SD²: Software Design Document

1. Project Overview

Niner Connect is an application software that gives students the ability to share and view schedules with other fellow students. They can post their schedules, connect, and communicate with other students who have similar classes. Our stakeholders include professors and students who have desired this in the past. Our software will address the problem by making a site where the students can share their schedules in a social media format and they can also communicate with other students who have the same classes. There will be ways to find other students within the site to connect with and view their schedules by providing the students with profiles.

https://github.com/skiruba/ITSC4155_MDSp24_Group4

Group Number 4
Project Name: Niner connect
Description: Ability to share scehdules in a social media format for users to connect with friends and fellow students with similar schedules.

Team Members Shawn Kiruba, Adam Dauda, Ian York, Stephen Zargo, Nguyen Pham

PRODUCT BACKLOG (TO-DO)

Story #	Card Front	Card Back	Sprint Number	Priority	Assigned To	Comments	Estimated Points	Estimated Hours	Estimated Velocity	Actual Velocity
ST-0	As a <user who wants this functionality> I need a <what the user wants> so that <why the user wants it>.	Acceptance Criteria <what the user should be able to do with the functionality>							(Estimated Points / Estimated Hours)	(Estimated Points / Actual Hours)
ST - 9	As a Moderator, I want the site to contain majority of the courses taught in UNCC	Implement a course catalog	Sprint 3	7th	Adam		3	2	1.50	#REF!
ST - 10	As a Moderator, I want our website database to be secure, so that the users passwords are safe.	Site encryption, SSO, is enabled.	Sprint 3	4th	Stephen		3	2	1.50	#REF!
ST - 11	As a UNCC student I would like to implement a system where i can post stuff for other people to like and comment	Implement a forum system for student interaction	Sprint 3	6th	Shawn		8	6	1.33	#REF!
ST - 12	As a UNCC student, I would like the layout to be clean, so that I can easily navigate it.	User friendly application	Sprint 3	10th	Ian		1	1	1.00	#REF!
ST - 13	As a UNCC student, I would like to be able to create an account, so I can save my preferences.	Profile page implemented	Sprint 3	9th	Nguyen		3	2	1.50	#REF!
ST - 14	As a UNCC student, i would like to be able to follow other students	Impliment a follow feature	Sprint 4	8th	Shawn		5	4	1.25	#REF!

Group Number 4
Project Name: Niner connect
Description: Ability to share scehdules in a social media format for users to connect with friends and fellow students with similar schedules.

Team Members Shawn Kiruba, Adam Dauda, Ian York, Stephen Zargo, Nguyen Pham

DOING

Story #	Card Front	Card Back	Estimated Points	Estimated Hours	Actual Hours	Estimated Velocity	Actual Velocity	Sprint Number	Priority	Assigned To	Comments
ST-0	As a <user who wants this functionality> I need a <what the user wants> so that <why the user wants it>.	Acceptance Criteria <what the user should be able to do with the functionality>				(Estimated Points / Estimated Hours)	(Estimated Points / Actual Hours)				
ST - 4	As a UNCC student, I need to be able to use my UNCC credentials so that I can use the website.	Logged into web site	1	1		1.00	#DIV/0!	Sprint 2	1st	Stephen	
ST - 5	As a UNCC student, I want to be able to search by courses, so that I can search for a match.	Searching on an entered search term categorized by courses, returns results reflecting the search term(s)	3	2		1.50	#DIV/0!	Sprint 2	3rd	Shawn	
ST - 6	As a UNCC student, I want to input my class schedule into the app, including course names, times, and locations, so I can find other students with similar schedules.	Searching for peers using class schedules and course to find other students to study with	5	4		1.25	#DIV/0!	Sprint 2	5th	Ian	
ST - 7	As a UNCC student, I want to be able to chat with the student I am matched with, so we can meet up and study.	Implement a working chat bar	3	2		1.50	#DIV/0!	Sprint 2	2nd	Adam	
ST - 8	As a Moderator and a UNCC student, I want the website to load quickly, so I don't waste time.	Less than 5 second load time between pages	1	1		1.00	#DIV/0!	Sprint 2	11th	Nguyen	

Group Numbe 4
Project Name Niner connect
Description: Ability to share scehdules in a social media format for users to connect with friends and fellow students with similar schedules.

Team Member Shawn Kiruba, Adam Dauda, Ian York, Stephen Zargo, Nguyen Pham

DONE

			Sprint 1							Sprint 2					
Story #	Card Front	Card Back	Sprint Number	Priority*	Estimated Points	Estimated Hours	Actual Hours	Estimated Velocity	Actual Velocity	Estimated Hours	Actual Hours	Estimated Velocity	Actual Velocity	Assigned To	Comments
ST - 0	As a <user who wants this functionality> I need a <what the user wants> so that <why the user wants it>.	Acceptance Criteria <what the user should be able to do with the functionality>						(Estimated Points / Estimated Hours)	(Estimated Points / Actual Hours)			(Estimated Points Estimated Hours)	(Estimated Points / Actual Hours)		
ST - 1	Creating user stories to plan and schedule what needs to get done during		Sprint 1	12th	1	1	1	1.00	1.00					Shawn, Stephen	
ST - 2	Creating low fidelity diagram and the context diagram for our created user stories		Sprint 1	13th	1	1	1	1.00	1.00					Ian, Adam, Nguyen	
ST - 3	Creating UML diagram and some activity diagrams for progression on		Sprint 2	14th	1					1	1	1.00	1.00	Ian, Nguyen	

User Stories:

As a UNCC student, I need to be able to use my UNCC credentials so that I can use the website.

As a UNCC student, I want to be able to search by courses, so that I can search for a match.

As a UNCC student, I want to input my class schedule into the app, including course names, times, and locations, so I can find other students with similar schedules.

As a UNCC student, I want to be able to chat with the student I am matched with, so we can meet up and study.

As a Moderator and a UNCC student, I want the website to load quickly, so I don't waste time.

As a Moderator, I want the site to contain majority of the courses taught in UNCC

As a Moderator, I want our website database to be secure, so that the users passwords are safe.

As a UNCC student, I would like to implement a system where I can post stuff for other people to like and comment.

As a UNCC student, I would like the layout to be clean, so that I can easily navigate it.

As a UNCC student, I would like to be able to create an account, so I can save my preferences.

As a UNCC student, I would like to be able to follow other students

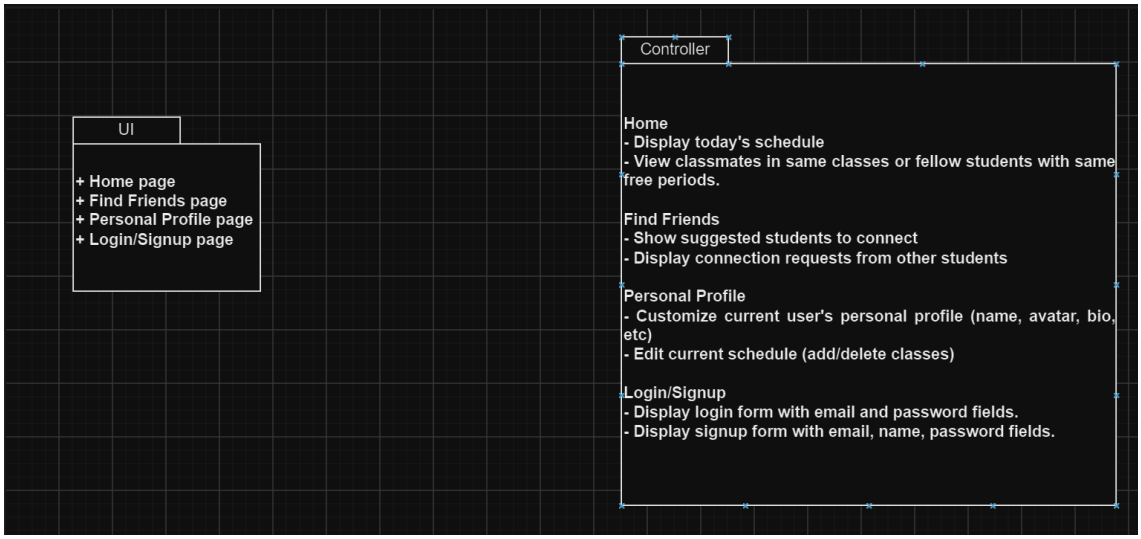
2. Architectural Overview

We went with the client-server architecture because it is straightforward and provides great scalability, security, and reliability that is beneficial to the development process in both short-term and long-term aspects. Basically having a client and server communicating back and forth helps establish the workflow nicely that is easily scalable and debuggable. Other architectures that we considered included microservices and event driven architecture. We decided that client-server architecture made the most sense for this application because having all resources in a centralized location is typical for applications of this nature. Client-server architecture will allow for easy maintenance, scalability, and security. These alternative architectures would've been uncommon for an application of this nature due to the services provided. Our system will include the ability for students to upload their schedule information. This information from all users will be stored on the server, which will allow users anywhere to access the information at the same time. By using a client-server architecture, we are able to take in information from all users, process their information to provide schedule matches and friend recommendations, and then return this output to the client without storing this information locally for each user. Overall, it makes the most sense for the data and processing to take place on a server and return the output to the client. This further allows us to

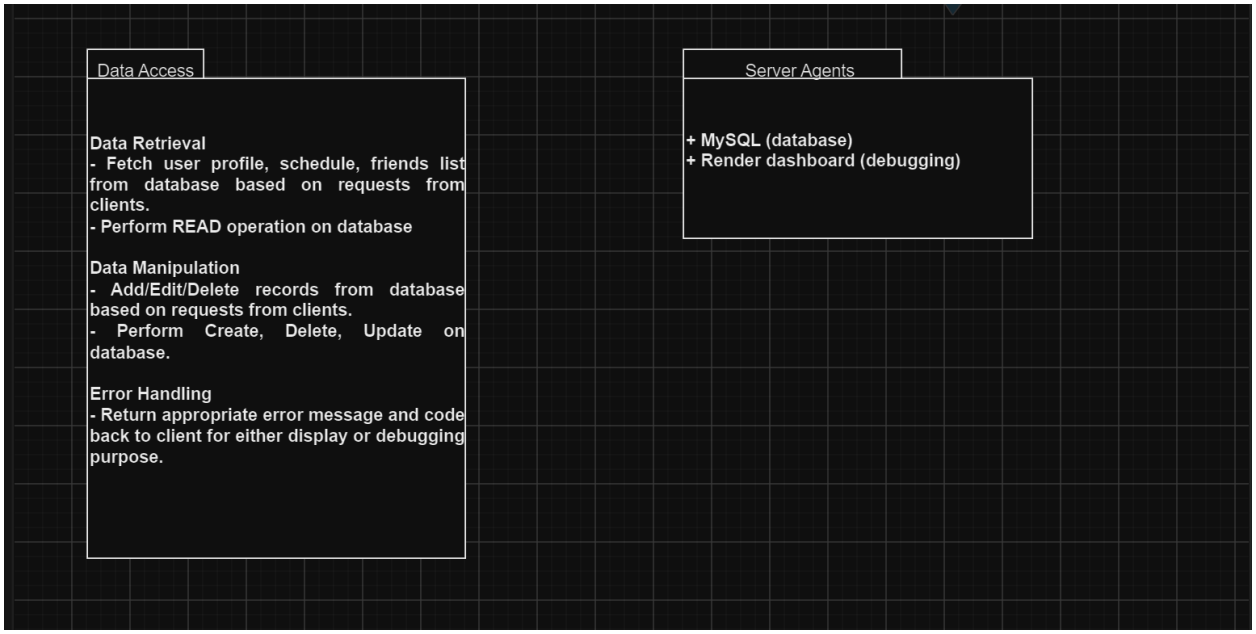
secure important user information, scale the service as it grows, upgrade the service, and simplifying the service for the user.

2.1 Subsystem Architecture

- Client:
    - User interface for students to interact with the application, such as viewing schedules, posting their schedules, and communicating with peers.
    - Handling user input and displaying information received from the server in a user-friendly manner.
    - Managing user authentication and session handling (e.g., logging in, logging out).
    - The client communicates with the server to request data, submit user actions (such as posting schedules), and receive updates.

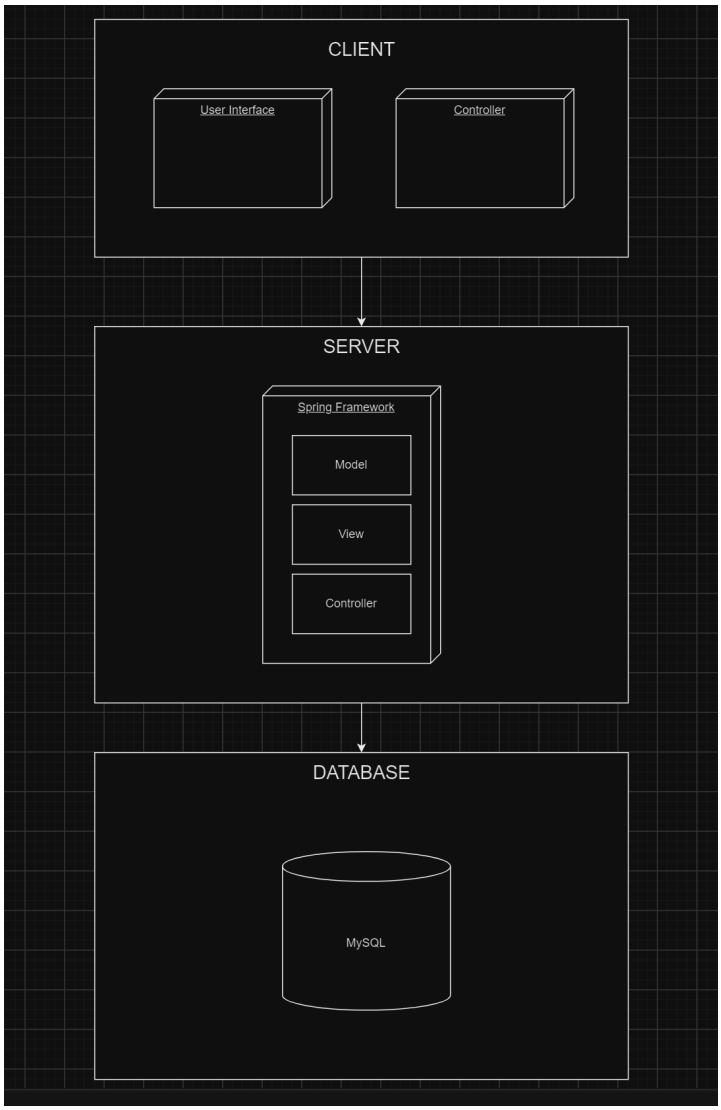


- Server:
    - User authentication: Verifying the identity of users and managing access control to ensure only authorized users can access the system.
    - Schedule management: Storing and organizing schedules posted by students, as well as facilitating schedule sharing between users.
    - Communication management: Handling messages and notifications exchanged between users, such as chat messages or notifications about schedule changes.
    - Data storage: Managing the database where user profiles, schedules, and other relevant data are stored securely.
    - The server processes requests from clients, retrieves or modifies data as needed, and sends responses back to clients.



2.2 Deployment Architecture

The client will handle data from user requests and communicate with the server to establish appropriate actions (displaying data, sending form data). The server will handle user authentication, persist user sessions, and manipulate data based on user requests, utilizing Spring Framework and MVC (Model, View, Controller). The database will be used to store relevant information and communicate with the server via JDBC Driver.

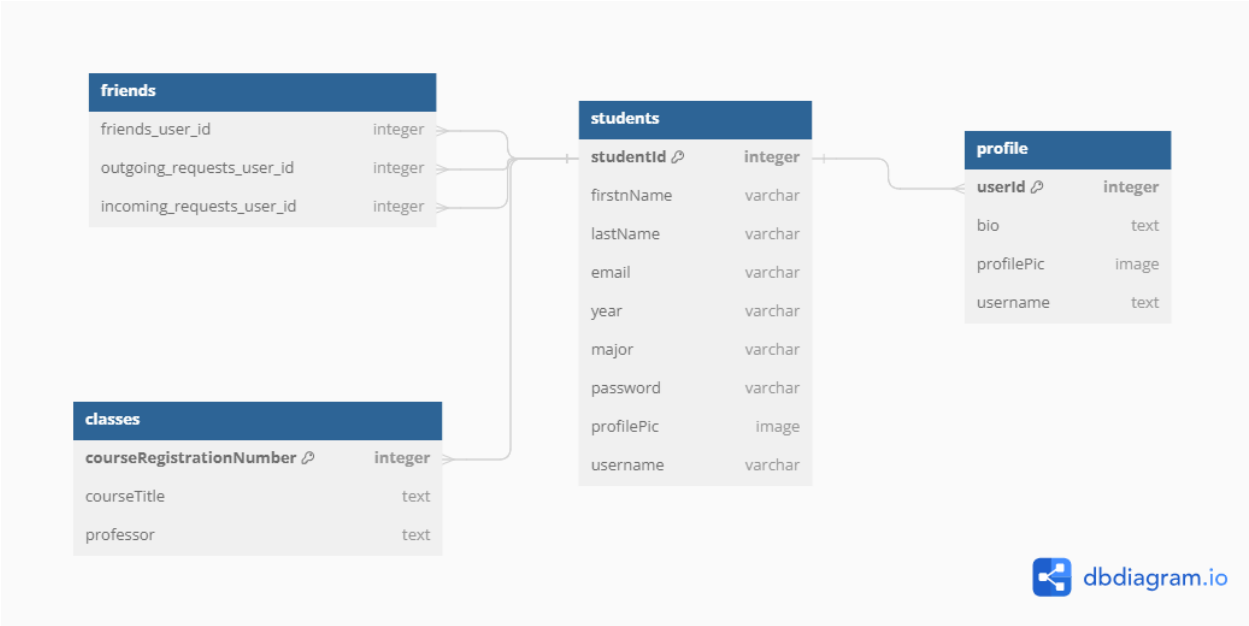


2.3 Data Model

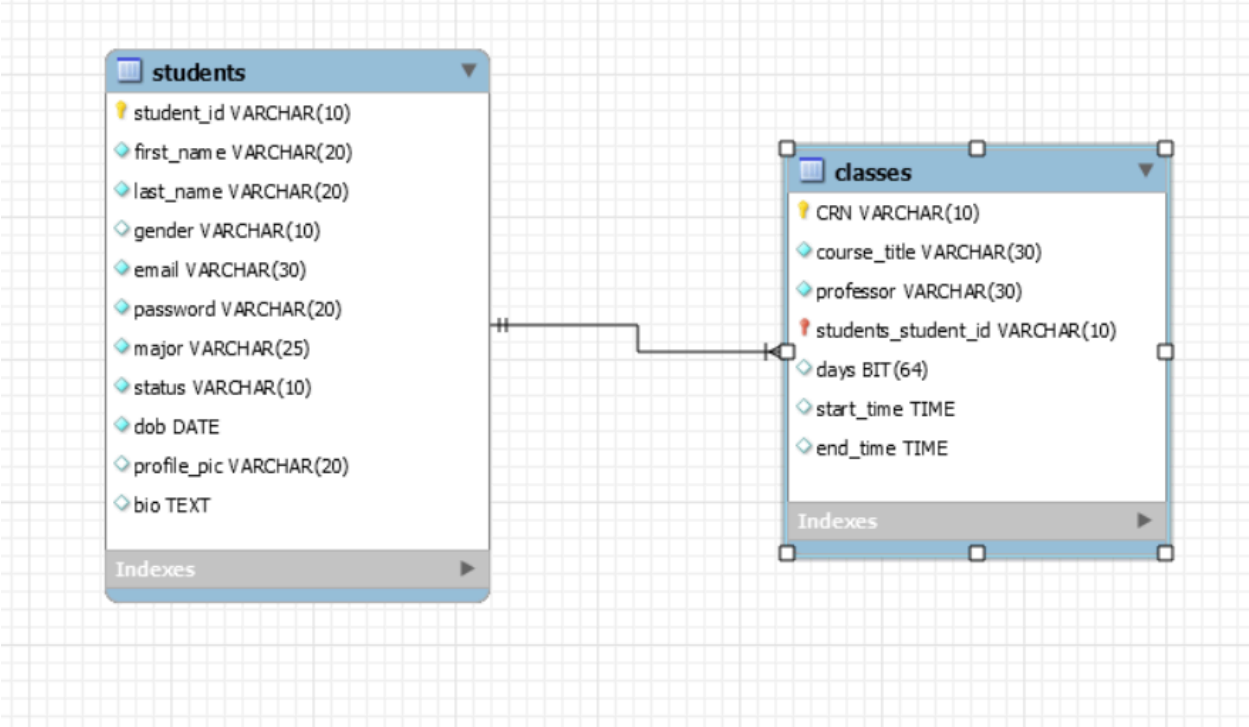
Our system will utilize a SQL database, as depicted in the diagram below. The main table will be Students, which will have many elements. The primary key element will be the student ID. The students table will include all necessary information about each student. This will include their first name, last name, email, year of graduation, major, password, profile picture, username, and their student ID. Each student will have their own profile. The profile table will have their user ID as the primary key, they biography, profile picture, and username. Each student will also be able to add friends. The friends table will include the student's friend's user ID, outgoing friend request user ID, and incoming friend request user ID. This will allow each student to send and receive friend requests, as well as store multiple friends. The primary function of our application is to allow students to add their class schedule. The classes table will store information about

each class, including the CRN as the primary key, the course title, and the professor of the course. This schema is subject to change.

Initial schema: (subject to change up until final deliverable)



Updated schema: (subject to change up until final deliverable)



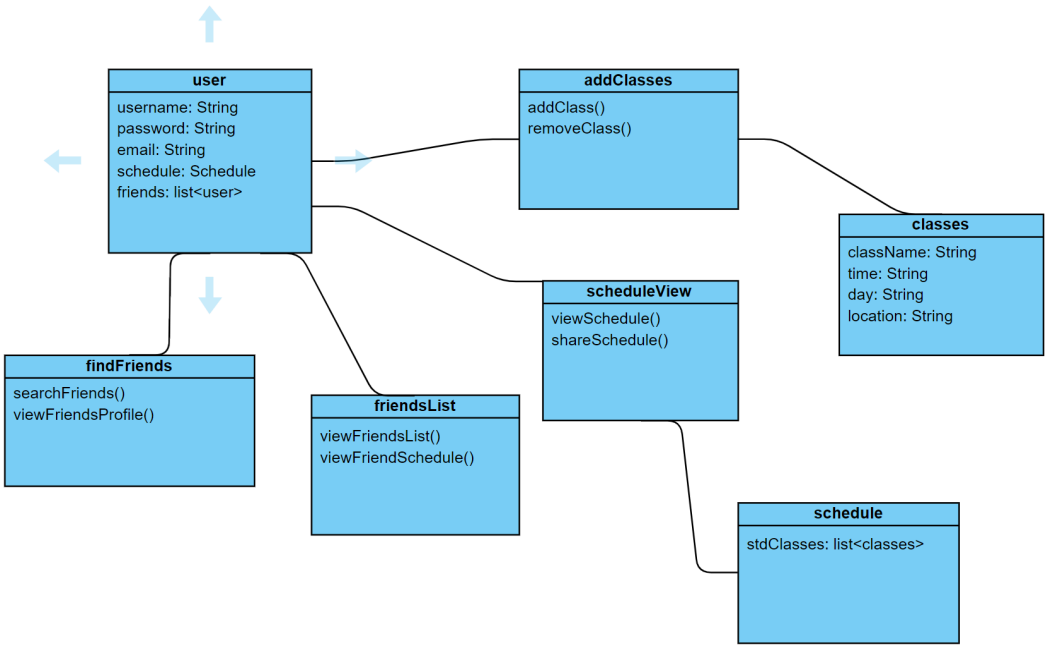
2.4 Global Control Flow

Our system's execution is event-driven. The system will operate based on events prompted by the user. For example, the user may want to add a class. At this point, the system will prompt the user for class information. The user will also be able to choose to view their schedule, view other students, or change their profile information. Each of these scenarios represents events that the system would respond to. Most of these events are caused by user actions, such as button clicks or key presses. There are no time-controlled actions in our application, although there are many event responses inside of the application that will search and find classes and students that are in similar classes. In terms of concurrency, there are a few different components such as the calendar feature which the user can not only use to view their calendar and their classes for the week but they are also able to view the students that they are friends with that are in those same classes.

3. Detailed System Design

The design that we went with for the web application is very straightforward and has pages that are immediately accessible to the user as well as effective for the application. We have a navbar that the students can access where they can visit pages such as home, today's schedule, add classes, find friends, and my friends. These were all made for our website where students can share schedules with their friends while also finding more friends that have similar classes in case they need to reach out to other students.

3.1 Static view



Another critical class in your web application is the user class, which is like the central hub for all users. It is analogous to a digital identity card that keeps all a user's information and identity and based on which they securely log in. If you take the gate to be able to enter a room, the access class would be the gatekeeper. It lets the user perform only those tasks that they are allowed to do and sportingly bans them from anything else. For example, it allows the user to view their schedule or chat with their friends. Next up is the schedule class. It is analogous to the user's organizer on the app. All their classes, assignments, and notes are stored in this class to help them keep track of their student responsibilities. Meanwhile, the Friends class functions as a social connector, making it easy for users to find and interact with each other. Whether you want to form study groups or simply catch up with classmates, this class facilitates those interactions seamlessly. And then there's the class category, which acts as a virtual course folder. It displays all the courses available in the program, allowing users to research and enroll in classes that pique their interest. These classes work harmoniously together to create a dynamic ecosystem where users can manage their academic lives, foster social connections, and explore new learning opportunities—all within a friendly and engaging platform. This robust framework allows users to easily navigate the app and get the most out of their academic journey while making meaningful connections with their peers.

3.2 Dynamic view

You must show the design of your system's behavior using UML sequence diagrams. These sequence diagrams should show the time-ordered sequence of interactions among classes to support an important system function. Your sequence diagrams should be consistent with the class diagrams given in Section 3.1. In other words, you should not have participating objects in an interaction that do not appear in a class diagram; if you find that this is the case, you should go back and revise your class diagram to include the new element. You may supplement your sequence diagrams with state-transition diagrams or activity diagrams (useful for describing algorithms), but these are not required.

Make sure to add your test plans, Previous and current sprints, and brief sprint reviews as well as any additional items you feel are helpful you feel are helpful at any point. Remember, it's not necessary to add documents that are not providing useful information. Only necessary items. If you don't have them ready, you can add them later using version control in Git Hub.