SRS Document

Group 2

S.A.A.Y.J

Centennial College

COMP225 – 012

Table of Contents

Table of Contents 2
Part A 3
    Deliverable #1 3
        Section 1: 3
        Section 2: 10
        Section 3: 14
    Deliverable #2 17
    Deliverable #3 18
    Deliverable #4 21
Part B 32
    Deliverable #1 32
    Deliverable #2 37
    Deliverable #3 41
    Deliverable #4 42
    Deliverable #5 43
    Deliverable #6 44
Part C 48
    Deliverable 1 48
        Diagram 1: Notetaking Tool 48
        Diagram 2: Task Manager Tool 49
    Deliverable 2 50
    Deliverable 4 51

Part A

Deliverable #1

Section 1:

1.1 Purpose

The software discussed in this SRS document, SAAYJ, is a new software that will perform the functions of note taking and scheduling apps with additional features. This software will implement features that enhance productivity and provide more customizable options. Some of these options include highly customizable note templates and privacy settings. The initial release of the software in this version will be version 1.0.0. Successful implementation of SAAYJ would empower users to streamline tasks, manage time more effectively, and accomplish more with ease.

Major challenges the SAAJY would solve for its target user base:

- Students Challenge 1: they dropped too many unrelated notes in one place; hence they cannot find notes for a particular subject when doing revision.
    - SAAYJ's solution: solving this issue by providing filtering, sorting, and searching features between an abundant number of notes and files.
- Students Challenge 2: they lack perseverance in studying or focusing.
    - SAAYJ's solution: by offering achievement awards, users will be motivated to complete the task within a time goal.
- Office workers Challenge 1: they spend too much time on particular events (e.g. meetings), which leads to the risk of missing or being late for the next scheduled event.
    - SAAYJ's solution: with our optional locational widget being on, our system will send notifications to remind users to move to the next scheduled event and record how much extensive time is spent in events to generate advice for users to manage time more efficiently.
- Office workers Challenge 2: when they want to share files with co-workers, current notetaking apps seldom have the feature of editing together.
    - SAAYJ's solution: by offering collaborative task management, users can edit files together simultaneously. Hence, the productivity of combining ideas can be improved.
- General user Challenge 1: available notetaking and scheduling software do not effectively serve their users, but users do not want to switch because of the effort associated with learning new software.

    - SAAYJ’s solution: there will be a connection with other applications that allow for data to be converted over to SAAYJ. In addition, there will be some options available to create experiences that best suit what users are familiar and comfortable with.
- General user Challenge 2: notetaking apps and scheduling apps on the current market have limited functionality, hence users are forced to compromise.
    - SAAYJ’s solution: SAAYJ will have the functionality of notetaking and scheduling apps along with additional features. Major features include items related to note organization and collaboration. The features available in the software packaged into one app will enhance individual productivity.

1.2 Document Conventions

Acronym	Description
API	Application programming interface
IaaS	Infrastructure as a service
IT	Information technology
OS	Operating system
PaaS	Platform as a service
QA	Quality assurance
SDK	Software development kit
SRS	Software requirement specification
SSO	Single sign-on
UX	User experience

1.3 Intended Audience and Reading Suggestions

The following list describes the roles of the intended audience reading this SRS document:

- Software development team members: Anyone involved in creating the software product described in this document. (Examples listed below.)
    - Developers: Members who code and test the software.
    - Testers: Members who test the software in any stage of development, including developers and QA team members.
    - Project Managers: Management members who manage (organize, provide direction, track deadlines, etc.) software development teams.
- IT team
    - Members acquiring hardware and setting up software configurations to develop the software described in this document.
    - Members that will manage the software through administrator privileges.
- UX team: Members providing feedback on the design of frontend features seen by end users.

- Marketing and analysts: Members familiar with software projects who analyze this project from a business perspective; observing costs, profitability, etc.

1.4 Project Scope

Perspective	In Scope	Out of Scope
Features	• Notetaking and scheduling • Reminders and notifications alerts • Connection with other users • Categorization • Filtering, sorting and searching • Security options and encryption • Awards achievement • Other additional features that are discussed	• Custom spelling and grammar check • Automatic content generation • Augmented reality and virtual reality interaction • Customizable 3D avatars for users • Analysis of note content • Real-time language translation
Interface	• Cross-platform compatibility • Options to use a large variety of default templates or highly customized templates • Responsive design on supported mobile and computer devices • Dark mode/light mode options • Options for handwritten interface	• Features and settings unintuitive to users • Access from unsupported devices (e.g. smart TVs, smartwatches, Samsung fridge) • Too many animated graphics which may dramatically lower hardware performance • Different user experiences between different platforms • Voice commands for hands-free commands (e.g. changing between files and tabs)
Integration	• Integration with popular external software applications (i.e. Google Calendar, Outlook, OneNote)	• Integration with unpopular external software applications • Integration with social media for automatic posting of notes

	• Integration with cloud storage (i.e. Google Drive, OneDrive) • Importing files and images from external applications • Exporting contents created in our application as PDF documents • Synchronization between notes and schedules • Integration with payment solution to take paid subscription payments	• Integration with translating applications • Instantaneously updates data to external applications • Inserting an advertisement (e.g. link routing to external websites)

Project Scope Statement	
Title	SAAYJ: notetaking plus scheduling app
Project Justification	Many people use digital devices to take notes (some for studies and some for working purposes) and manage their schedules, SAAYJ is a mobile application or a web application that combines both features all in one, plus some useful functions that general applications do not have.
Project Scope Description	By creating an account and subscribing to the service of SAAYJ, users can enjoy all initial free components in the application, including notetaking, scheduling, customizing application appearance, sharing with other users, and collecting achievement by meeting time targets.
Project Objective	With user-friendly functionalities, SAAYJ will be users' daily companion on the journey to a more effective and more fulfilling life.
Requirements	- To optimize users' productivity and efficiency in academic, work or personal matters, SAAYJ provides **access from any mobile devices**, including smartphones and tablets, online or offline. Hence, users can read their notes and modify their schedules anytime, anywhere. - To promote users' perseverance in studying or completing tasks, SAAYJ offers **awards achievements** when users reach certain goals. Users can also share their gained achievements with other users and decorate the customized application appearance or user profile with the badges. - To organize notes and schedules better, users can use **filtering and sorting** features to group certain notes to different topics and hold multiple variations of schedule (i.e. class schedule, work schedule, daily schedule). - SAAYJ provides an **optional location widget** to help users keep track of when they should leave for their scheduled event. When users stay

	in a particular place for longer than estimated, SAAYJ will send a notification to remind users to leave for the next scheduled event. - For the convenience of sharing users' notes or schedules with their friends or co-workers, users can **connect with other users** by adjusting privacy settings. Users can choose what information they want to share, and whom they want to share with. - Paid items will be offered for users who paid for **monthly subscriptions**, including complex templates, advanced encryption options, larger storage on the cloud, etc.
Assumptions	- Time targets will be updated daily, weekly, or monthly, according to the type of goals. - When users turn on location features continuously for over 1 month, a system message will pop up to confirm whether users want SAAYJ to keep accessing their location. - Whenever there is a new feature added, a window "What's New" will pop up when users open our application.
Deliverables	- A single system including all components - Detailed user guide on introduction of all features and tools

1.5 References

Adamant. (2023, April 11). *What are the differences between renting and buying a server?* https://adamant.ua/en/blog/what-are-the-differences-between-renting-and-buying-a-server

Anderson, B. Nicholson, B. (2022, June 12). *SQL cs. NoSQL databases: what's the difference?* IBM. https://www.ibm.com/blog/sql-vs-nosql/

AppBrain. (n.d.). *Top Android OS versions*. Retrieved on January 28, 2024, from https://www.appbrain.com/stats/top-android-sdk-versions

Ali, S. (2022, March 24). *Best Technology stack used for mobile app development*. Medium. https://medium.com/predict/best-technology-stack-used-for-mobile-app-development-b62e4f2e296c

Ashley. (2021, May 20). *Online shopping system context ciagram*. Wondershare EdrawMax. https://www.edrawmax.com/templates/1004644/

AWS cloud products. (n.d.). Amazon Web Services. Retrieved on January 28, 2024, from https://aws.amazon.com/products/?aws-products-all.sort-by=item.additionalFields.productNameLowercase&aws-products-all.sort-

order=asc&awsf.re%3AInvent=*all&awsf.Free%20Tier%20Type=*all&awsf.tech-category=*all

Azure products. (n.d.). Microsoft. Retrieved on January 28, 2024, from https://azure.microsoft.com/en-us/products

Cabrera, I. (2022, April 1). *What is a user flow diagram and how to create one?* Venngage. https://venngage.com/blog/user-flow-diagram/

Can I use... (n.d.). *ECMAScript 2015 (ES6)*. Rerieved on January 28, 2024, from https://caniuse.com/?search=es6

Galiya, J. (2023, December 17). *Diving deep into iOS programming languages: advanced techniques for iOS app development*. Radixweb. https://radixweb.com/blog/best-programming-languages-for-ios-app-development

Gillis, A. S., & Botelho, B. (2023, March). *MongoDB*. TechTarget https://www.techtarget.com/searchdatamanagement/definition/MongoDB

Google Cloud. (n.d.). *Google Cloud Platform services summary*. Retrieved on January 28, 2024, from https://cloud.google.com/terms/services

Howarth, J. (2023, December 6). iPhone vs Android User Stats (2024 Data). *Exploding Topics*. https://explodingtopics.com/blog/iphone-android-users

IBM. (n.d.). *IaaS vs. PaaS vs. SaaS*. https://www.ibm.com/topics/iaas-paas-saas

IBM Cloud products. (n.d.). IBM. Retrieved on January 28, 2024, from https://www.ibm.com/cloud/products

Watts, S. & Raza, M. (2019, June 15). *Saas vs PaaS vs IaaS: what's the difference & how to choose*. BMC. https://www.bmc.com/blogs/saas-vs-paas-vs-iaas-whats-the-difference-and-how-to-choose/

Indeed Editorial Team. (2022, June 24). *What is out of scope and how to avoid it in your project*. Indeed. https://www.indeed.com/career-advice/career-development/out-of-scope

Jilg, D. (2023, October 1). *iOS versions market share in October 2023*. Telemetry Deck. https://telemetrydeck.com/blog/ios-market-share-10-23/

Mendes, A. (2023, October 10). *What's the best tech stack for your mobile app in 2024?* Imaginary Cloud. https://www.imaginarycloud.com/blog/techstack-mobile-app/

Microsoft. (n.d.). *Azure Cosmos DB*. Retrieved on January 28, 2024, from https://azure.microsoft.com/en-ca/products/cosmos-db

Miguel, P. G. (2024, January). *Guide to the 23 best cloud service providers in 2024*. The CTO. Retrieved on January 28, 2024, from https://thectoclub.com/tools/best-cloud-service-providers/

MongoDB. (n.d.). *NoSQL vs SQL Databases*. https://www.mongodb.com/nosql-explained/nosql-vs-sql

Lucidchart. (n.d.). *How to make a user flow diagram*. https://www.lucidchart.com/blog/how-to-make-a-user-flow-diagram

Kher, K. (2020, December 15). *LAMP vs MERN vs MEAN stack*. Medium. https://medium.com/@kartikkher777/lamp-vs-mern-vs-mean-stack-1a0fa7b01c43#:~:text=This%20is%20because%20the%20LAMP,was%20created%20by%20Google%20Inc

Oberlo. (n.d.). *Most popular web browsers in 2023*. https://www.oberlo.com/statistics/browser-market-share

ProjectPro. (2024, January 19). https://www.projectpro.io/article/aws-vs-azure-who-is-the-big-winner-in-the-cloud-war/401

Similarweb. (n.d.). *Top browsers market share*. Retrieved on January 28, 2024, from https://www.similarweb.com/browsers/

Spoiala, C. (2015, April 2). *Cloud offering: comparison between IaaS, PaaS, SaaS, BaaS*. Assist Software. https://assist-software.net/blog/cloud-offering-comparison-between-iaas-paas-saas-baas

Statcounter. (n.d.). *Android version market share worldwide*. Retrieved January 28, 2024, from https://gs.statcounter.com/os-version-market-share/android

StatCounter. (n.d.) *Browser market share worldwide*. Retrieved on January 28, 2024, from https://gs.statcounter.com/browser-market-share

StatCounter. (n.d.). *iOS version market share worldwide*. Retrieved on January 28, 2024, from https://gs.statcounter.com/ios-version-market-share/

StatCounter. (n.d.). *Mobile operating system market share worldwide*. Retrieved on January 28, 2024, from https://gs.statcounter.com/os-market-share/mobile/worldwide/

SysGen. (2023, October 31). *The pros and cons of cloud vs. in-house servers*. https://sysgen.ca/cloud-vs-in-house-servers/

TechTarget Contributor. (2023, February). *Android Oreo*. TechTarget. https://www.techtarget.com/searchmobilecomputing/definition/Android-Oreo

T. N. (2023, October 21). *Project Scope Statement Template*. Techno-PM. https://www.techno-pm.com/blogs/project-starter/project-scope-statement-template

Visual Paradigm Online. (n.d.). *What is a system context diagram?* https://online.visual-paradigm.com/knowledge/system-context-diagram/what-is-system-context-diagram/

W3Schools. (n.d.). *JavaScript versions*. Retrieved on January 28, 2024, from https://www.w3schools.com/js/js_versions.asp

Section 2:

2.1 Product Perspective

The software described in this SRS document, SAAYJ, is a new standalone software. This software also would include options for users to connect to popular features of applications through those applications' APIs.

2.2 Product Features (Functions)

The following list describes key features of SAAYJ:

- Notetaking
- Calendar/scheduling
- Connections between different users
- Awards achievements for meeting targets (e.g. study time)
- Highly customizable application appearance (frontend)
- Increased privacy and safety options for users (compared to other similar applications)
- Data storage
    - Option for mobile application to store data locally (not on the cloud)
    - Option to store data on the cloud (through mobile or web browser application with an internet connection)
- Notifications alerts
- Connections to other popular applications' features (through their APIs)

2.3 User Classes and Characteristics

The software described in this document, SAAYJ, will have internal members of the organization and external end users interact with it.

Internal members:

- Content moderators
- IT team members:
    - Cybersecurity
    - Administrators
    - Customer service representatives

External end users:

- Students (grade 7 to post-secondary)
- People taking notes for work
    - Office meetings and other settings
    - Other environments where notetaking on mobile devices is permitted
- People making lists for personal errands

2.4 Operating Environment

SAAYJ will operate on client devices through mobile and web browser applications. The browser application should use responsive web design for mobile and PCs to provide users with more flexible options for how they can interact with the software. These client applications will connect to backend servers to perform certain functions if required.

Frontend

The target consumer base for SAAYJ is users using mobile devices, primarily smartphones. This is because the major features of this software, the biggest one being notetaking, must be accessible in a variety of physical settings. The added convenience of using other features (like customizing templates and sharing items) easily through mobile devices would make this software stand out from similar software created by competitors. The web application of this software provides users with additional ways to conveniently use the software, such as from a PC interface.

Client device applications used for SAAYJ:

- Android
    - Version 8+
    - Uses of Android Studio environment with Android SDK and framework
    - Uses Java and Koltin languages
- iOS
    - Version 15+
    - Uses Apple’s Xcode environment with their iOS SDK and UIKit framework
    - Uses Swift and Objective-C languages

- Desktop (not part of initial release)

Based on data from Statcounter (2024) and Exploding Topics (2023), a large majority of mobile users globally, over 98%, use either Android or iOS; the percentage for each OS varies depending on the region, but the total between Android and iOS users remains nearly constant. Data from Statcounter (2024) and AppBrain (2024) show that over 82% of Android users use version 8 or above. TechTarget (2023) states that Android 8 was released in 2017; this means Android versions approximately seven years or newer will be compatible with SAAYJ. According to Jilg (2023) on TelemetryDeck, over 99% of iOS users use versions 15 or above; this is why SAAYJ will be available on iOS 15+.

Web browser applications used for SAAYJ:

- Google Chrome version 51+
- Safari version 10+
- Microsoft Edge version 14+
- Mozilla Firefox version 52+
- Opera (not part of the initial release)

Based on data from Statcounter (2024), Similarweb (2024) and Oberlo (2024), these browsers were selected because they have the largest market share values globally; over 90% between these browsers. Although the ranking for these browsers varies with different regions, they are always the most used according to Statcounter and Similarweb. The browser versions selected are based on information from W3Schools (2024) that list browser versions that started supporting ECMAScript 6 launched in 2015 (the closest major revision before this one was in 2009); this provides support for older browsers versions over eight years old.

Backend

The backend of the software described in this SRS document will use Microsoft Azure infrastructure and platform. Using Azure will allow the system to be highly scalable and allow us to easily upgrade hardware and software when desired without paying for more resources until it is needed. In addition, it helps financial planning because smaller recurring payments can be made compared to larger lump payments for owning the infrastructure. These benefits will also be useful for our project because of the level of uncertainty in the project's funding and when its market share will get large increases.

The tools available on the Azure platform will also be useful in our project. For example, tools for SSO functionality, cloud file management and NoSQL databases. The level of security provided with Azure services is sufficient for the majority of the software's intended user base. According to Miguel (2024) and ProjectPro (2024), Azure is also known for its performance in hybrid cloud solutions (when a company uses servers, they own along with server hosting services for a solution), which is useful when features that require servers directly owned and managed are required in the future.

2.5 Assumptions and Dependencies

There are limitations to the amount of planning that can be done for this software project because of the amount of information currently obtainable. Major assumptions and dependencies of the software are discussed below in this section; this does not include all possible assumptions and dependencies; more are likely to be discovered in the future.

- There is uncertainty about when users will start using SAAYJ, for example when they convert from other similar software.
    - The popularity of software cannot be accurately predicted; public opinion of it and its reputation cannot be predicted or controlled.
    - Most people who use notetaking and scheduling software already have apps they prefer.
        - There are a lot of software used to take notes and organize schedules like the one in this document, but with different available features.
        - Users are likely partially attached to apps they currently use because they are familiar with their interfaces and their past data is there.
        - Competition would be popular well-established apps, including ones pre-installed on smartphones.
    - The uncertainty regarding when usage increases makes it logical to use IaaS and PaaS cloud solutions to be able to scale up easily and reduce costs when usage is lower.
- Current funding for the software being developed has not been sufficiently defined.
    - This limits the number of features in the earlier versions of the software.
    - Makes it logical to use IaaS solutions to reduce upfront costs associated with owning and maintaining backend hardware.
- Users of cloud connected services for SAAYJ are required to consent to terms and conditions to prevent it from being used inappropriately. Primary examples:
    - Not modifying the software’s code.
    - Not intentionally engaging in activity that disrupts the software’s network
    - Consent to allow cloud content to be reviewed by moderators (with some user protections in place).
        - Removes liability for cases when the software is used for malicious or criminal purposes.
- The software being developed, SAAYJ, will **not be used to store or directly produce formal documents** (e.g. reports and manuals).

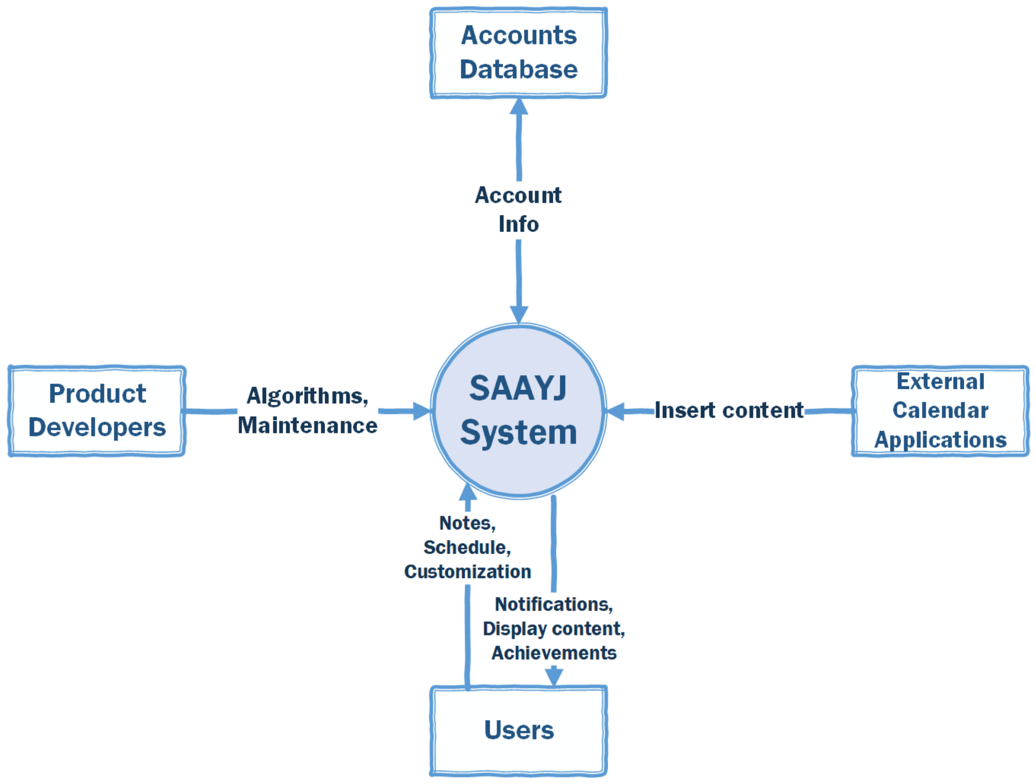
    - Cloud sync failures or delays are assumed to not cause catastrophic issues for individuals or organizations.
        - **Efforts will still be made to minimize cloud sync failures and delays to improve reliability.**
    - **The initial release of the software will include features to allow users to take notes, organize them in meaningful ways, and use them to collaborate with other users.**
- The initial release and updates will only include core features and features that are available with an unpaid subscription. Some features part of later updates will have limited availability or be unavailable without a paid subscription.
    - Features available for unpaid and paid subscriptions will both continue to be released after the initial versions of the software (may not be listed in this document).
    - Possible paid feature examples:
        - Tools for enterprise customers
        - Additional data encryption
        - Use of collaboration tools without restrictions
- For the base unpaid subscription packages for SAAYJ, it is assumed that the level of security used in most other software will be sufficient. **Users are assumed to not be storing highly confidential information (e.g. ID details and passwords).**
    - Enhanced security options will be available for additional payment after initial updates.
- The base features, which are included in the initial release and updates of SAAYJ, will not require hardware used for specialized or processing power intensive computations.
    - IaaS cloud server options should perform sufficiently for the initial versions of the software.
    - Customized servers owned and maintained by the company can be acquired in the future to use along with cloud IaaS when features that require them are released. (Most of these features would be ones associated with paid subscriptions.)

Section 3:

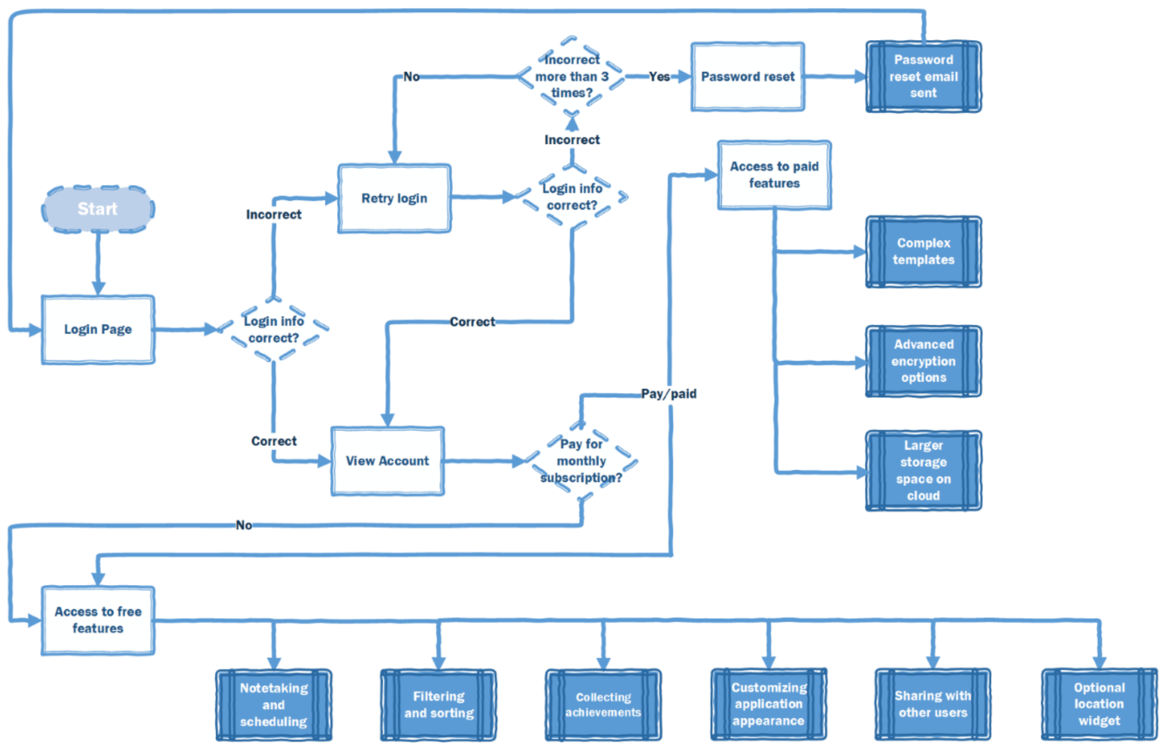
3.1 User Interfaces

- There will be a mobile interface that supports both platforms of Android and iOS, and a web interface that supports several famous web browsers, including Firefox, Google Chrome, Microsoft Edge, Apple Safari, and Opera. For backend, IaaS or PaaS server solutions and MEAN or MERN stacks will be used.

- System Context Diagram:


Accounts Database
Account Info
Product Developers
Algorithms, Maintenance
SAAYJ System
Insert content
External Calendar Applications
Notes, Schedule, Customization
Notifications, Display content, Achievements
Users

- Graphical User Interface:



3.2 Hardware Interfaces

The intended devices SAAYJ will be compatible with smartphones, laptops or desktops.

- Touchscreen, or mouse and keyboard, or digital pens: to take notes in a variety of ways.
- A camera: to take photos and insert them into notes.
- Microphone: for text-to-speech features.
- A scanner: to scan their handwritten physical notes into our system and convert them into digital notes.

3.3 Software Interfaces

- Calendars: SAAYJ can connect with other calendar applications, such as Google Calendar and Outlook, for easy conversion to SAAYJ and unified management.
- Files applications: users can copy content from documents from files applications to conveniently use the content in SAAYJ.
- Photo gallery: users can insert images from their galleries to customize the application interface.
- Payment solution: there must be a built-in solution to take payments from users (for paid subscriptions).

Deliverable #2

Stakeholder Name	Stakeholder Position	External / Internal	Stakeholder Contact Details	Operational / Executive	Interest (High, Medium, Low)
Ava Smith	Marketer	Internal	AvaSmith17@saayj.ca	Operational	High
Nora Edwards	Customer	External	Nora.Edwards@gmail.com	Operational	Low
John Ali	IT Team Member	Internal	John.Ali@saayj.ca	Operational	Medium
James Owen	Content Moderator	Internal	James.Owen@saayj.ca	Operational	Medium
Dean Miller	Investor	External	Dean.Miller@Zgroup.com	Executive	High
Mark Howard	Security Analyst	Internal	Mark.Howard@saayj.ca	Operational	Medium
Jay Davidson	Programmer	Internal	Jay.Davidson@saayj.ca	Operational	Medium
Peter Hendricks	Cyber Security	Internal	Peter.Hendricks@saayj.ca	Operational	Medium
Roxana Maria	Software Engineer	Internal	Roxana.Maria@saayj.ca	Executive	High
Andrew Philips	Database Administrator	Internal	Andrew.Philips@saayj.ca	Operational	High
Bill Ross	Digital Designer	Internal	Bill.Ross@saayj.ca	Operational	Medium
Larry Holmes	Customer Support	Internal	Larry.Holmes@saayj.ca	Operational	Low
Jessica Castillo	Software Tester	Internal	Jess.Castillo@saayj.ca	Operational	Medium
Antonio Smith	Project Manager	Internal	Antonio.Smith@saayj.ca	Executive	High
James Rodriguez	Data Analyst	Internal	James.Rodriguez@saayj.ca	Operational	Medium

Deliverable #3

Interview Questions		
Questions	Stakeholder Position	Answer
Do you have any preferences or rules regarding the user interface's design?	Digital Designer	We mainly use a simplistic design; however, we offer many different themes in the settings for the user's preference.
How are important bugs ranked, and what role can developers play in quickly resolving issues that are found?	Tester	The severity of a bug determines its priority. Developers may help by working together on bug triage, swiftly addressing important issues, and giving thorough bug reports.
How is team collaboration facilitated, and how can developers ensure effective communication?	Project Manager	We hold frequent meetings and use platforms like Slack. By contributing actively and giving frequent updates, developers will guarantee effective communication.
What security precautions must be taken to protect user information and the program itself?	Security Analyst	Security measures include authentication, authorization, encryption, logging, application security testing, and bot testing. ensuring the confidentiality and integrity of user data.
How will user feedback be gathered, and how will updates or modifications be implemented?	Tester	We have a dedicated website showcasing our product where we announce upcoming updates and have users share their comments. Updates will follow an iterative process with regular releases.
How are project timelines set, and how can the development team assist in meeting deadlines?	Project Manager	The priorities and scope of a task determine the timeline. Accurate estimations and proactive communication of obstacles are provided by the team.

Will it be freely accessible or is there a paywall?	Financial Analyst	It’s freely accessible, however half of the features will be limited until the user signs up for a monthly payment.
What disaster recovery and business continuity measures are in place to ensure uninterrupted financial operations in case of system failures?	Financial Analyst	To reduce downtime and guarantee ongoing financial operations, the application has a strong disaster recovery plan that includes frequent backups, failover methods, and a tried-and-true business continuity strategy.
How does the program handle sensitive data like transaction details when it comes to financial data security?	Financial Analyst	To protect financial information, the application uses role-based access controls, encryption techniques for data in transit and at rest, and frequent security assessments.
What are the current marketing strategies, and how can developers align their work with these initiatives?	Marketing Manager	Content marketing and social media campaigns. Developers can align by highlighting new features in marketing materials and creating user-centric content.
How is the intended audience determined, and what information about user personas can you offer to help guide the development process?	Marketing Team Member	Using market research, we pinpoint the target market and build comprehensive user personas that encompass demographics, preferences, and pain areas to inform development choices.
What measures are in place to ensure the software meets industry compliance standards and regulations?	Quality Assurance Lead	We are committed to adhering to industry compliance standards. Regular compliance audits are conducted, and necessary adjustments are made to ensure the software aligns with the relevant regulations and standards.

How do you handle security aspects within the database, such as user access controls, encryption, and audit trails?	Database Administrator	User access controls are implemented through role-based permissions. Sensitive data is encrypted at rest and during transmission. We maintain audit trails to track changes and access, and we regularly review and update security measures to align with industry best practices.
Are there any ongoing database maintenance tasks, such as index rebuilds or statistics updates, to optimize overall database performance?	Database Administrator	Yes, routine maintenance tasks like index rebuilds and statistics updates are scheduled during low-traffic periods to minimize the impact on performance. We use automated processes and monitoring tools to track and execute these tasks.
How do you handle database backups and recovery procedures to ensure data availability and minimize downtime in case of system failures?	Database Administrator	We have a robust backup strategy of regular full and incremental backups, tested recovery procedures, and minimal downtime.
How can developers remain up to date on the newest security procedures and how is security awareness managed?	Security Analyst	Regular training is conducted; developers stay informed by participating, accessing resources, and seeking guidance on security matters.
Outline user access controls, and how can developers maintain a secure access environment?	Security Analyst	Role-based controls are employed; developers work together to resolve access-related issues and enforce appropriate access controls in code.

Deliverable #4

Functional Requirements

Requirement ID	Requirement Title	Short Description	Priority	Requester
FRO001	Alerts	The system must give a notification for any upcoming events/ scheduled tasks	High	IT Team Member
FRO002	Tasks	The system must allow the user to add tasks to this page	High	Software Tester
FRO03	Time Tracking for Tasks	The system should implement a time-tracking feature for tasks, helping users analyze how much time they spend on different activities	Medium	Data Analyst
FRO04	Task Archiving	The system should allow the user to archive completed tasks	Medium	Software Engineer
FRO05	Task Dependencies	The system must allow the user to set dependencies between tasks, enabling users to define relationships between tasks and ensuring that they are completed in the correct order	High	Programmer
FRO06	Hierarchal Task Structures	The system must allow users to create hierarchical structures for tasks, breaking down larger tasks into smaller sub-tasks for better organization	High	Project Manager

FRO07	Machine Learning Task Suggestions	The system should have machine learning algorithms to provide users with intelligent task suggestions based on their historical usage patterns and preferences	High	Marketer
FRO08	Task progress Tracking	The system must incorporate a task tracking feature that keeps track of how far the user is with their task	High	Programmer
FRO09	Collaborative Task Management	The system must allow the users to collaborate on shared tasks	High	Content moderator
FRO010	Discussion Threads for Tasks	The system should allow for discussion threads in tasks if users have questions or comments during collaboration	Medium	Customer support
FRO011	Integration with Task Management Tools	The system should allow for integration with other task management tools (Notes App, etc.)	Medium	Project Manager
FRO012	Schedule	The system must allow the user to create schedules that fit their needs	High	Marketer
FRO013	Timer	The system should allow the user to set a time and keep a history of the timers they have made	Medium	Investor
FRO014	Achievements	The system should allow the user to obtain	Medium	Marketer

		achievements based on how many tasks they complete		
FRO015	Profile	The system must allow users to customize their profiles and provide them with more flexibility for customizing the app to how they like	High	Customer
FRO016	Filter/Sorting	The system must allow the user to organize and sort their schedules or tasks for easy access	High	IT Team Member
FRO017	Settings	The system must allow the user to be able to edit their preferences regarding privacy, login information, etc.	High	IT Team Member
FRO018	Notetaking	The system must allow the user to make notes on any topic they like in a dedicated space in the app	High	Marketer
FRO019	Collaborative Notetaking	The system should enable users to collaborate on notes or tasks with team members or friends	Medium	Content moderator
FRO020	Voice to Notetaking	The system should incorporate a feature that allows users to dictate notes using voice commands, which are then converted to text	Medium	Administrator
FRO021	Offline Access	The system should enable users to access and edit their notes and schedules offline,	Medium	Software Engineer

		with automatic synchronization once an internet connection is established		
FRO022	Integration with Note-Taking Devices	The system should enable integration with popular note-taking devices (e.g., digital pens, tablets) to facilitate seamless transfer of handwritten notes or drawings into the app	Medium	Marketer
FRO023	Attachments and File Uploads	The system must allow users to attach files, documents, or images to their notes or tasks for additional context or reference	High	Investor
FRO024	Smart Tags	The system should implement an intelligent tagging system that allows users to categorize and organize their notes and tasks using tags	Medium	Customer
FRO025	Advanced Search and Filters	The system must provide users with advanced search functionalities, allowing them to search notes and tasks based on keywords, dates, tags, and other criteria. Additionally, include filters for more refined searches	High	Data Analyst
FRO026	Customizable Templates	The system should allow users to create and save customizable	Medium	Digital Designer

		templates for different types of notes or tasks		
FRO027	Friends	The system should allow the user to connect with friends and share achievements and schedules with them. More features regarding this topic can be added to this in the future...	Medium	Marketer
FRO028	Export/Share Schedule	The system should allow the user to export and share their schedules as a PDF, CSV, etc.	Medium	Software Engineer
FRO029	Customizable Notifications	The system should provide users with the ability to customize the types and frequency of notifications they receive for tasks and events	Medium	Software Tester
FRO030	Real-Time Collaboration	The system must facilitate real-time collaboration on shared notes and tasks, allowing multiple users to edit and view changes simultaneously	High	Marketer
FRO031	Interactive Calendar Views	The system must allow the user to access and customize their calendar views	High	Administrator
FRO032	Dark Mode	The system could allow the user to toggle between a light and dark mode	Low	Digital Designer
FRO033	Location Based Reminders	The system should allow the user to	Medium	Content Moderator

		receive an alert depending on their location		
FRO034	Integration with Cloud Storage	The system should allow the user to access other cloud storage services for easy access to files	Medium	Customer Support
FRO035	Customizable Keyboard Shortcuts	The system should allow the user to set their own keyboard shortcuts	Medium	Programmer
FRO036	Event RSVP	The system could allow the user to receive RSVPs for certain scheduled events	Low	Project Manager
FRO037	Cross-Platform Clipboard Integration	The system should allow the user to clip a text or image from one program to the current app	Medium	Software Engineer
FRO038	Automated Backup and Recovery	The system must allow for automatic backups for emergency recovery needs	High	Cyber Security
FRO039	Integration With Voice Assistant	The system must allow the user to interact with a voice assistant	High	Administrator
FRO040	Smart Tagging + Auto-Categorization	The system should work with the smart tagging and be able to automatically categorizes notes and tasks based on content analysis	Medium	Data Analyst
FRO041	Frequently Asked Questions (FAQ)	The system should have the ability to allow administrators to	Medium	Customer Support

		include a FAQ for users to refer to and be able to update as needed		

Requirement ID	Requirement Title	Short Description	Priority	Requester
NFRO001	Google Calendar Integration	The system should allow the user to connect their Google calendar to the app to allow a seamless transition for their schedules	Medium	IT Team Member
NFRO002	Consistent User Experience	The system should provide a consistent user experience across different platforms, including desktop, web, and mobile	Medium	Customer
NFRO003	Location	The system could allow the user to add a location to certain tasks if they are to take place somewhere they may need a reminder	Low	Investor
NFRO004	Themes	The system could allow the user to customize the app however they would like with different color schemes, fonts, etc.	Low	Digital Designer
NFRO006	Mobile Compatibility	The system should allow the app to be compatible with mobile devices	Medium	IT Team Member
NFRO007	Web Browser Compatibility	The system must allow the app to be compatible with multiple web	High	IT Team Member

		browsers (Chrome, Firefox, explore, etc.)		
NFRO008	Network Data Transfer	The system will allow the app to save/upload data at an exceptional speed for the user (depending on what they're uploading)	High	Marketer
NFRO009	Data Security and Privacy	The system must implement robust security measures to ensure the confidentiality and privacy of user data, including encrypted storage and secure transmission	High	Cyber Security
NFRO010	Scalability	The system must allow the app to handle a growing number of users, notes, and tasks without compromising performance or user experience	High	Digital Designer
NFRO011	Backup and Recovery	The system should implement regular automated backups of user data and provide a straightforward recovery process in case of data loss or accidental deletion	Medium	Data Analyst
NFRO012	Offline Sync Efficiency	The system should ensure efficient synchronization of data when transitioning between offline and online modes, minimizing potential data conflicts	Medium	Cyber Security
NFRO013	Performance Optimization	The system must optimize the app's	High	Programmer

		performance to provide fast loading times and smooth interactions, even when dealing with large numbers of notes and tasks		
NFRO014	User friendly Interface	The system must prioritize a clean and intuitive user interface to enhance the user experience, making it easy for users to navigate, add, and manage notes and tasks	High	Digital Designer
NFRO015	Gamification Elements	The system could award the user with badges along with the achievements they receive	Low	Digital Designer
NFRO016	Social	The system could allow the user to share their achievements and notes on their social media platforms if desired	Low	Marketer
NFRO017	Multi-device Syncing	The system should support seamless syncing across devices, allowing users to access their notes and schedules consistently whether on a computer, tablet, or smartphone	Medium	Data Analyst
NFRO018	Offline First	The system should have key functionalities remain accessible while offline	Medium	Project Manager
NFRO019	Cross Browser Capability Test	The system should undergo rigorous testing to ensure consistent and	Medium	IT Team Member

		optimal performance across various web browsers, including but not limited to Chrome, Firefox, Safari, and Edge		
NFRO020	Accessibility Compliance	The system must adhere to accessibility standards (e.g., WCAG) to ensure that individuals with disabilities can use the application effectively	High	Digital Designer
NFRO021	Load Balancing	The system must implement load balancing mechanisms to distribute network traffic efficiently across servers, preventing overloading and ensuring optimal performance during peak usage	High	IT Team Member
NFRO022	Automated Performance Testing	The system must undergo regular automated performance testing to identify and address any performance bottlenecks, ensuring the application's responsiveness and reliability	High	Programmer
NFRO023	Session Token Expiry	The system should provide options for users to have sessions expiry periodically and force them to log in again	Medium	Cyber Security

NFRO024	Notifications for Network and Cloud Sync Issues	The system could notify users that the device they are using has issues synchronizing files.	Low	Software Engineer
NFRO025	Tracking (Feature) Usage Patterns	The system could track usage patterns for analytics.	Low	Data Analyst
NFRO026	Error Logging (Client Devices)	The system should have error reports for client device malfunction that will automatically be sent to a central repository (if users consent).	Medium	Data Analyst
NFRO027	Internationalization	The system must support multiple languages and regional preferences.	High	Customer

Part B

Deliverable #1

Use Case Name	List of Related Requirements ID	Actor(s)	Brief Description
Create a task	FR02, FRO5, FRO6, FR07, FRO34, FRO38	Customer	The actor will click on a "create a task" button, in response, the software will prompt them for a title and description for their task.
Edit a task	FRO6, FRO9, FRO34, FRO38	Customer	The actor will modify existing task details within the system
Track a task	FRO3, FR08, FRO10, FRO34, FRO38	Customer	The actor monitors the progress of a task within the system.
Complete a task	FR4, FRO34, FRO38	Customer	The actor marks a task as completed within the system, indicating its successful execution or resolution
Edit project notes at the same time as other members	FRO27, FRO30,	Customer	The actors collaborate in real-time to edit project notes concurrently
Manage Task collaboratively	FR009, FR010, FRO16, FRO19, FRO29	Customer, Content Moderator	The actors will collaborate on shared tasks, engaging in discussions and tracking progress collectively
Create Subtasks within Tasks	FRO8, FRO27, FRO29	Project Manager, Stakeholders	The actor breaks down larger tasks into smaller subtasks within the system, assigning them to team members for more granular task management
Set Task Priority Levels	FRO2, FRO6	Team Leader, Team Members	The actor sets priority levels for tasks within the system, helping the

			other actors focus on high-priority tasks and meet project deadlines effectively
Create Custom Content Templates	FRO18, FRO23, FRO26, FRO35	Customer	The actor designs and saves customizable templates for different types of content within the system
Manage software interface	FR07, FR08, FR09, FRO32, FRO35,	Customer	The actor customizes the software interface according to personal preferences and requirements
Manage security settings	FRO7,FRO9, FRO38	Customer	The actor configures security settings within the system to ensure data protection and access control
Access FAQ	FR010	Customer	The actor accesses frequently asked questions (FAQ) within the system for reference and assistance
Update FAQ content	FRO41	Customer support	The actor updates and maintain FAQ content within the system to provide accurate and helpful information
Create Smart Tags for Organization	FRO24, NFRO09	Content Moderators	The actor creates smart tags within the system, allowing for automatic categorization and organization of notes and tasks based on predefined criteria
Manage Schedule	FR03	Investor	The actor organizes and manages schedules, including events, tasks, and appointments, within the system
Manage Tasks	FR02	Customer	The actor oversees and manages tasks, including creation, assignment, and

			tracking, within the system
Import Schedule (from other applications)	FRO12, FRO28	Customer	The actor imports schedules and events from external applications into the system for unified management
RSVP to events	FRO2, FRO12, FRO27, FRO36	Organizers, Attendees	The actor responds to event invitations within the system, confirming attendance or indicating availability
Start Study Session	FRO13, FRO14, FRO29	Customer	The customer will push a button to start tracking a study session that contribute to achievements
Create a note	FRO18, FRO20, FRO23	Customer	he actor creates a new note within the system, adding content and details as needed
Edit a note	FRO19	Content Creators, Student	The actor modifies the content or details of an existing note within the system
Customize workspace	FRO16, NFRO20	Customer, Content Moderator	The actor customizes the workspace layout, organization, and features within the system
Organize workspace	FRO16, FRO25	Project Managers, Team Leads	The actor organizes and manages workspace elements, including notes, tasks, and schedules, within the system
Collaborate for notes in real time	FRO19, FRO26, NFRO09	Team Members, Editors	The actors will collaborate in real-time to edit and contribute to shared notes concurrently
Schedule Automated Backups	FRO37, FRO38	System Administrator, IT Team	The actor automated backup schedules within the system to ensure regular backups of user data,

			minimizing the risk of data loss
Access Notes Offline	FRO21, NFRO12	Mobile Users, Commuters	The actor accesses their notes and tasks offline within the system, ensuring productivity even in areas with limited or no internet connectivity
Log into system	FRO17	Employees, Customers, Administrators	The actor securely logs into the system using authentication credentials
Organize data	FRO16, FRO25, FRO24, FRO2, FRO40	Content moderator	The actor organizes and manages data elements within the system, including notes, tasks, and schedules
Manage network	FRO34, FRO19, FRO38	IT team member, security analyst	The actors manage network configurations and security measures within the system
Set Location-Based Reminders	FRO32, NFRO03	Mobile Users, Travelers	The actors set location-based reminders within the system, receiving alerts when they are near specific locations, such as picking up groceries or attending meetings
Send notifications	FRO1, FRO33	Notification system	The system sends notifications to users based on predefined triggers, events, or updates.
Back up user data	FRO38	Backup system	The system automatically backs up user data to ensure data integrity and availability in case of loss or corruption
Track feature usage	NFRO22	Data Analyst	Data analysts track user interactions and feature usage patterns within the system for

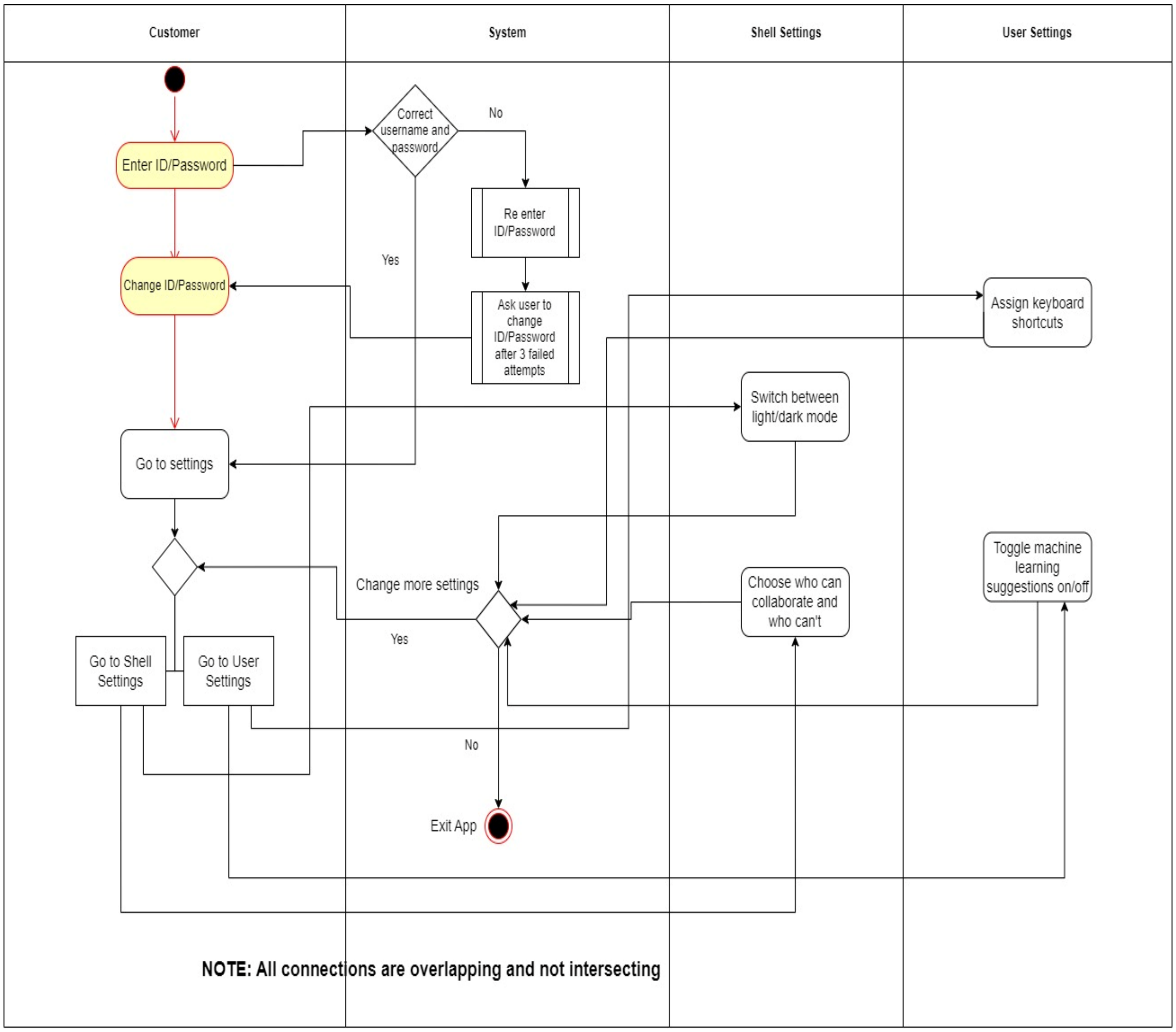
			analysis and optimization

Deliverable #2

Use Case	**Manage Software Interface**
Iteration:	2, last modification: February 5 by Roxana Maria.
Primary Actor:	Customer
Goal in Context:	To be allowed to collaborate on tasks while also managing other software Interfaces like switching between light/dark mode, task tracking, and machine learning task suggestions. This also contains keyboard shortcuts.
Preconditions:	The system must be fully configured, and user must log in(successfully) using their ID and password to change and save their preferences.
Trigger:	The Customer wants to share and collaborate on a task and/or wants to change between light/dark mode and adding keyboard shortcuts. The Customer could also want to have tracking on for their tasks so they can keep up with each of them and have machine learning algorithms provide users with intelligent task suggestions based on the Customer's historical usage.
Scenario:	1. The Customer logs in to SAAYJ. 2. The Customer goes to settings. 3. The Customer then clicks on Manage Software Interface.

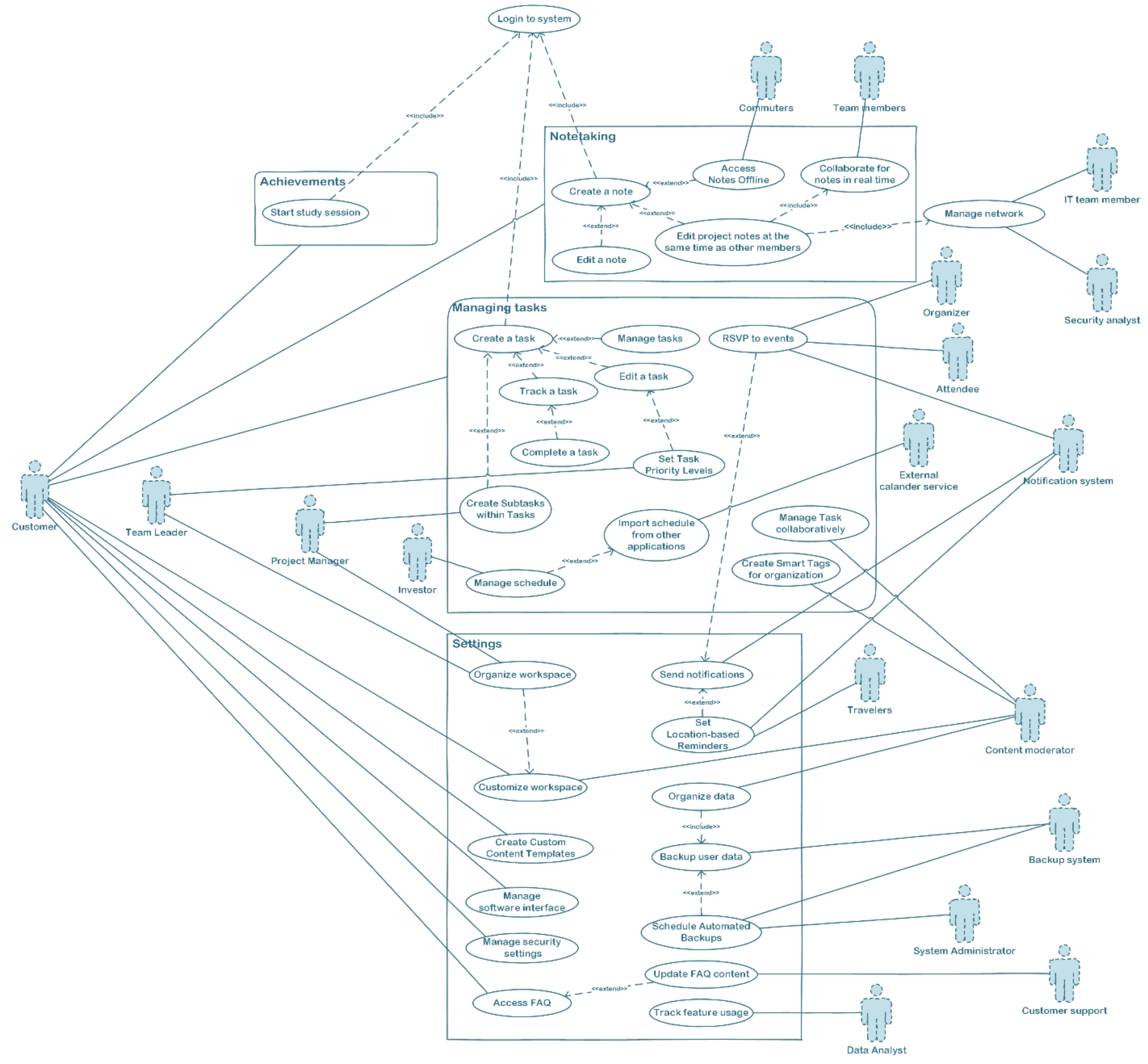
	4. All the settings regarding the Software Interface are located here. 5. The Customer can switch between light and dark mode based on their preference. 6. The Customer selects “Keyboard Shortcuts”. 7. The Customer can then assign different keyboard shortcuts. 8. The Customer selects “Task Tracking and machine-learning task suggestions”. 9. The Customer has the option to toggle on and off for task tracking and machine-learning task suggestions. 10. The user selects “Collaboration”. 11. The Customer can then choose who can share tasks and who can’t.
Exceptions:	1. The username and/or password are incorrect - See Use Case **Manage Security Settings** 2. The Customer has questions - See Use Case **Access FAQ** 3. The customer is unable to configure task-tracking settings - See Use Case **Track A Task.**
Priority:	Moderate priority, will be implemented after all the basic functions.
When Available:	Third Increment
Frequency of Use:	Frequent - Multiple times daily.
Channel to Actors:	Via Mobile Phones.

Secondary Actors:	Programmer
Channels to Secondary Actors:	Programmer codes for the use case to be implemented and usable Via Mobile Phones.
Open Issues:	1. Collaboration and shared tasks could lead to data leaks and breaches. What will Cybersecurity do to ensure this does not happen? 2. Can shared tasks still be accessible and edited while offline? 3. How will the shared tasks be organized and how will the user tell who the document is being shared with?

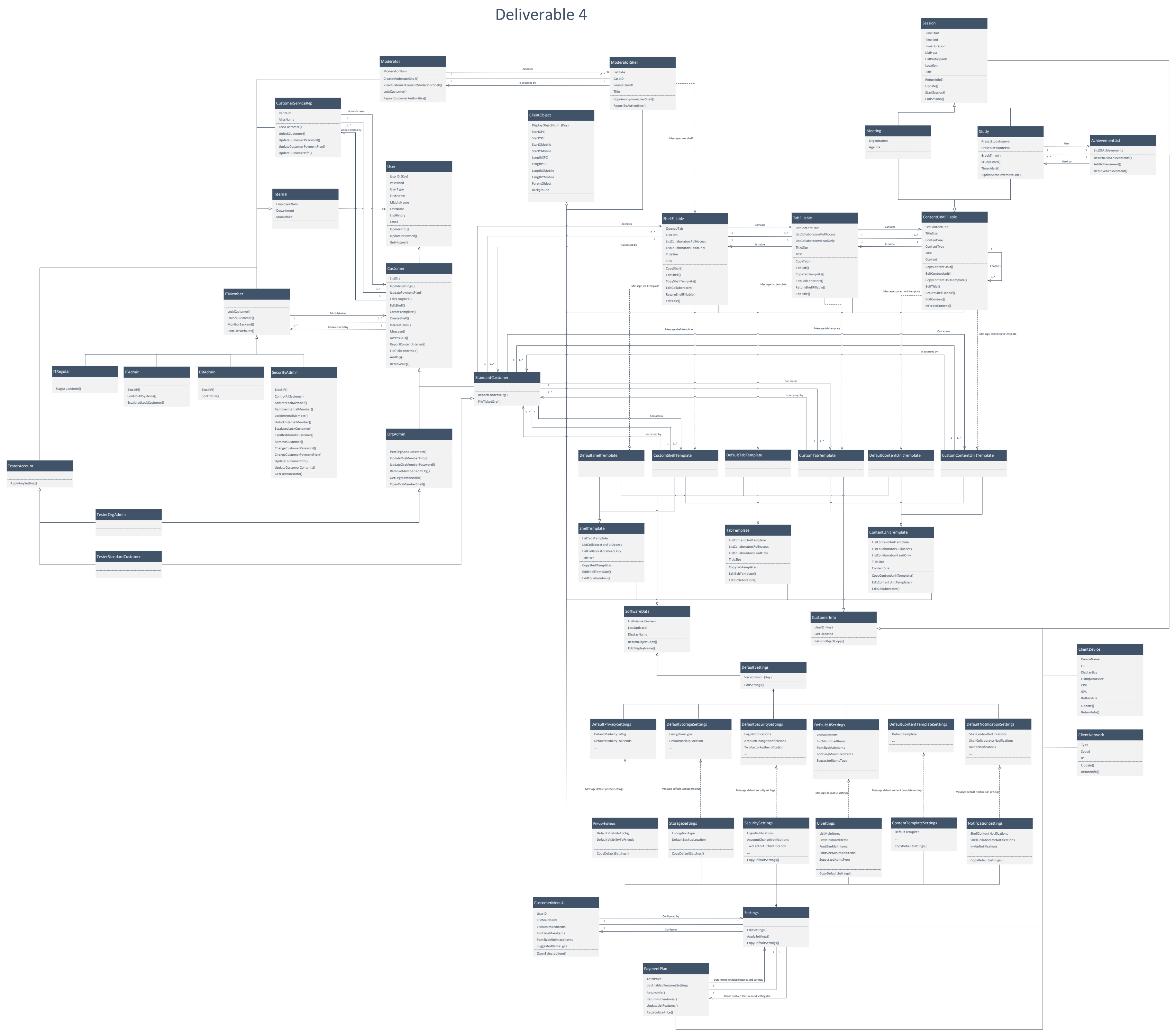

Customer
System
Shell Settings
User Settings
Enter ID/Password
Correct username and password
No
Re enter ID/Password
Yes
Change ID/Password
Ask user to change ID/Password after 3 failed attempts
Assign keyboard shortcuts
Switch between light/dark mode
Go to settings
Change more settings
Toggle machine learning suggestions on/off
Choose who can collaborate and who can't
Yes
Go to Shell Settings
Go to User Settings
No
Exit App
NOTE: All connections are overlapping and not intersecting

Deliverable #3

SAAYJ



Deliverable #4



- The contents of the classes in this diagram are visible when magnified in the digital version of this document.
- This document is also included in a Visio file submitted with this file.
- The relationships and connections shown in this diagram may not be all inclusive; the diagram is intended to show all major relationships and connections.
- **Connections that do not intersect in the diagram are shown with lines going over other lines with a small curve; connections that do intersect contact a right angle.**

Deliverable #5

Class: Moderator

Description: This class lets moderators view customer content using a moderator shell. It inherits from the Internal class and can create and view moderator shells.

Responsibility:	**Collaborator:**
Allows the moderator to view customer content.	ModeratorShell class (interacts with it).
Enables the moderator to lock a customer account.	
Allows reporting of customer authorities.	

Class: ModeratorShell

Description: This class, inheriting from ClientObject, enables moderators to access information discreetly. It allows them to clone anonymous user shells from ShellFillable and view them through the Moderator class.

Responsibility:	**Collaborator:**
Stores information related to tabs (e.g., which tabs are available to the moderator).	Moderator Class
Enables the moderator to lock a customer account.	ClientObject
Allows reporting of customer authorities.	ShellFillable

Class: ClientObject

Description: Serves as a fundamental entity for managing and rendering graphical elements on various platforms, including desktop and mobile devices.

Responsibility:	**Collaborator:**
Represents graphical objects displayed within the client's interface.	
Manages positional and dimensional properties for rendering on different platforms.	
Stores unique identifiers and parent-child relationships for object hierarchy if applicable.	
Tracks background details for visual representation within the user interface.	

Class: SoftwareData

Description: Serves as a central repository for managing and storing crucial data related to software entities within the system.

Responsibility:	**Collaborator:**
Manages and provides access to data related to software entities.	
Tracks the list of internal owners, last updated timestamp, and display name of the software.	
Provides a method to return a copy of the software data object for external use or manipulation.	

Class: CustomerInfo

Description: Represents an object within the client's interface, essential for managing user-specific data and interactions within the system.

Responsibility:	**Collaborator:**
Manages user-specific objects and data within the client's interface.	
Stores the UserID to associate objects with specific users, facilitating personalized interactions and data management.	
Tracks the LastUpdated timestamp to monitor changes and ensure data freshness.	
Implements a method to safely share object copies, preserving the original data's integrity and security.	

Class: AchievementList

Description: Represents a collection of achievements. It manages a list of achievements and provides methods to interact with them.

Responsibility:	**Collaborator:**
Maintaining a list of achievements.	Study
It allows adding new achievements to the list.	CustomerInfo
It provides functionality to remove achievements from the list.	
The class can return the entire list of achievements.	

Class: ContentUnitTemplate

Description: Represents a template for creating content units within the system. It defines the structure and access permissions for content creation and collaboration.

Responsibility:	**Collaborator:**
Defines the structure and access permissions for creating content units.	ClientObject
Manages templates for consistent content layout and style across the system.	
Tracks collaborators with different access levels (full access or read-only) for content based on this template.	
Provides methods for copying, editing, and managing collaborators associated with content unit templates.	

Class: TabTemplate

Description: Defines the structure and settings for creating tabs within the system's user interface.

Responsibility:	**Collaborator:**
Defines the structure and settings for creating tabs in the user interface.	ClientObject
Manages predefined templates for consistent content organization and collaboration within tabs.	
Tracks collaborators and their access levels for content within tabs based on this template.	
Provides methods for copying, editing, and managing collaborators associated with tab templates.	

Class: ShellTemplate

Description: Represents a template for defining the structure and settings of a shell interface in the system.

Responsibility:	**Collaborator:**
Defines the structure and settings of a shell interface in the system.	ClientObject
Manages predefined tab templates for consistent tab creation within the shell.	
Tracks collaborators and their access levels for content within tabs based on this shell template.	
Provides methods for copying, editing, and managing collaborators associated with shell templates.	

Class: DefaultShellTemplate

Description: The default layout and structure for the shell interface, managed by internal users.

Responsibility:	**Collaborator:**
Provides a standard shell interface layout and structure for system-wide consistency.	ClientObject
	SoftwareData
	ShellFillable

Class: CustomShellTemplate

Description: Represents a customizable shell interface template for customers.

Responsibility:	**Collaborator:**
Allows customers to tailor the shell interface layout to meet their specific needs and branding.	ClientObject
	SoftwareData
	ShellFillable

Class: DeafultTabTemplate

Description: Defines the default structure and content arrangement within tabs, managed by internal users.

Responsibility:	**Collaborator:**
Ensures a consistent tab layout and content organization across the system.	ClientObject
	SoftwareData
	TabFillable

Class: CustomTabTemplate

Description: Enables customers to create and customize tabs within the shell interface.

Responsibility:	**Collaborator:**
Provides flexibility for customers to design tab layouts and content organization according to their preferences.	ClientObject
	SoftwareData
	TabFillable

Class: DefaultContentUnitTemplate

Description: Defines default structure and content elements within content units, managed by internal users.

Responsibility:	**Collaborator:**
Maintains a standard format and layout for content units, ensuring uniformity in content presentation.	ClientObject
	SoftwareData
	ContentUnitFillable

Class: CustomContentUnitTemplate

Description: Allows customers to define and customize content unit structures according to their needs.

Responsibility:	**Collaborator:**
Empowers customers to create unique content unit layouts and structures tailored to their content requirements.	ClientObject
	SoftwareData
	ContentUnitFillable

Class: ContentUnitFillable

Description: Represents a fillable content unit that can be managed and customized by users.

Responsibility:	**Collaborator:**
Defines a fillable content unit template for creating and managing content.	TabFillable
Provides methods for copying, editing, and managing titles, content, and associated templates.	ShellFillable
	ClientObject

Class: TabFillable

Description: Defines the structure and settings for creating tabs within the system's user interface.

Responsibility:	**Collaborator:**
Defines the structure and settings for creating tabs in the user interface.	ClientObject
Manages predefined templates for consistent content organization and collaboration within tabs.	ContentUnitFillable
Tracks collaborators and their access levels for content within tabs based on this template.	
Provides methods for copying, editing, and managing collaborators associated with tab templates.	

Class: ShellFillable

Description: A versatile shell or template that can be filled with content. It is designed to be adaptable and customizable for various purposes.

Responsibility:	**Collaborator:**
Handles the creation, modification, and utilization of shell templates.	StandardCustomer
Manages lists of collaborators who have different levels of access (full access or read-only) to the shell.	TabFillable
The class stores information about the shell's title and size.	ContentUnitFillable
Provides methods for copying shells, editing their content, and updating titles.	ClientObject

Class: Study

Description: Represents a study session management system. It helps users maintain focus during study sessions and manage breaks effectively.

Responsibility:	**Collaborator:**
Manages study sessions, including tracking study time and break intervals.	AchievementList
Provides functionality to alert users when study or break intervals are complete.	Session
Collaborates with an AchievementList to update achievements based on study milestones.	ContentUnitFillable

Class: Meeting

Description: A scheduled gathering or assembly where individuals come together to discuss specific topics, share information, and make decisions.

Responsibility:	**Collaborator:**
For handling meeting-related information and actions.	Session
Manages the meeting agenda, which outlines the topics to be covered during the meeting.	ContentUnitFillable
Stores information about the organization or group hosting the meeting.	

Class: Session

Description: Manages session details, including start and end times, participants, location, and title.

Responsibility:	**Collaborator:**
handles session-related information and actions.	CustomerInfo
Tracks the start and end times of the session and calculates the duration.	
Maintains lists of hosts and participants.	
Stores information about the session's location and title.	

Class: CustomerMenuUI

Description: Represents the user interface for displaying the customer menu and user interface overlay.

Responsibility:	**Collaborator:**
Displays User menu	Settings
Handles user interactions reallated to menu items	ClientObject

Class: Settings

Description: Configures the CustomerMenuUI and its settings profile to each user.

Responsibility:	**Collaborator:**
Contains enabled features and settings.	PaymentPlan: reads enabled features and settings list
Configures the CustomerMenuUI and PaymentPlan	MenuUI Class: configures the class

Class: PaymentPlan

Description: A payment plan profile for each user that manages available features and settings to that user's account.

Responsibility:	**Collaborator:**
Calculate total price	Setting: Determines enabled features and settings
Return information, list of features, and updates features	CustomerInfo
Recalculates price	

Class: PrivacySettings

Description: manages privacy-related settings within the system.

Responsibility:	**Collaborator:**
Manage default visibility settings for organization and more.	Settings
Provide functionality to copy privacy settings and more.	DefaultPrivacySettings (Copy default privacy settings)

Class: StorageSettings

Description: Manages storage-related settings within the system.

Responsibility:	**Collaborator:**
Manage encryption type and default backup location settings and more.	Settings
Provide functionality to copy default settings and more.	DefaultStorageSettings (Copy default storage settings)

Class: ClientDevice

Description: Stores information about the device's specifications and to update its settings and retrieve device information

Responsibility:	**Collaborator:**
Storing and providing info about client device, its name, CPU, GPU, battery life	CustomerInfo
Update the settings or configurations of the client device	

Class: NotificationSettings

Description: Manages notification settings within the system.

Responsibility:	**Collaborator:**
Handles the management of shell content notifications, shell collaborator notifications, and invite notifications and more.	Settings
Provide functionality to copy notification settings and more.	DefaultNotificationSettings (Copy default notification settings)

Class: UISettings

Description: Manages user-related settings within the system.

Responsibility:	**Collaborator:**
Manage lists of main and minimized items, font sizes, and suggested item types for the user interface and more.	Settings
Provide functionality to copy UI settings and more	DefaultUiSettings (Copy default UI settings)

Class: ClientNetwork

Description: Represents the connection established by a client device with the system.

Responsibility:	**Collaborator:**
Storing information including connection type, speed, and IP address	Customerinfo
Update the settings or configurations of the client network	

Class: SecuritySettings

Description: Manages security-related settings within the system.

Responsibility:	**Collaborator:**
Manage login, account change, and two-factor authentication notification settings and more.	Settings
	DefaultSecuritySettings (Copy default security settings)

Class: ContentTemplateSettings

Description: Manages content template settings within the system.

Responsibility:	**Collaborator:**
Handles the management of default content templates and more.	Settings
Provide functionality to copy default settings and more	DefaultContentTemplateSettings (Copy default content template settings)

Class: DefaultNotificationSettings

Description: Manages notification settings within the system.

Responsibility:	**Collaborator:**
Handles the management of shell content notifications, shell collaborator notifications, and invite notifications	DefaultSettings
	NotificationSettings

Class: DefaultPrivacySettings

Description: manages privacy-related settings within the system.

Responsibility:	**Collaborator:**
Manage default visibility settings for organization and more.	DefaultSettings
	PrivacySettings

Class: DefaultStorageSettings

Description: Manages storage-related settings within the system.

Responsibility:	**Collaborator:**
Manage encryption type and default backup location settings and more.	DefaultSettings
	StorageSettings

Class: DefaultUISettings

Description: Manages user-related settings within the system.

Responsibility:	**Collaborator:**
Manage lists of main and minimized items, font sizes, and suggested item types for the user interface and more.	DefaultSettings
	UiSettings

Class: DefaultSecuritySettings

Description: Manages security-related settings within the system.

Responsibility:	**Collaborator:**
Manage login, account change, and two-factor authentication notification settings and more.	DefaultSettings
	SecuritySettings

Class: DefaultContentTemplateSettings

Description: Manages content template settings within the system.

Responsibility:	**Collaborator:**
Handles the management of default content templates and more.	Settings
	DefaultContentTemplateSettings

Class: DefaultSettings

Description: Manages default settings within the system.

Responsibility:	**Collaborator:**
Manages and provides access to default settings.	SoftwareData
Handles editing of default settings through the EditSettings() method.	

Class: CustomerServiceRep

Description: Represents a customer service representative who handles customer-related tasks and interactions within the system.

Responsibility:	**Collaborator:**
Manages customer interactions and tasks such as locking/unlocking accounts, updating passwords	Customer
Manages payment plans, and customer information.	

Class: Internal

Description: Represents internal employees within the system.

Responsibility:	**Collaborator:**
Represents internal employees' data such as employee numbers, departments, and office locations.	User
Serves as a data structure for internal employee information.	

Class: User

Description: Represents a general user in the system.

Responsibility:	**Collaborator:**
Manages user information, authentication, and activity history.	

Class: Customer

Description: Represents a customer entity in the system.

Responsibility:	**Collaborator:**
Manages customer-specific operations such as updating settings, payment plans, templates.	User
Manages shells, interactions, messaging, organization management, and internal reporting/ticketing.	ITMember

Class: ITMember

Description: Represents a member of the IT department responsible for system maintenance and user management.

Responsibility:	**Collaborator:**
Manages customer account locking/unlocking for security purposes.	Internal Customer
Monitors and maintains the backend systems for optimal performance and security.	Internal
Edits default settings and configurations related to user accounts or system preferences as needed by the IT department.	

Class: TesterStandardCustomer

Description: Represents a tester account with standard customer privileges for testing customer-level functionalities.

Responsibility:	**Collaborator:**
Tests standard customer functionalities within the system.	TesterAccount

Class: OrgAdmin

Description: Represents an organizational administrator responsible for managing organizational-level tasks and members.

Responsibility:	**Collaborator:**
Manages organizational announcements, member information, passwords, and access within the organization.	Customer

Class: TesterAccount

Description: Represents a tester account used for testing various system functionalities and settings.

Responsibility:	**Collaborator:**
Provides a testing environment for experimenting with different system settings and functionalities.	Internal

Class: TesterOrgAdmin

Description: Represents tester account with organizational administrative privileges for testing organizational-level functionalities.

Responsibility:	**Collaborator:**
Tests organizational administrative functionalities within the system.	TesterAccount

Class: ITRegular

Description: Represents a regular IT staff member responsible for handling routine IT tasks.

Responsibility:	**Collaborator:**
Flags and reports issues for further investigation by IT administrators.	ITMember

Class: ITAdmin

Description: Responsible for managing IT-related tasks and systems.

Responsibility:	**Collaborator:**
Manages IP blocking, system control, and escalated customer account locking.	ITMember

Class: DBAdmin

Description: A database administrator responsible for managing database-related tasks and systems.

Responsibility:	**Collaborator:**
Manages IP blocking and control of database systems.	ITMember

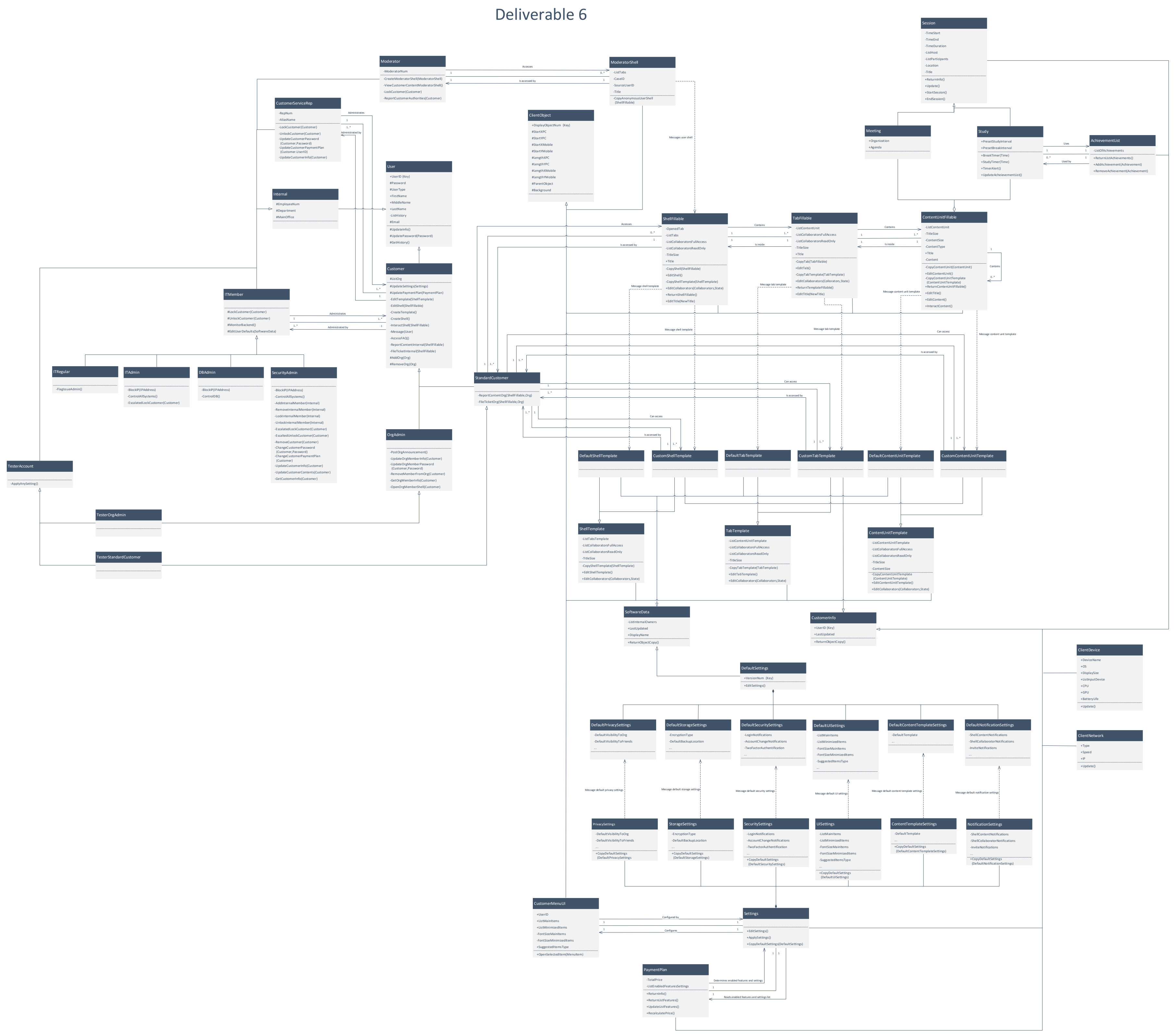
Class: SecurityAdmin

Description: A database administrator responsible for managing database-related tasks and systems.

Responsibility:	**Collaborator:**
Manages security-related tasks such as IP blocking, system control, member management	ITMember
Manages customer account management with escalated security measures.	

Deliverable #6

- The contents of the classes in this diagram are visible when magnified in the digital version of this document.
- This document is also included in a Visio file submitted with this file.
- The relationships and connections shown in this diagram may not be all inclusive; the diagram is intended to show all major relationships and connections.
- **Connections that do not intersect in the diagram are shown with lines going over other lines with a small curve; connections that do intersect contact a right angle.**



Class Attribute Description

- User: This class contains attributes and methods that are inherited by all user accounts of the system (including internal and external company members). Every user account class inherits from this class. Attributes of this class contain the personal information of each user, such as UserID and FirstName.
- Customer: This class contains attributes and methods that are inherited by all customer accounts (external users). This class inherits from the User class. The OrgAdmin and StandardCustomer classes inherit from this class. This class is administrated by the ITMember and CustomerServiceRep classes. This class can access the PaymentPlan class by requesting a recalculation of the price of the chosen payment plan.
- Internal: This class contains attributes that are inherited by all internal accounts of the company managing the SAAYJ. This class inherits from the User class. The ITMember, CustomerServiceRep, Moderator, and TesterAccount classes all inherit from this class. Attributes of this class contain basic information about each internal employee, such as EmployeeNum and Department.
- StandardCustomer: This class contains methods that are actions each standard customer can take, such as organizing report content and filing tickets. This class inherits from the Customer class. A standard customer can be a member of an organization. The TesterStandardCustomer class inherits from this class. This class can access the ShellFillable, CustomShellTemplate, CustomTabTemplate, and CustomContentUnitTemplate classes.
- OrgAdmin: This class contains methods that are actions each organization administrator can take, such as updating an organization member’s password. This class is for enterprise customers. This class inherits from the Customer class. The TesterOrgAdmin class inherits from this class.
- ITMember: This class contains methods that are inherited by all IT employee accounts in the company managing the SAAYJ. This class inherits from the Internal class. The ITRegular, ITAdmin, DBAdmin, and SecurityAdmin classes inherit from this class. This class administrates the Customer class by monitoring customer activities such as locking customer accounts that have been inactive for a long time.
- ITRegular: This class contains methods that are actions each regular IT employee can take. This class inherits from the ITMember class.
- ITAdmin: This class contains methods that are actions each IT admin can take, such as blocking suspicious IP addresses. This class inherits from the ITMember class. This class’s level is higher than the ITRegular class.
- DBAdmin: This class contains methods that are actions each database admin can take, such as managing the database. This class inherits from the ITMember class.
- SecurityAdmin: This class contains methods that are actions each security admin can take, such as removing internal accounts of resigned employees in the company managing the SAAYJ. This class inherits from the ITMember class. This class has the highest level among all the other subclasses of the ITMember class.
- CustomerServiceRep: This class contains attributes that contain basic information about each customer service representative, such as RepNum and AliasName. This class inherits from the Internal class. This class administrates the Customer class by managing customer accounts such as updating customer passwords.
- Moderator: This class contains methods that are actions each moderator can take, such as viewing customer content moderator shell. This class inherits from the Internal class. This class can access the ModeratorShell class by creating and viewing the moderator shells.
- ModeratorShell: This class contains attributes and methods that are used to clone an anonymous user shell from the ShellFillable class so that the moderator can view it without identifying the owner of the shell. This class inherits from the ClientObject class. This class is accessed by the Moderator class.
- TesterAccount: This class contains methods that are actions each testing engineer can take in the testing period. This class inherits from the Internal class. The TesterOrgAdmin and TesterStandardCustomer classes are inherited from this class.
- TesterOrgAdmin: This class represents each tester account of organization administrators. This class inherits from the TesterAccount and OrgAdmin classes. Testing engineers use this class to test the functionality of an organization administrator account.
- TesterStandardCustomer: This class represents each tester account of standard customers. This class inherits from the TesterAccount and StandardCustomer classes. Testing engineers use this class to test the functionality of a standard customer account.
- Session: This class contains attributes and methods that are used to manage a time session, such as ending a session and recording the time duration of the session. This class inherits from the CustomerInfo class. The Meeting and Study classes are inherited from this class.
- Meeting: This class contains attributes that contain basic information about each meeting attended by the customer. This class inherits from the Session class. The whole part relationship between the Meeting class and the ContentUnitFillable class is aggregation, in other words, meeting is an optional component of the content unit.
- Study: This class contains attributes and methods that are used to manage each study session created by the customer. This class inherits from the Session class. The whole part relationship

between the Study class and the ContentUnitFillable class is aggregation, in other words, study session is an optional component of the content unit. This class uses the AchievementList class by updating the list of achievements whenever the customer creates a study session and achieves a study time goal.

- AchievementList: This class contains attributes and methods that are used to manage the achievement list of each customer. This class inherits from the CustomerInfo class. This class is used by the Study class to return the list of achievements using data gained from the study timer.
- ClientDevice: This class contains attributes that contain information about each client's physical device, such as OS and BatteryLife. This class inherits from the CustomerInfo class.
- ClientNetwork: This class contains attributes that contain information about each client network, such as IP and Speed. This class inherits from the CustomerInfo class.
- PaymentPlan: This class contains attributes and methods that are used to manage the payment plan chosen by each customer. This class inherits from the CustomerInfo class. This class determines enabled features and settings and shares them with the Settings class. This class can be accessed by the Customer class.
- CustomerMenuUI: This class contains attributes and methods that are used to customize each customer menu UI. This class inherits from the ClientObject class. This class is configured by the Settings class, in other words, customers can customize the menu UI by editing settings.
- ClientObject: This class serves as the base class for any object that interacts directly with clients in the system. It could contain methods and attributes common to objects that require client data or need to communicate with clients.
- ShellFillable: This class defines a set of methods and attributes necessary for objects that can be "filled" into a shell or framework within the application, such as content areas or user interfaces. User account classes that inherit from the Customer class can create and interact with ShellFillable objects and the objects contained within them. ShellFillable objects contain TabFillable objects, which means they also contain ContentUnitFillable objects.
- TabFillable: This class focuses on the aspects of filling content or features into a tab structure within the application's user interface, providing a method to organize and display information in tabbed sections. TabFillable objects are contained in ShellFillable objects. TabFillable objects contain ContentUnitFillable objects.
- ContentUnitFillable: This class is designed for objects that can fill or provide content to specific units or blocks within the application, such as text blocks, images, or videos, ensuring a modular approach to content management. ContentUnitFillable objects are contained in TabFillable objects, which are contained in ShellFillable objects.
- DefaultShellTemplate: This class represents a default template for shells within the application, providing a basic structure that can be customized or extended by other classes for specific needs.
- CustomShellTemplate: This class allows for the creation of custom shell templates, offering flexibility beyond the default templates, enabling specific layouts or functions tailored to particular requirements.
- DefaultTabTemplate: This class provides a default layout and structure for tabs within the application, ensuring a consistent user experience across different sections of the application.
- DefaultContentUnitTemplate: This class offers a default template for content units, ensuring consistency in how content is displayed and managed within the system.
- CustomContentUnitTemplate: This class enables the creation of custom templates for content units, allowing for unique designs or functionalities specific to certain types of content.
- CustomTabTemplate: This class allows for the customization of tab templates, providing the ability to create unique tab designs or functionalities beyond the default settings.
- ShellTemplate: This class acts as a general class for defining templates for shells, potentially serving as a base class for both default and custom shell templates.
- SoftwareData: This class encapsulates data related to the software itself, such as version information, configurations, and settings that are not specific to users or content but are essential for the software's operation.
- TabTemplate: This class serves as a base class for defining templates for tabs within the application, encompassing both default and custom tab designs.
- CustomerInfo: This class contains attributes and methods related to customer information, including personal details, account settings, and preferences, essential for personalizing the customer experience.
- DefaultSettings: This class represents a set of default settings for the application, covering various aspects such as privacy, security, UI preferences, and more, ensuring a baseline configuration that can be customized.
- ContentUnitTemplate: This class serves as a blueprint for creating content units within the system. Content units could be any form of content displayed to the user, such as articles, images, videos, or interactive elements. The template defines the structure, style, and possibly some default content for these units.
- DefaultPrivacySettings: This class contains the application's baseline privacy configurations. This class sets the standard privacy measures that apply to all users upon account creation or system reset, covering aspects like data sharing, visibility settings, and consent requirements.
- DefaultStorageSettings: This class specifies the initial settings for data storage within the application. These settings might include data retention policies, default storage limits, and the organization of user data. It ensures a basic level of data management and efficiency.

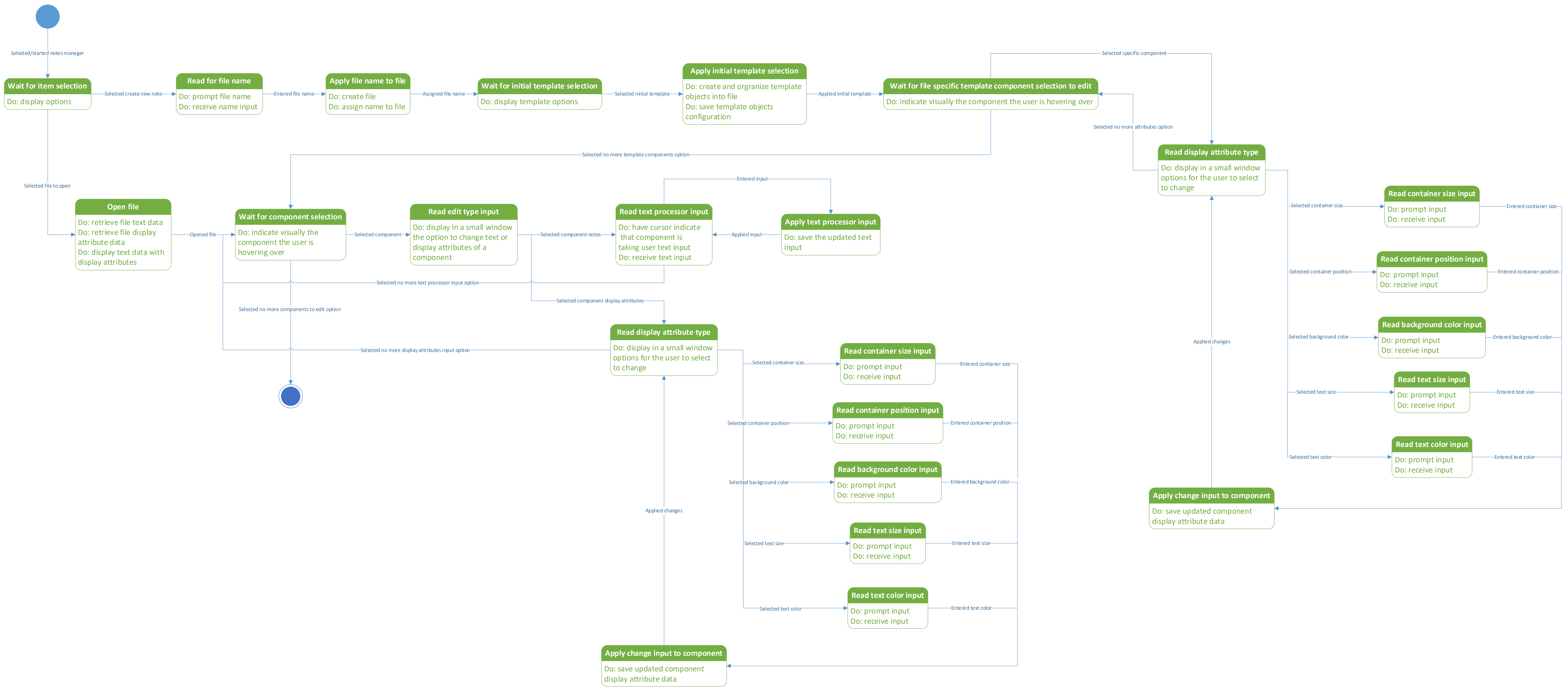
- DefaultSecuritySettings: This class outlines the foundational security protocols and measures for the application. This could encompass password policies, encryption standards, and default access controls, providing a secure starting point for user accounts and data protection.
- DefaultUISettings: This class defines the default user interface preferences and configurations. This class sets the initial look and feel of the application, including themes, layout, and navigation options, ensuring a coherent and user-friendly experience from the start.
- DefaultContentTemplateSettings: This class establishes the basic settings for content templates used within the application. This includes layouts, font styles, and color schemes for content units, ensuring consistency and brand alignment across all content.
- DefaultNotificationSettings: This class provides the initial setup for how notifications are handled and presented in the application. This covers the types of events that trigger notifications, default notification methods (e.g., email, SMS, in-app), and user opt-in settings.
- PrivacySettings: This class acts as a customizable class for user-specific privacy preferences. Users can modify these settings to control how their data is shared, who can view their information, and other privacy-related configurations, offering flexibility beyond the default settings.
- StorageSettings: This class allows users or administrators to adjust storage options according to specific needs. This might include customizing storage limits, managing data backups, and organizing files, providing more control over data management.
- SecuritySettings: This class enables customization of security measures for enhanced protection for each user. Users or administrators can configure settings such as two-factor authentication, custom password policies, and permissions, tailoring security to meet specific requirements.
- UISettings: This class offers a framework for personalizing the user interface for each user. Users can adjust settings related to the application's appearance, layout, and navigation, allowing for a personalized experience that caters to individual preferences.
- ContentTemplateSettings: This class provides a mechanism for creating or modifying content templates specific to each user. This class allows for the customization of content presentation, including layout adjustments, style modifications, and the integration of unique elements, facilitating brand differentiation and creative expression.
- NotificationSettings: This class enables detailed control over notification preferences and behaviors specific to each user. Users can customize which actions trigger notifications, choose preferred notification channels, and set notification frequencies, ensuring they receive relevant information in their preferred manner.
- Settings: This class could act as a container or manager for various types of settings within the application for each user account, providing a centralized point of access and modification for application settings.

Part C

Deliverable 1

Diagram 1: Notetaking Tool

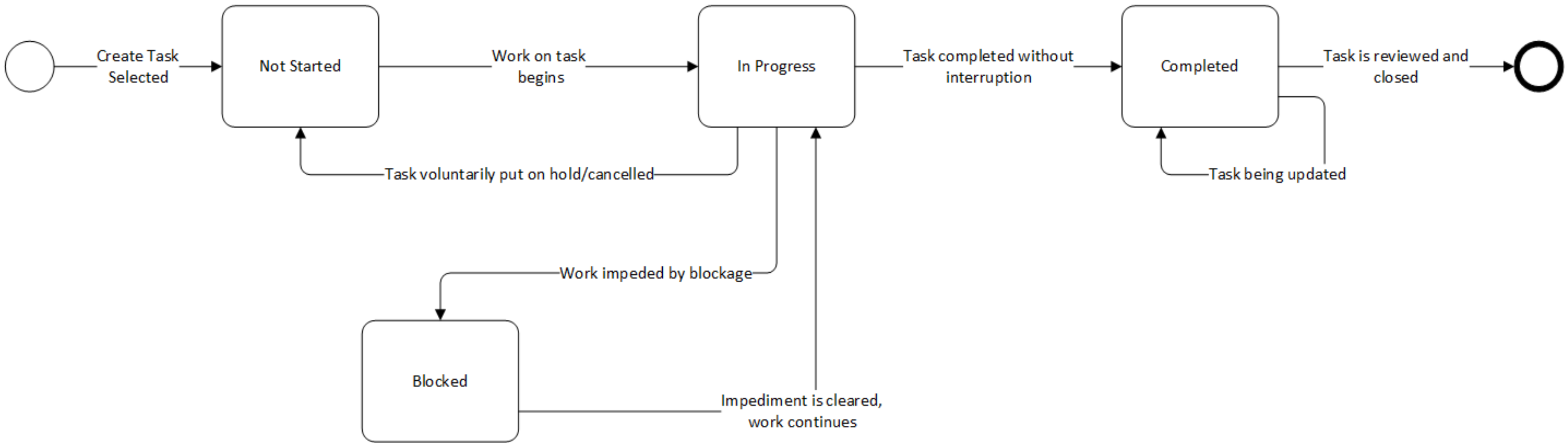
The diagram below is the state diagram for a tool that is intended to allow customers to easily create and edit only notes. After initializing the tool, users have the option to create a new note or edit an existing note. When creating and editing a note file/project, users can edit the format of objects inside the file holding notes. The information that users store, can only be added to a file after a file has already been created.



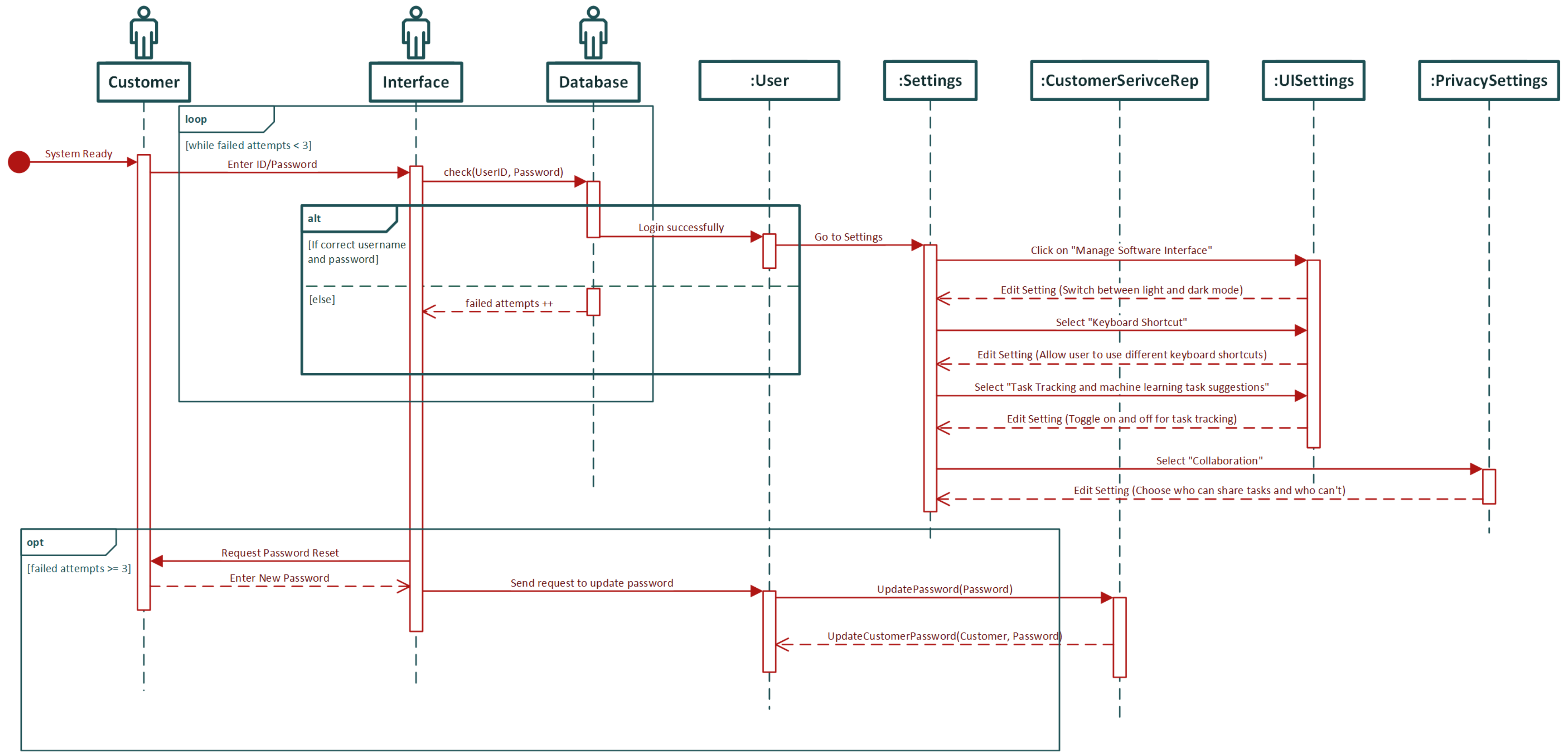
Selected/started notes manager
Wait for item selection
Do: display options
Selected create new note
Read for file name
Do: prompt file name
Do: receive name input
Entered file name
Apply file name to file
Do: create file
Do: assign name to file
Assigned file name
Wait for initial template selection
Do: display template options
Selected initial template
Apply initial template selection
Do: create and orgranize template objects into file
Do: save template objects configuration
Applied initial template
Wait for file specific template component selection to edit
Do: indicate visually the component the user is hovering over
Selected specific component
Selected no more attributes option
Read display attribute type
Do: display in a small window options for the user to select to change
Selected container size
Read container size input
Do: prompt input
Do: receive input
Entered contianer size
Selected container position
Read container position input
Do: prompt input
Do: receive input
Entered container position
Selected background color
Read background color input
Do: prompt input
Do: receive input
Entered background color
Selected text size
Read text size input
Do: prompt input
Do: receive input
Entered text size
Selected text color
Read text color input
Do: prompt input
Do: receive input
Entered text color
Applied changes
Apply change input to component
Do: save updated component display attribute data
Selected no more template components option
Selected file to open
Open file
Do: retrieve file text data
Do: retrieve file display attribute data
Do: display text data with display attributes
Opened file
Wait for component selection
Do: indicate visually the component the user is hovering over
Selected component
Read edit type input
Do: display in a small window the option to change text or display attributes of a component
Selected component notes
Read text processor input
Do: have cursor indicate that component is taking user text input
Do: receive text input
Entered input
Apply text processor input
Do: save the updated text input
Applied input
Selected no more text processor input option
Selected no more components to edit option
Selected component display attributes
Read display attribute type
Do: display in a small window options for the user to select to change
Selected no more display attributes input option
Selected container size
Read container size input
Do: prompt input
Do: receive input
Entered contianer size
Selected container position
Read container position input
Do: prompt input
Do: receive input
Entered container position
Selected background color
Read background color input
Do: prompt input
Do: receive input
Entered background color
Selected text size
Read text size input
Do: prompt input
Do: receive input
Entered text size
Selected text color
Read text color input
Do: prompt input
Do: receive input
Entered text color
Applied changes
Apply change input to component
Do: save updated component display attribute data

Diagram 2: Task Manager Tool

This illustrates the lifecycle of a task within the system, from creation to completion. It includes states like 'In Progress', 'Blocked', and 'Completed', depicting the progression and possible impediments a task may encounter before reaching verification


Create Task Selected
Not Started
Work on task begins
In Progress
Task completed without interruption
Completed
Task is reviewed and closed
Task voluntarily put on hold/cancelled
Task being updated
Work impeded by blockage
Blocked
Impediment is cleared, work continues

Deliverable 2



Customer
Interface
Database
:User
:Settings
:CustomerSerivceRep
:UISettings
:PrivacySettings
loop
[while failed attempts < 3]
System Ready
Enter ID/Password
check(UserID, Password)
alt
[If correct username and password]
Login successfully
Go to Settings
Click on "Manage Software Interface"
Edit Setting (Switch between light and dark mode)
Select "Keyboard Shortcut"
Edit Setting (Allow user to use different keyboard shortcuts)
Select "Task Tracking and machine learning task suggestions"
Edit Setting (Toggle on and off for task tracking)
Select "Collaboration"
Edit Setting (Choose who can share tasks and who can't)
[else]
failed attempts ++
opt
[failed attempts >= 3]
Request Password Reset
Enter New Password
Send request to update password
UpdatePassword(Password)
UpdateCustomerPassword(Customer, Password)

The first event, System Ready, is derived from the external environment. After the system of SAAYJ is totally ready, the actor, Customer, enters UserID and Password to the System, in order to login to his account. A check(UserID, Password) event is passed to the User class, which looks up the UserID and Password in a database. If the username and password match with each other, Customer succeeds in logging into his account, a Go to Settings event is passed to the Settings class, and Customer can edit different setting options on his preference. If not, the number of failed attempts for login increases and Customer re-enters UserID and Password until the number of failed attempts equals to three. If Customer fails for three times, a Request Password Reset event is passed from the System to the Customer. Hence, Customer enters a new password and System send a request to the User class to update password. An UpdatePassword(Password) event is passed to the CustomerServiceRep class and it returns a result (Success or Failure) to the User class.

Deliverable 4

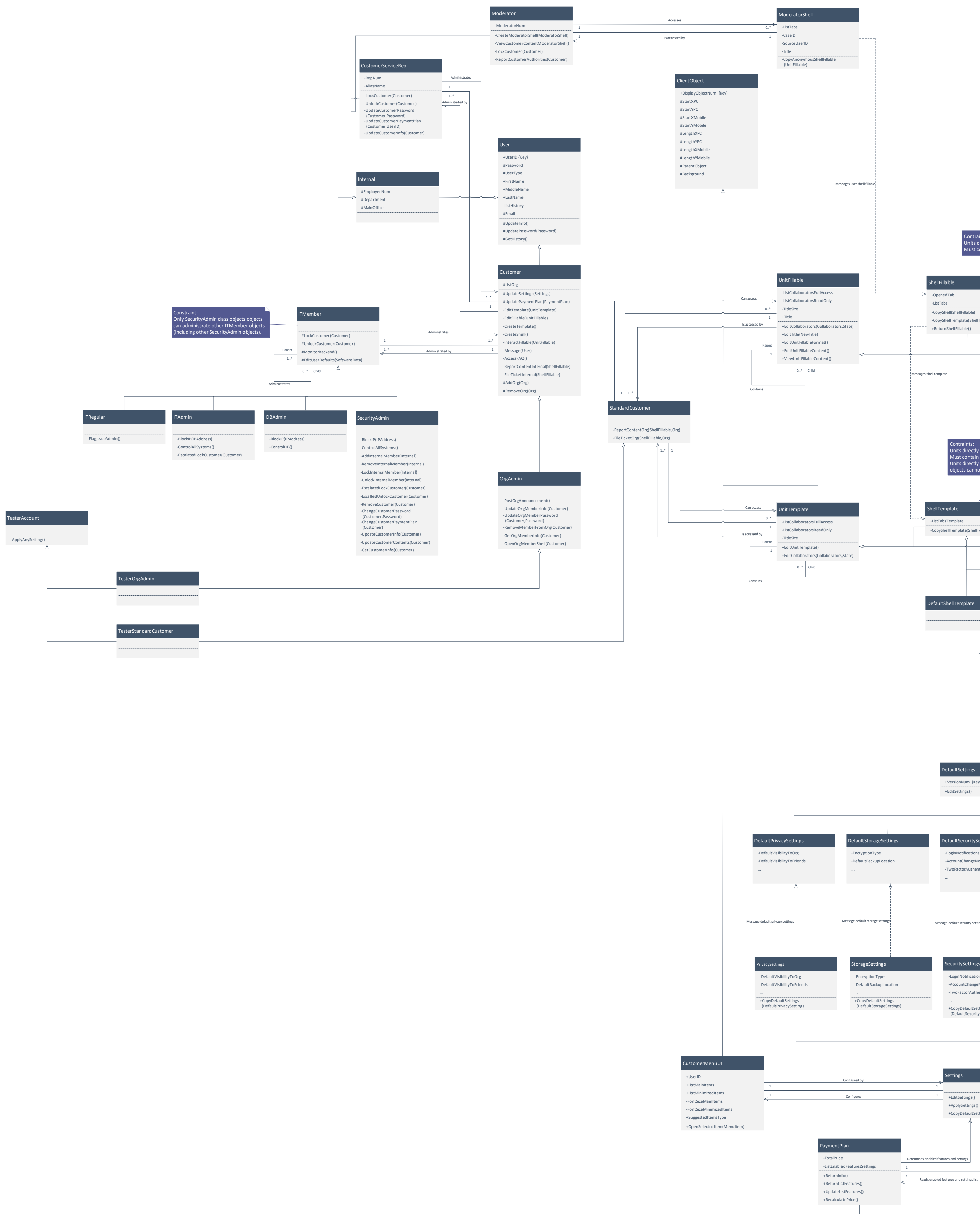
This class diagram in this section is a modified version of the class diagram in Deliverable 6 in Part B. The diagram has been edited to include the party analysis pattern to relevant parts of the diagram, which are parts describing IT member accounts, components storing personal user content, and templates for components storing personal user content. The party analysis pattern is useful for these parts of the diagram because it groups together similar classes under a class and shows how other classes interact with them in similar ways because of the similar classes' shared features. When there are common relationships between the similar classes, they are modelled by an association (some sort of line) connecting the class the similar classes are grouped under to itself to indicate how the similar classes relate to each other; constraints that define the relationship between specific combinations of similar classes are also shown alongside classes in part of those relationships.

In the section for the IT member accounts, the only changes were the constraint and administrative relationship between accounts under it. The constraint is to specify that only security admin level accounts can administrate other IT member accounts. This also means that only security admin level accounts can administrate other security admin level accounts.

The section for the components storing personal user content had the UnitFillable class added to group the ShellFillable, TabFillable, and ContentUnitFillable class together into a party. These classes were designed to have similar client display traits and have a relationship where one contains instances of others. Constraints are linked to each class in the party to describe accurately describe the relationship between specific classes.

The section for templates of components that store personal user content had the UnitTemplate class added to group the ShellTemplate, TabTemplate, and ContentUnitTemplate classes together into a party. These classes were designed to have a relationship where one contains instances of others and have similar client display traits. Constraints are linked to each class in the party to specify the members each class can contain, and to specify further conditions of members of the subclasses of the party classes.

Deliverable 4
Diagram with Party Analysis Pattern



Moderator
-ModeratorNum
-CreateModeratorShell(ModeratorShell)
-ViewCustomerContentModeratorShell()
-LockCustomer(Customer)
-ReportCustomerAuthorities(Customer)
Accesses
Is accessed by
ModeratorShell
-ListTabs
-CaseID
-SourceUserID
-Title
-CopyAnonymousShellFillable (UnitFillable)
CustomerServiceRep
-RepNum
-AliasName
-LockCustomer(Customer)
-UnlockCustomer(Customer)
-UpdateCustomerPassword (Customer,Password)
-UpdateCustomerPaymentPlan (Customer.UserID)
-UpdateCustomerInfo(Customer)
Administrates
Administrated by
ClientObject
+DisplayObjectNum {Key}
#StartXPC
#StartYPC
#StartXMobile
#StartYMobile
#LengthXPC
#LengthYPC
#LengthXMobile
#LengthYMobile
#ParentObject
#Background
User
+UserID {Key}
#Password
#UserType
+FirstName
+MiddleName
+LastName
-ListHistory
#Email
#UpdateInfo()
#UpdatePassword(Password)
#GetHistory()
Internal
#EmployeeNum
#Department
#MainOffice
Messages user shell fillable
Customer
#ListOrg
#UpdateSettings(Settings)
#UpdatePaymentPlan(PaymentPlan)
-EditTemplate(UnitTemplate)
-EditFillable(UnitFillable)
-CreateTemplate()
-CreateShell()
-InteractFillable(UnitFillable)
-Message(User)
-AccessFAQ()
-ReportContentInternal(ShellFillable)
-FileTicketInternal(ShellFillable)
#AddOrg(Org)
#RemoveOrg(Org)
Constraint:
Only SecurityAdmin class objects objects can administrate other ITMember objects (including other SecurityAdmin objects).
ITMember
#LockCustomer(Customer)
#UnlockCustomer(Customer)
#MonitorBackend()
#EditUserDefaults(SoftwareData)
Parent
Child
Adminastrates
UnitFillable
-ListCollaboratorsFullAccess
-ListCollaboratorsReadOnly
-TitleSize
+Title
+EditCollaborators(Collaborators,State)
+EditTitle(NewTitle)
+EditUnitFillableFormat()
+EditUnitFillableContent()
+ViewUnitFillableContent()
Can access
Contains
ShellFillable
-OpenedTab
-ListTabs
-CopyShell(ShellFillable)
+ReturnShellFillable()
Messages shell template
ITRegular
-FlagIssueAdmin()
ITAdmin
-BlockIP(IPAddress)
-ControlAllSystems()
-EscalatedLockCustomer(Customer)
DBAdmin
-BlockIP(IPAddress)
-ControlDB()
SecurityAdmin
-BlockIP(IPAddress)
-ControlAllSystems()
-AddInternalMember(Internal)
-RemoveInternalMember(Internal)
-LockInternalMember(Internal)
-UnlockInternalMember(Internal)
-EscalatedLockCustomer(Customer)
-EscaltedUnlockCustomer(Customer)
-RemoveCustomer(Customer)
-ChangeCustomerPassword (Customer,Password)
-ChangeCustomerPaymentPlan (Customer)
-UpdateCustomerInfo(Customer)
-UpdateCustomerContents(Customer)
-GetCustomerInfo(Customer)
StandardCustomer
-ReportContentOrg(ShellFillable,Org)
-FileTicketOrg(ShellFillable,Org)
Contraints:
OrgAdmin
-PostOrgAnnouncement()
-UpdateOrgMemberInfo(Customer)
-UpdateOrgMemberPassword (Customer,Password)
-RemoveMemberFromOrg(Customer)
-GetOrgMemberInfo(Customer)
-OpenOrgMemberShell(Customer)
TesterAccount
-ApplyAnySetting()
UnitTemplate
-ListCollaboratorsFullAccess
-ListCollaboratorsReadOnly
-TitleSize
+EditUnitTemplate()
+EditCollaborators(Collaborators,State)
ShellTemplate
-ListTabsTemplate
TesterOrgAdmin
DefaultShellTemplate
TesterStandardCustomer
DefaultSettings
+EditSettings()
DefaultPrivacySettings
-DefaultVisibilityToOrg
-DefaultVisibilityToFriends
...
DefaultStorageSettings
-EncryptionType
-DefaultBackupLocation
...
-LoginNotifications
...
Message default privacy settings
Message default storage settings
PrivacySettings
-DefaultVisibilityToOrg
-DefaultVisibilityToFriends
...
+CopyDefaultSettings (DefaultPrivacySettings
StorageSettings
-EncryptionType
-DefaultBackupLocation
...
+CopyDefaultSettings (DefaultStorageSettings)
CustomerMenuUI
+UserID
+ListMainItems
+ListMinimizedItems
-FontSizeMainItems
-FontSizeMinimizedItems
+SuggestedItemsType
+OpenSelectedItem(MenuItem)
Configured by
Configures
Settings
+EditSettings()
+ApplySettings()
PaymentPlan
-TotalPrice
-ListEnabledFeaturesSettings
+ReturnInfo()
+ReturnListFeatures()
+UpdateListFeatures()
+RecalculatePrice()
Determines enabled features and settings
Reads enabled features and settings list