Smart Study Helper App is designed to bridge this gap by providing a user-friendly platform that combines automated task generation, intelligent task prioritization, progress visualization, and various other features. The development of this application intends to improve students' academic performance and mental well-being by reducing stress and enhancing learning experiences.

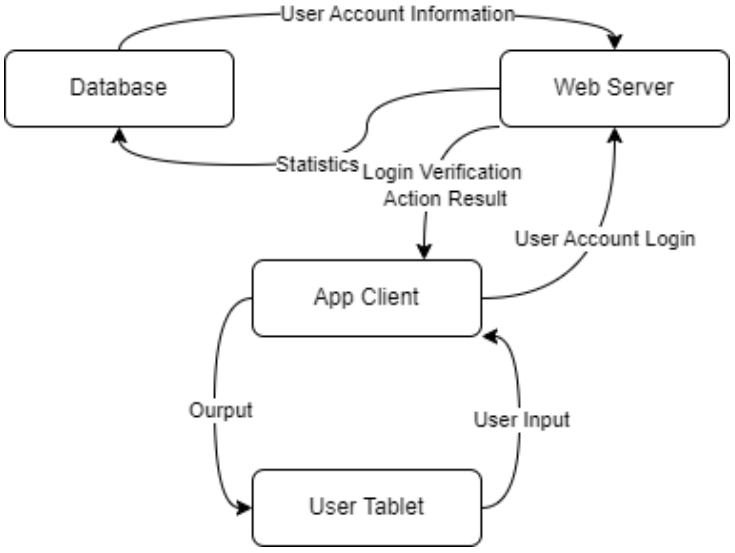

Figure 1: Context Diagram

6.3 Specifying a Business Use Case (BUC)

- **BUC Name:** Integrated Study Management
- **Goal:** To provide a comprehensive solution that allows students to manage their academic tasks, schedules, and study sessions effectively.

- **Actors:** Students, Educational Institutions, Teachers.
- **Preconditions:** User registration and course information uploaded or inputted.
- **Postconditions:** Enhanced student learning experiences, improved academic performance, reduced stress levels.
- **Main Success Scenario:** Students effectively use the app to manage their study tasks, leading to improved academic outcomes and reduced stress. Educational institutions and teachers observe enhanced student management skills and learning experiences.
- **Extensions:** Development of additional features based on user's feedback, improvement of machine learning model, larger user base.

7 Business Data Model and Data Dictionary

7.1 Business Data Model

7.2 Data Dictionary

7.2.1 Users

Users		
Attribute	userName	userPwd
Description	Unique identifier of a user	The password of an account
Type	`VARCHAR(50)`	`VARCHAR(50)`
Allowed Values	N/A	N/A
Default Value	N/A	N/A
Constraints	`PRIMARY KEY, NOT NULL`	`NOT NULL CHECK (CHAR_LENGTH(userPwd) >8)`
Source	User input when setting up the account	User input when setting up the account
Usage	Authentication	Authentication
Attribute	pInterval	
Description	preferred Pomodoro interval	

Type	`INT`	
Allowed Values	N/A	
Default Value	N/A	
Constraints	`NOT NULL CHECK (pInterval >= 0 AND pInterval <= 300)`	
Source	User input when setting up the account	
Usage	Scheduling	

7.2.2 Courses

Courses		
Attribute	subject	courseCode
Description	The subject of a course	The code of a course
Type	`VARCHAR(16)`	`VARCHAR(16)`
Allowed Values	N/A	N/A
Default Value	N/A	N/A
Constraints	`NOT NULL, PART OF PRIMARY KEY`	`NOT NULL, PART OF PRIMARY KEY`
Source	Extracted from the course outline	Extracted from the course outline
Usage	Record course information	Record course information
Attribute	courseId	
Description	The unique identifier of a course	
Type	`VARCHAR(16)`	
Allowed Values	N/A	
Default Value	N/A	
Constraints	`PRIMARY KEY`	
Source	Uniquely generated when a course outline is uploaded	
Usage	Record course information	

7.2.3 Tasks

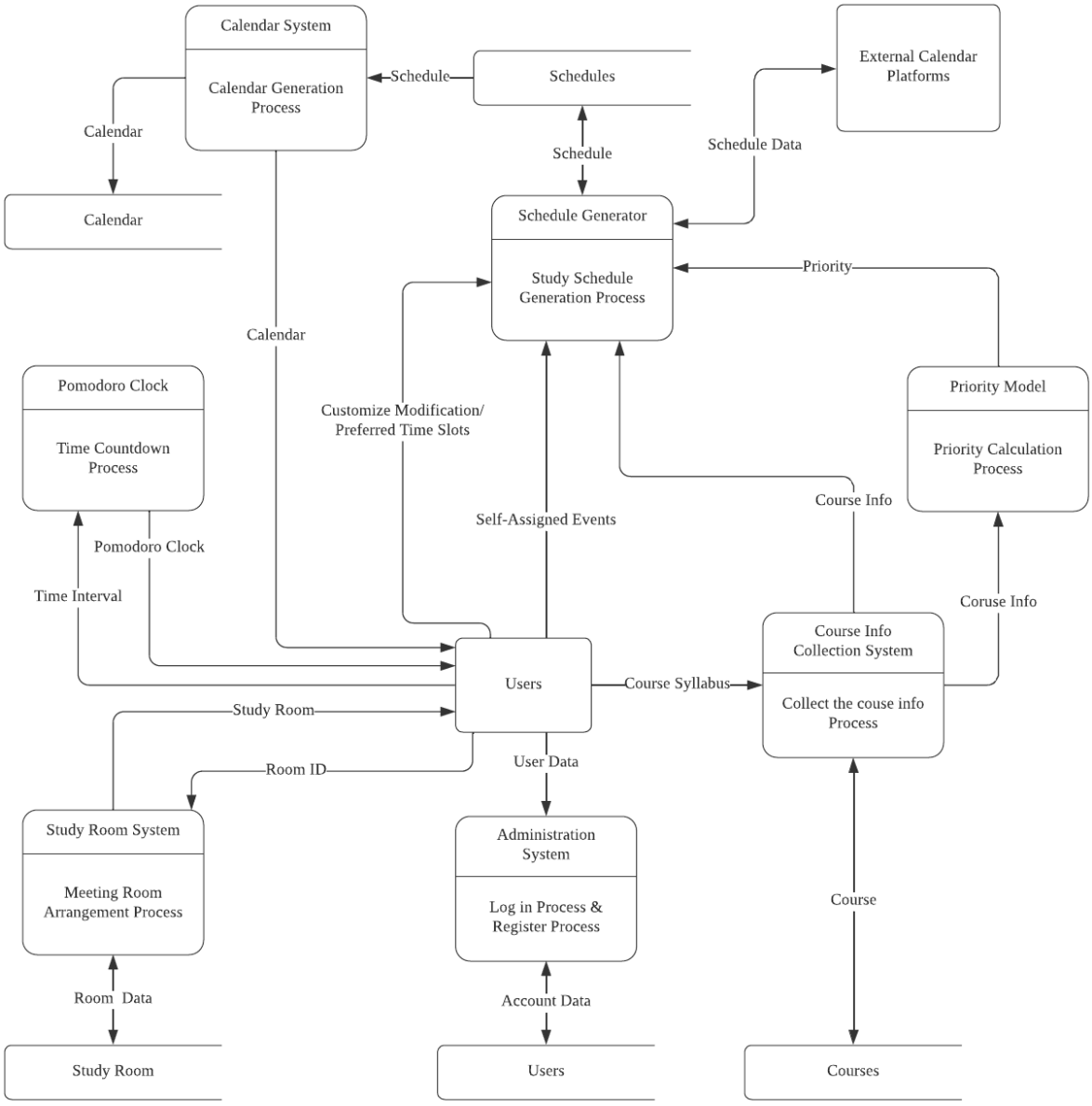

Figure 2: Data Flow Diagram

Tasks		
Attribute	weight	taskType
Description	The percentage weight associated with a task	The type of a task
Type	`DECIMAL(10,2)`	`INT`

Allowed Values	N/A	`0 - QUIZ`
		`1 - ASSIGNMENT`
		`2 - PRESENTATION`
		`3 - MIDTERM`
		`4 - EXAM`
		`5 - REPORT`
		`6 - OTHER`

Default Value	0	0
Constraints	`NOT NULL CHECK (weight >= 0 AND weight <= 100)`	`NOT NULL`
Source	Extracted from the course outline	Extracted from the course outline
Usage	Record course information	Record course information
Attribute	subject	courseCode
Description	The subject of a course	The code of a course
Type	`VARCHAR(16)`	`VARCHAR(16)`
Allowed Values	N/A	N/A
Default Value	N/A	N/A
Constraints	`NOT NULL`	`NOT NULL`
Source	Extracted from the course outline	Extracted from the course outline
Usage	Record course information	Record course information
Attribute	courseId	priority
Description	The unique identifier of a course	The priority ranking of a task
Type	`VARCHAR(16)`	INT
Allowed Values	N/A	N/A
Default Value	N/A	0
Constraints	`NOT NULL`	`NOT NULL`
Source	Uniquely generated when a course outline is uploaded	Calculated with `weight`, `deadline` and taskType

Usage	Record course information	Record priority of a task.
Attribute	deadline	
Description	The deadline of a task	
Type	`DATE`	
Allowed Values	N/A	
Default Value	N/A	
Constraints	`NOT NULL`	
Source	Extracted from the course outline	
Usage	Record course information	

8 The Scope of the Product

8.1 Product Boundary

The Smart Study Helper App aims to serve as a comprehensive solution for students to manage their study schedules and tasks effectively. Its boundary extends from user registration, task generation, and prioritization, to progress visualization and study planning. It will interact with external calendar services and will allow users to collaborate with peers through its social network component. However, the app's boundary does not extend to managing non-academic tasks or any other aspects of a user's daily life not related to their study.

8.2 Product Use Case Table

Use Case Name	Primary Actor	Description
User Registration	Student	Allows new users to create an account
Upload Syllabus	Student	Enables users to upload course syllabus in PDF format
Task Generation	Application	Automatically generates tasks based on uploaded syllabus

Task Prioritization	Application	Prioritizes tasks based on ML algorithms
Progress Visualization	Student	Users can view the status of their tasks
Facial Recognition	Application	Detect users' stress levels by recognizing users' face
Classmates Lookup	Student	Users can look up online classmates who register the same course
Connected Learning	Student	Users can connect with their friends or classmates to study together
User Login	Student	Allows users to log into their accounts using username and password
Friend List Management	Student	Supports friend list for sending/rejecting/accepting friend requests
Export to Other Calendars	Application, Student	Exports event and schedule data to third-party calendar apps
Estimate Task Duration	Application	Calculates the estimated time for each task from course outlines

8.3 Individual Product Use Cases (PUC's)

8.3.1 User Registration

- **Goal:** Enable students to create a unique account.
- **Actors:** Student.
- **Preconditions:** Student has accessed the application or web.
- **Postconditions:** Student has an active account and can access the app's features.

Figure 3: Use Case Diagram

- **Main Flow:** Student enters required information, sets a password, and completes registration.

8.3.2 Upload Syllabus

- **Goal:** Allow students to upload their course syllabus.
- **Actors:** Student.
- **Preconditions:** Student is logged in.
- **Postconditions:** Syllabus is stored and ready for task generation.
- **Main Flow:** Student selects the course syllabus in PDF format and uploads it.

8.3.3 Task Generation

- **Goal:** Create tasks from the uploaded syllabus.
- **Actors:** Application.
- **Preconditions:** Syllabus has been uploaded.
- **Postconditions:** Tasks are generated and displayed to the student.
- **Main Flow:** Application processes the syllabus and generates relevant tasks.

8.3.4 Task Prioritization

- **Goal:** Rank tasks based on importance.
- **Actors:** Application.
- **Preconditions:** Tasks have been generated.
- **Postconditions:** Tasks are prioritized and presented in an organized manner.
- **Main Flow:** Application uses ML algorithms to analyze and prioritize tasks.

8.3.5 Progress Visualization

- **Goal:** Offer students a view of their task progress.
- **Actors:** Student.
- **Preconditions:** Tasks have been generated and prioritized.
- **Postconditions:** Student can visually assess task status.
- **Main Flow:** Student accesses a dashboard or progress section to view status of tasks.

8.3.6 Facial Recognition

- **Goal:** Identify student stress levels using facial cues.
- **Actors:** Application.
- **Preconditions:** Student grants camera access.
- **Postconditions:** App detects potential stress indicators.
- **Main Flow:** Application uses facial recognition technology to analyze student's face for stress signs.

8.3.7 Classmates Lookup

- **Goal:** Allow students to find classmates in the same course.
- **Actors:** Student.
- **Preconditions:** Student is registered in a course.
- **Postconditions:** Student views a list of classmates in the same course.
- **Main Flow:** Student accesses course section and views registered classmates.

8.3.8 Connected Learning

- **Goal:** Provide students with a platform to study collaboratively.
- **Actors:** Students.
- **Preconditions:** Students have active accounts.
- **Postconditions:** Students can connect and study together.
- **Main Flow:** A student sends a connection request to another student. Once accepted, they can access video chat with joint study tools and features.

8.3.9 User Login

- **Goal:** Allow students to log into their account.
- **Actors:** Student.
- **Preconditions:** Student has an active account.
- **Postconditions:** Student is logged in and can access personalized data.
- **Main Flow:** Student provides username and password; if credentials match, access is granted.

8.3.10 Friend List Management

- **Goal:** Manage friend connections within the application.
- **Actors:** Student.
- **Preconditions:** Student is logged in.
- **Postconditions:** Updated friend list status (friend request sent/received/accepted/rejected).
- **Main Flow:** Student navigates to the friend list section, sends/receives/accepts/rejects friend requests.

8.3.11 Export to Other Calendars

- **Goal:** Export event and schedule data to third-party calendar apps.
- **Actors:** Application, Student.
- **Preconditions:** Student has events or schedules in the application.
- **Postconditions:** Events and schedules are exported and viewable in third-party apps.
- **Main Flow:** With authentication, the student initiates export to a chosen calendar app; data is formatted and transferred.

8.3.12 Estimate Task Duration

- **Goal:** Calculate the estimated time needed for each task from course outlines.
- **Actors:** Application.
- **Preconditions:** Tasks have been extracted from course outlines.
- **Postconditions:** Each task has an associated time estimate.
- **Main Flow:** Application processes tasks from course outlines and estimates time requirements for each.

9 Functional Requirements

9.1 Authentication

FR1: The user could create an account with a username and a password.

Rationale: Users would need an account protected by a password to keep their schedule personal and private.

Priority: HIGH

FR2: The user could log in to their account by providing a corresponding username and password.

Rationale: Users would need to access their accounts.

Priority: HIGH

FR3: The user should be able to log out of the account.

Rationale: Users do not have to stay logged in.

Priority: HIGH

FR4: The user should have access to their scheduling information once logged in.

Rationale: User's scheduling information should be available to them.

Priority: HIGH

FR5: The system should provide a mechanism for users to retrieve their passwords in case they forget them.

Rationale: A user who has forgotten their password should be able to recover their account.

Priority: HIGH

9.2 User input

FR6: The user could upload multiple PDF files containing course outlines to the system.

Rationale: Multiple courses would have multiple course outlines.

Priority: HIGH

FR7: The system prompts users to choose their preferred study interval at which the system tends to allocate study sessions.

Rationale: The user would want to have their study session schedules at their preferred time.

Priority: MEDIUM

FR8: The user could change their preferred study interval.

Rationale: The user's preferred study interval could change over time.

Priority: MEDIUM

FR9: The system prompts users to set their preferred Pomodoro intervals.

Rationale: The user might have preferred Pomodoro intervals.

Priority: MEDIUM

FR10: The user could change their Pomodoro interval.

Rationale: The user's preferred Pomodoro intervals could change over time.

Priority: MEDIUM

FR11: The system supports a friend list allowing users to send friend requests.

Rationale: The user needs to send friend requests to add friends.

Priority: LOW

FR12: The user could accept an incoming friend request.

Rationale: The user could accept the friend request if they wish to befriend the request sender.

Priority: LOW

FR13: The user could reject an incoming friend request.

Rationale: The user could reject the friend request if they wish not to befriend the request sender.

Priority: LOW

FR14: The user could change the progress of each task

Rationale: The progress of each task should align with reality.

Priority: HIGH

FR15: On finishing each sub-task, the system should ask for the user's feedback on whether the pace is comfortable.

Rationale: The schedule needs feedback to adjust the pace.

Priority: HIGH

9.3 Data

FR16: The system could extract information including `courseName, taskType, taskName, weight, deadline` from the uploaded course outline PDF files.

Rationale: Course outlines would be provided as PDF files containing course information.

Priority: HIGH

FR17: The system could generate a list of tasks from uploaded PDFs.

Rationale: To generate a study plan, tasks to be finished are needed.

Priority: HIGH

FR18: The system could assign priority to each task generated.

Rationale: From priority the system could allocate adequate time to finish each task.

Priority: HIGH

FR19: The system could detect the user's attention level.

Rationale: The user's attention level is needed to allocate rest time.

Priority: MEDIUM

FR20: The system should notify the user when the user's attention level is low.

Rationale: The system would keep the user engaged in their study task.

Priority: MEDIUM

FR21: The user should be able to view the progress of each task.

Rationale: The user would want to see their progress.

Priority: HIGH

FR22: With authentication, the system could import event and schedule data from other calendar apps including Calendar, Outlook, and Google Calendar.

Rationale: Study session allocation would consider avoiding conflicts with existing events.

Priority: MEDIUM

FR23: With authentication, the system could export event and schedule data to other calendar apps including Calendar, Outlook, and Google Calendar.

Rationale: The user would want to use their own calendar of choice.

Priority: MEDIUM

FR24: The user should be able to export their study plan as a PDF.

Rationale: The user could want to have an overview of their study plan offline.

Priority: LOW

9.4 Scheduling

FR25: The system could calculate the estimated time needed to finish each task.

Rationale: Total required tie is needed to decide the amount of subtasks needed.

Priority: HIGH

FR26: The user could regenerate a study plan accordingly once a task's progress is changed by the user.

Rationale: Previously generated study plans could be outdated.

Priority: HIGH

FR27: The estimated time needed to finish each task would dynamically adjust based on user feedback on past task completion progress.

Rationale: Estimated time needed may not be perfect.

Priority: HIGH

FR28: The system could prioritize tasks extracted based on `taskType`, `weight` and `deadline`.

Rationale: The system would ensure high-priority tasks are finished on time.

Priority: HIGH

FR29: With data containing task information, preferred study time, schedule, and Pomodoro intervals gathered the system could generate a detailed study plan containing multiple sub-tasks for each task in the course outline available in-app and export it to other calendars.

Rationale: A detailed study plan contains all the sub-tasks and their settings.

Priority: HIGH

FR30: The system would have a Pomodoro clock ready for each sub-task.

Rationale: A Pomodoro clock would help the user balance their study and rest.

Priority: MEDIUM

FR31: With a detailed study plan, the system could pair friends from the friend list with similar study plans to have video-based online study sessions.

Rationale: Friends with similar tasks at similar times could collaborate and inspire each other.

Priority: LOW

10 Look and Feel Requirements

10.1 Appearance Requirements

The application's design should be user-friendly and intuitive. Key elements to consider include:

AR1: Clear typography: Text should be legible at all standard screen resolutions and sizes.

AR2: Color scheme: The color palette should be eye-catching but not overwhelming. Calming and neutral tones should be used that helps to focus.

AR3: Icons and graphics: Visual aids should be recognizable and easy to learn.

AR4: Layout: The design should be responsive, ensuring usability across devices of varying screen sizes, from smartphones to tablets to desktop monitors.

10.2 Style Requirements

The style of the application should make the application easy and clear to use, providing a sense of community for students. Aspects to focus on include:

STR1: Navigation: Menus and navigation tools should be logically organized and easy to access.

STR2: Consistency: Elements like buttons, text fields, and icons should maintain a consistent design throughout the application.

STTR3: Interactivity: Interactive elements like buttons or dropdowns should provide feedback, indicating appropriate responsiveness.

STR4: Accessibility: The design should cater to all users, including those with disabilities and elder adults. Features like text-to-speech and adjustable font sizes should be considered.

STR5: Animations: Any animations used should be subtle and not distractive so users can focus more on the main content.

11 Usability and Humanity Requirements

11.1 Ease of Use Requirements

UHR1: The navigation must be intuitive to use by users with `MIN_EDUCATION` education background.

Fit Criterion: MIN_UNDERSTAND% of users with at least MIN_EDUCATION of education could navigate through functions within MAX_TRIAL_TIME of exploring.

- U: Total number of users
- E: The set of users with at least MIN_EDUCATION
- N: The number of users who can navigate through functions within MAX_TRIAL_TIME

$$N \geq \left(\frac{\texttt{MIN\_UNDERSTAND\%}}{100} \right) \cdot |E|$$

11.2 Personalization and Internationalization Requirements

N/A

11.3 Learning Requirements

UHR2: The system must be understood by users within MAX_TRIAL_TIME of exploring.

Fit Criterion: MIN_UNDERSTAND% of users with at least MIN_EDUCATION of education could understand the system within MAX_TRIAL_TIME of exploring.

- U: Total number of users
- E: The set of users with at least MIN_EDUCATION
- N: The number of users who can navigate through functions within MAX_TRIAL_TIME

$$N \geq \left(\frac{\texttt{MIN\_UNDERSTAND\%}}{100} \right) \cdot |E|$$

11.4 Understandability and Politeness Requirements

UHR3: The language in the app must be grammatically correct MIN_grammar% of the time.

Fit Criterion: A group of MIN_TESTER_NUM users could find at most MAX_BAD_GRAMMAR of grammar mistakes.

UHR4: The language in the app must be non-offensive.

Fit Criterion: A group of MIN_TESTER_NUM users could find no more than MAX_OFFENSIVE of offensive messages.

- U: Total number of users testing the system
- O: The number of offensive messages found by a group of users
- G: The group of at least MIN_TESTER_NUM users

$$O \leq \texttt{MAX\_OFFENSIVE}$$

11.5 Accessibility Requirements

UHR5: Color combinations used in the interface must be distinguished by users with color blindness.

Fit Criterion: A group of MIN_TESTER_NUM users could find at most MAX_COLOR_AMBIGUOUS of indistinguishable color combinations in the UI filtered with Color Oracle.

- C: The number of indistinguishable color combinations found by a group of users, filtered with Color Oracle
- G: The group of at least MIN_TESTER_NUM users

$$C \leq \texttt{MAX\_COLOR\_AMBIGUOUS}$$

12 Performance Requirements

12.1 Speed and Latency Requirements

SLR1: Major actions, such as uploading syllabuses, generating tasks, and prioritizing tasks, should be completed in a timely manner.

Fit Criteria: During testing under normal load conditions, major actions are executed within a 2-second threshold.

12.2 Safety-Critical Requirements

SCR1: All data, especially sensitive academic information, must be securely encrypted to ensure protection against unauthorized access.

Fit Criteria: Throughout security testing, no incidents of data breaches or unauthorized data accesses are observed.

12.3 Precision or Accuracy Requirements

PAR1: ML-based features, particularly task prioritization, must achieve a high degree of accuracy.

Fit Criteria: When tested against predefined scenarios, ML algorithms show an accuracy rate of 90% in task categorization and prioritization.

12.4 Robustness or Fault-Tolerance Requirements

RFTR1: The application must demonstrate resilience against unexpected inputs or actions and provide means of efficient data recovery.

12.5 Capacity Requirements

CR1: The system should be robust enough to manage a large number of concurrent users.

Fit Criteria: During load testing, the application manages the equivalent load of 10,000 users and data pertaining to 1 million courses without any performance issues or system crashes.

CR2: The system should store vast quantities of course data without any compromise in performance.

Fit Criteria: During load testing, the application manages the equivalent load of 1 million courses without any performance issues or system crashes.

12.6 Scalability or Extensibility Requirements

SER1: The design and architecture of the application should support ease of modification and the addition of new features.

12.7 Longevity Requirements

N/A

13 Operational and Environmental Requirements

13.1 Expected Physical Environment

OER1: The system must be operable under the same physical environment that the desktop computer running it is operable.

Fit Criterion: When the machine is operable, the system is operable at least `MIN_OPERABLE%` of time.

- T: Total time the machine is operable
- S: Total time the system is operable when the machine is operable

$$\frac{S}{T} \geq \frac{\texttt{MIN\_OPERABLE\%}}{100}$$

13.2 Requirements for Interfacing with Adjacent Systems

OER2: The system could interface with calendar APIs when called.

Fit Criterion: At least `MIN_API_SUCCESS%` of requests made to supported calendar APIs are successful.

- R: Total number of requests made to the supported calendar APIs
- S: Number of successful requests made to the supported calendar APIs

$$\frac{S}{R} \geq \frac{\texttt{MIN\_API\_SUCCESS\%}}{100}$$

13.3 Productization Requirements

N/A

13.4 Release Requirements

OER3: The released version must pass all known regression tests.

Fit Criterion: The released version must pass `MIN_REGRESSION_PASS%` of regression tests.

- T: Total number of regression tests conducted on the released version
- P: Number of regression tests passed by the released version

$$\frac{P}{T} \geq \frac{\texttt{MIN\_REGRESSION\_PASS\%}}{100}$$

14 Maintainability and Support Requirements

14.1 Maintenance Requirements

MR1: The bugs and issues should be addressed within a response time corresponding to their severity.
Fit Criteria: If a bug significantly impacts the regular user experience, the response time for resolution does not exceed `MAX_RESPONSE_TIME` hours.

MR2: The updates and patches should be delivered periodically.
Fit Criteria: A feature update, aligned with user preferences and market trends, can be released each season.

MR3: The application should be documented comprehensively, with a user manual, technical report, and clear code comments for further navigation and maintenance.
Fit Criteria: A user manual can provide tutorials for full product feature utilization. A technical report should be updated at least once every season's maintenance. Clear code comments should be provided to facilitate developers' navigation and maintenance.

14.2 Supportability Requirements

SPR1: Users shall have easy access to a helpdesk for support.
Fit Criteria: Users are able to access the email contact, phone contact, and chatbot to address common user questions in `MAX_SUPPORT_STEP` steps.

SPR2: The response time for offering support should be on time corresponding to the support type.
Fit Criteria: For critical functionality disasters, the support should be delivered within `MAX_RESPONSE_TIME` hours.

SPR3: The application should have a mechanism for collecting users' feedback for product continuous improvement.
Fit Criteria: Users are able to provide feedback when they want.

14.3 Adaptability Requirements

ADR1: The product should be compatible with common systems and *API*s to facilitate seamless integration and data exchange.
Fit Criteria: The product can interact with the *Google Calendar API* through explicit *HTTP* calls or using the *Google Client Libraries*.

ADR2: The product should implement modular architecture, allowing for the addition or replacement of components without causing issues.
Fit Criteria: The products' independent modules can interact with

each other through de-coupled interfaces.

15 Security Requirements

15.1 Access Requirements

SR1: The system is accessible only if the correct combo of username and password is provided.

Fit Criterion: An error message suggesting an incorrect password or username is displayed.

SR2: Users could not access other users' data.

Fit Criterion: Only data linked to the currently logged-in account could be displayed.

SR3: Ensure data encryption during data transfers to prevent unauthorized access.

Fit Criteria: Data being transferred should be encrypted using industry-standard algorithms, with no plain-text data leaks detected.

15.2 Integrity Requirements

SR4: No Unauthorised entity could modify the database.

Fit Criterion: A group of `MIN_TESTER_NUM` unauthorised users could not modify data.

SR5: Have a strict role-based access control to prevent unauthorized data manipulation.

Fit Criteria: Different user roles should have differing access levels, with no unauthorized data access incidents.

15.3 Privacy Requirements

SR6: The system will not release user information to a third party.

Fit Criterion: The system does not provide user information to a third party.

15.4 Audit Requirements

SR7: Maintain an audit log of all activities within the application for traceability and accountability.

Fit Criteria: All user and system activities should be logged with time stamps and relevant meta-data.

15.5 Immunity Requirements

SR8: Provide regular security patches and updates to the software to rectify known vulnerabilities.

Fit Criteria: No known vulnerability should persist in the system for more than a month without a patch.

SR9: The system should protect authentication data from brute force attacks.

Fit Criteria: Restriction after a certain number of failed login attempts; option for the user to unlock account via email or phone.

16 Cultural Requirements

16.1 Cultural Requirements

CR1: The application shall be easily translatable into various languages without causing ambiguity in meaning.
Fit Criteria: The application can be translated into at least `MIN_LANGUAGE` languages(Spanish, French, Chinese, Italian and German).

CR2: The product shall avoid elements such as confusing icons, symbols, or offensive images that might cause controversy in different cultural environments.
Fit Criteria: Users are able to feel comfortable and respected.

17 Compliance Requirements

17.1 Legal Requirements

CPR1: The system must comply with local laws and regulations.

Fit Criterion: The system does not violate local laws or regulations.

17.2 Standards Compliance Requirements

CPR2: The code must comply with *Flack8* coding standards.

Fit Criterion: Merged code passes *Flake8* linter workflow check.

18 Open Issues

The application features should identify the dependency relationships for better organizing the development process.

19 Off-the-Shelf Solutions

19.1 Ready-Made Products

The off-the-shelf product should be compatible with our application currently using technology stack, infrastructure and operating system (*Windows* or *IOS*, excluding *Linux*). Also, the selection of ready-made products shall be based on cost-effectiveness and alignment with the project's functionality requirements.

19.2 Reusable Components

For text and PDF extraction, reusable components could include robust text parsing algorithms, and document structure analysis tools, which can be used across various document types. In the context of modelling, reusable components could consist of machine learning models that have been trained on diverse datasets. Furthermore, in the field of facial recognition, reusable components may encompass pre-trained facial detection models and privacy

protection mechanisms. Also, some existing website extensions or components can also be reused to assist the application development.

19.3 Products That Can Be Copied

The configuration location and integrated module should be clearly defined in the documentation, also developer should maintain records of the updated versions of the copied source product.

20 New Problems

20.1 Effects on the Current Environment

- The implementation of a new system may change the way students interact with existing tools and platforms. If the new tool enables integration with existing popular scheduling platforms, users may stop using other scheduling tools. Some users may need to adapt to these changes, and this transition needs to be handled carefully to ensure a smooth user experience.
- The data security requirements of the new tool and the user access control may require changes to the current security infrastructure. Any potential impact on existing security protocols should be fully assessed, and measures taken to mitigate risks.
- The new system may change user workflows and procedures. If it automates and prioritizes tasks, this may impact the way they plan and manage assignment completion and course scheduling. Understanding and responding to these changes as early as possible is critical to ensure smooth operation.

20.2 Effects on the Installed Systems

This section specifies the interfaces between the new system and existing systems or components.

- **Interface with Google Calendar**: The system should integrate with Google Calendar to synchronize and visualize task deadlines. The interface will involve certification, data exchange, and event management.
- **Interface with Outlook Calendar**: Similar to Google Calendar, the system shall integrate with Outlook Calendar to synchronize events. The interface will involve certification and event management.
- **Interface with Machine Learning Server**: The system relies on machine learning algorithms to prioritize tasks. This interface includes sending data to the machine learning server, processing suggestions, and integrating them into the user interface.
- **Interface to connect with users**: To facilitate collaborative learning, the system should enable users to connect with their peers. This interface includes user authentication, data exchange, and video chat integration.
- **Interface with videoconferencing hardware** Users will use webcams, microphones, and speakers for video chat during collaborative learning. This interface is required to access these hardware components and manage live video conferencing.

20.3 Potential User Problems

- **User Confusion**: The addition of machine learning-driven task prioritization, facial recognition, and online collaborative learning session features may confuse or create resistance from some users who are unfamiliar with these concepts.

  Prevention and mitigation measures:

  - Develop a user-friendly onboarding process and provide extensive training materials to ensure users can easily understand and use the new features.
  - Provide customization options that allow users to customize the system to their preferences. This flexibility will help users take better control of their experience.

- **Privacy Concerns**: Users may have privacy concerns, especially with attention-monitoring features such as facial recognition. They may be concerned that their facial data will be collected and used.

  Prevention and Mitigation Measures:

  - Privacy concerns are addressed by implementing strong data protection measures. Users will be informed of what their data will be used for and how it will be processed and will be able to opt out of certain features.

- **Technical Issues**: Technical issues or system incompatibilities may lead to adverse reactions, such as issues related to platform support or software bugs.

  Prevention and Mitigation Measures:

  - Rigorous testing and quality assurance are conducted to detect and correct technical issues before they affect users. Updates and bug fixes will be performed regularly to ensure system stability.

20.4 Limitations in the Anticipated Implementation Environment That May Inhibit the New Product

- The planned servers may not have enough processing power or storage capacity to handle the expected growth in users and data volume.
- Available network bandwidth may not be sufficient to support real-time videoconferencing for collaborative learning sessions, leading to potential performance issues.
- Implementing system integration with external calendaring platforms (e.g., Google Calendar, Outlook) takes into account that these platforms may have limitations or constraints that affect the quality of the integration.
- Hardware for facial recognition may not be readily available or may not meet the accuracy requirements for detecting the user's level of attention.

- The implementation environment may be subject to specific data privacy regulations that may affect the collection and storage of user data.

20.5 Follow-Up Problems

This section anticipates potential challenges, unintended consequences, and limitations that the project may encounter during development and implementation.

- The implementation of new features or technologies in the system may inadvertently result in non-compliance with existing laws and regulations, such as user privacy protection aspect. The project team will conduct regular assessments. Any necessary changes will be made to address potential legal issues.

21 Tasks

21.1 Project Planning

See Project Scheduling Section in Development Plan.

21.2 Planning of the Development Phases

See Project Scheduling Section in Development Plan.

22 Migration to the New Product

The project is a stand-alone application, not an upgrade or replacement of an existing product. There is no need to involve a product migration. The Migration to the New Product section is not needed.

23 Costs

The following are some of the major cost items that may need to be considered:

- **Cloud services**: using a cloud infrastructure (e.g. *Amazon AWS*, *Microsoft Azure*, or *Google Cloud*) requires consideration of the cost of using cloud services.
- **Server hosting**: using an independent server, server hosting, and maintenance costs need to be considered.
- **Database**: database storage and access costs.
- **Data Backup**: regular data backup and storage costs.
- **Working Time**: takes NUMBER_Of_MEMBERS group members WORK_HOURS a week.

24 User Documentation and Training

24.1 User Documentation Requirements

UDR1: A user manual that covers all functionalities of the website with step-by-step instructions should be provided in a digital format. The manual should be written in an easily understandable language, accompanied by illustrative diagrams.

UDR2: User manual should be updated with each release to reflect any changes made.

UDR3: A help/support channel should be provided so that users can ask questions and give feedback which will be taken into consideration for future developments and improvements.

24.2 Training Requirements

UTR1: A short tutorial should be provided if it is the user's first time using the website.

UTR2: A training video should be provided with demos on how to use major features of the website.

UTR3: Training videos should be added or updated with each release to reflect any changes.

25 Waiting Room

- Add a dark mode support for the website
- Add a colour blind support for the website
- Add sound effects to the website to make it more user friendly and interesting
- Implement a feature where the users are able to integrate with other popular music apps like Spotify to play their favourite playlists while studying
- Implement a badges or achievements system such as points for completing tasks which can make the use of the website more engaging and rewarding
- Provide data analysis and insight on how the user performs within a period of time and provide feedback on how to improve their way of study
- Add support to other platforms such as mobile applications and desktop applications

26 Ideas for Solution

26.1 Idea 1

Requirement: FR6, The user could upload multiple PDF files containing course outlines to the system.
Potential solution: We could use a *Python* web framework *Flask* to set up

a web app which supports accessing and uploading multiple files with PDF extensions.
Advantages: This will allow users to upload their PDF easily and more securely, and with no additional effort required for developers to implement and test these features, as *Flask* has already taken care of it for us.
Disadvantages: Using *Flask* to support this feature doesn't present any foreseeable disadvantages in the near future.

26.2 Idea 2

Requirement: FR18, The system could assign priority to each task generated.
Potential solution: A inference pipeline could be set up to give inference on the unclassified tasks using type of task, weight, and deadline as the inputs. The inference pipeline could use a trained model of task priority classification.
Advantages: Setting an inference pipeline could be more flexible and could potentially give better results than implementing an fixed algorithm for giving predictions. Several steps for the inference pipeline could be constructed and it will be easier to make changes once it gets set up, simply adding, changing or removing steps when there is a future requirement.
Disadvantages: There will be an overload of defining each step and connecting them.

26.3 Idea 3

Requirement: FR21, The user should be able to view the progress of each task.
Potential solution: Could implement a cloud based solution for holding and retrieving data of task, time, etc. A potential choice could be using *AWS S3*
Advantages: This would allow users to check their progress of task and time in a more available, reliable, and well-managed service. The data can be well secured and can be accessed in many ways for example using *AWS S3*, *AWS console*, *AWS Command Line*, providing multiple ways for developers to implement

Disadvantages: There is a cost associated with it when using third party cloud services

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

1. What knowledge and skills will the team collectively need to acquire to successfully complete this capstone project? Examples of possible knowledge to acquire include domain specific knowledge from the domain of your application, or software engineering knowledge, mechatronics knowledge or computer science knowledge. Skills may be related to technology, or writing, or presentation, or team management, etc. You should look to identify at least one item for each team member.

   The following knowledge and skills will the team need to acquire to successfully complete this capstone project:

   - GitHub features such as issue tracker, using workflows/*Github Actions*, setting up *Dev Containers*, etc.
   - Machine learning and neural network knowledge for the developing and optimizing task priority classification models
   - Machine learning libraries and frameworks such as *pandas*, *scikit-learn*, and *Keras*
   - *Python* web framework for web development such *Streamlit* and *Flask*
   - Agile methodologies such as sprint planning and for team management, efficient collaborations and constant delivery, which is important for incremental and iterative development like this project

2. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?

Knowledge or Skills	**Approaches**	**Assigned Team Member**	**Reason**
GitHub	Use *ChatGPT*, Google, watch online tutorials, or ask supervisor for help	Shuting, Shi	Strong interest in using *GitHub* for project management, previous experience with other projects.
Machine learning and neural network	Use *ChatGPT*, *Google*, watch online tutorials, and read research papers	Qiang, Gao	Strong interest in ML and neural network, watched many online tutorials and read many related books.
Machine learning libraries and framework	Use *ChatGPT*, *Google*, watch online tutorials, or ask supervisor for help	Qianni, Wang	Experience with many ML projects where these libraries are being used in AI programs and previous co-op work terms.
Python web framework	Use *ChatGPT*, *Google*, watch online tutorials, or ask supervisor for help	Chenwei, Song	Experience in web development in previous co-op work terms.
Agile methodologies	Use *ChatGPT*, *Google*, and watch online tutorials	Jingyao, Qin	Experience with Agile mindset in previous co-op work terms, strong interest in project management as the team lead.