Alex Dixon, Ethen Holzapfel, & Kristen McKinsey Wilson

CSC 440 – Applied Software Engineering

Design specification report

Team Project

Contents

[Figures 2](#_Toc23709470)

[I. Introduction 4](#_Toc23709471)

[A. Problem Statement 4](#_Toc23709472)

[B. Proposal 4](#_Toc23709473)

[II. System Description 4](#_Toc23709474)

[III. System Requirements 5](#_Toc23709475)

[A. Functional Requirements 5](#_Toc23709476)

[B. Non-Function Requirements 21](#_Toc23709477)

[IV. Use Case Diagram 22](#_Toc23709478)

[V. Class Diagram 23](#_Toc23709479)

[VI. Sequence Diagrams 24](#_Toc23709480)

[VII. Activity Diagrams 36](#_Toc23709481)

[VIII. State Diagrams 50](#_Toc23709482)

[IX. Database Design 51](#_Toc23709483)

[A. ER Schema 51](#_Toc23709484)

[B. Table Schema 51](#_Toc23709485)

[X. Conclusion 51](#_Toc23709486)

[XI. Data Dictionary 52](#_Toc23709487)

# Figures

[Figure 1 - Manage Student Workers Button GUI 5](#_Toc23709488)

[Figure 2 - Add New Student Worker Button GUI 5](#_Toc23709489)

[Figure 3 - New Student Worker Information GUI 6](#_Toc23709490)

[Figure 4 - New Student Worker Entered Information GUI 6](#_Toc23709491)

[Figure 5 - View All Student Workers GUI 7](#_Toc23709492)

[Figure 6 - Remove Student Worker Confirmation GUI 7](#_Toc23709493)

[Figure 7 - Manage Subjects Button GUI 8](#_Toc23709494)

[Figure 8 - View All Subjects GUI 8](#_Toc23709495)

[Figure 9 - New Subject Information GUI 9](#_Toc23709496)

[Figure 10 - Create New Subject Confirmation GUI 9](#_Toc23709497)

[Figure 11 - All Subjects With Newly-Added GUI 10](#_Toc23709498)

[Figure 12 - Select Subject GUI 11](#_Toc23709499)

[Figure 13 - Remove Subject Confirmation GUI 11](#_Toc23709500)

[Figure 14 - All Subjects With Newly-Removed GUI 12](#_Toc23709501)

[Figure 15 - View All Student Workers GUI 12](#_Toc23709502)

[Figure 16 - Student Worker Class List GUI 13](#_Toc23709503)

[Figure 17 - Edit Class Information GUI 13](#_Toc23709504)

[Figure 18 - Display Student Workers Button GUI 14](#_Toc23709505)

[Figure 19 - Display Student Workers GUI 14](#_Toc23709506)

[Figure 20 - Switch Views Button GUI 15](#_Toc23709507)

[Figure 21 - Calendar Day View GUI 15](#_Toc23709508)

[Figure 22 - Login GUI 16](#_Toc23709509)

[Figure 23 - New Work Shift GUI 17](#_Toc23709510)

[Figure 24 - Manage Subjects Button GUI 18](#_Toc23709511)

[Figure 25 - Current Subject List GUI 18](#_Toc23709512)

[Figure 26 - Select Subject GUI 19](#_Toc23709513)

[Figure 27 - Subject Flyer GUI 19](#_Toc23709514)

[Figure 28 - Student ID Input GUI 20](#_Toc23709515)

[Figure 29 - Student Worker GUI 21](#_Toc23709516)

[Figure 30 - Use Case Diagram 22](#_Toc23709517)

[Figure 31 - Class Diagram 23](#_Toc23709518)

[Figure 32 - Add Student Worker Sequence Diagram 24](#_Toc23709519)

[Figure 33 - Edit Own Class Schedule Sequence Diagram 25](#_Toc23709520)

[Figure 34 - View Own Schedule Sequence Diagram 26](#_Toc23709521)

[Figure 35 - Login Sequence Diagram 27](#_Toc23709522)

[Figure 36 - View Student Worker Schedules Sequence Diagram 28](#_Toc23709523)

[Figure 37 - View Availability Schedule Sequence Diagram 28](#_Toc23709524)

[Figure 38 - Edit Work Schedule Sequence Diagram 29](#_Toc23709525)

[Figure 39 - Create Subject Flyer Sequence Diagram 30](#_Toc23709526)

[Figure 40 - Check Schedule Sequence Diagram 30](#_Toc23709527)

[Figure 41 - Edit Class Schedule Sequence Diagram 31](#_Toc23709528)

[Figure 42 - Edit Student Worker Info Sequence Diagram 32](#_Toc23709529)

[Figure 43 - Remove Student Worker Sequence Diagram 33](#_Toc23709530)

[Figure 44 - Add Subject Sequence Diagram 34](#_Toc23709531)

[Figure 45 - Remove Subject Sequence Diagram 35](#_Toc23709532)

[Figure 46 - Add Student Worker Activity Diagram 36](#_Toc23709533)

[Figure 47 - Add Subject Activity Diagram 37](#_Toc23709534)

[Figure 48 - Check Schedule Activity Diagram 38](#_Toc23709535)

[Figure 49 - Create Subject Flyer Activity Diagram 39](#_Toc23709536)

[Figure 50 - Edit Class Schedule Activity Diagram 40](#_Toc23709537)

[Figure 51 - Edit Own Class Schedule Activity Diagram 41](#_Toc23709538)

[Figure 52 - Edit Student Worker Info Activity Diagram 42](#_Toc23709539)

[Figure 53 - Edit Work Schedule Activity Diagram 43](#_Toc23709540)

[Figure 54 Login Activity Diagram 44](#_Toc23709541)

[Figure 55 Remove Student Worker Activity Diagram 45](#_Toc23709542)

[Figure 56 Remove Subject Activity Diagram 46](#_Toc23709543)

[Figure 57 View Availability Schedule Activity Diagram 47](#_Toc23709544)

[Figure 58 View Student Worker's Schedules Activity Diagram 48](#_Toc23709545)

[Figure 59 View Own Schedule Activity Diagram 49](#_Toc23709546)

[Figure 60 TutorScheduler State Diagram 50](#_Toc23709547)

[Figure 61 TutorScheduler ER Diagram 51](#_Toc23709548)

# Introduction

## Problem Statement

The Eastern Kentucky University Student Success Center employs around 35 EKU students each semester for the position of EKU Guru. EKU Gurus are student tutors and mentors who work in the Student Success Center in the library and at the Guru Hub in the Whitlock building. Gurus help students with over 200 subjects, including Gen. Ed. courses and many major courses from English and history to chemistry and computer science. Since Gurus are also students, they must work around their schedule class meeting times and follow certain schedule guidelines implemented by the university to ensure student workers are successful both as employees and as students. With the magnitude of this workforce and the complex schedule regulations for Gurus, developing the work schedule for each employee every semester becomes a challenging task.

## Proposal

To address this issue, we propose a software system to help simplify the complicated task of designing semester-long work schedules for student employees at the Student Success Center. The system will provide the user with an easy to use interface that will make creating the student worker schedule simpler and more visually appealing. Users will be able to input student workers’ class schedules into the system. The system will verify that the schedule the user creates meets all the requirements set forth by the university. For example, the system will verify that each student worker has at least a 15-minute break between a class and a work shift. The system will also ensure that no student worker works a shift more than 5 hours without taking a 30-minute break. The system will also verify that the schedule fulfills the scheduling guidelines for the Student Success Center. For example, the Success Center wants to have a math tutor on the schedule at all times, so the system will alert the user if this is not met. The system will also use the created schedule to generate flyers showing the days and times tutors are available for each subject.

# System Description

The system will allow the manager to input student workers’ class schedules through an intuitive, form-like interface so that this task is as simple and fast as possible for the manager. Additionally, a separate web form will allow student workers to input their own class schedule that will be saved in the system to reduce the manager’s workload and simplify the class schedule input for student workers. The system will also allow the manager to design the work schedule for each student worker using a weekly or daily calendar interface. This will allow the manager to visualize the class and work schedule for each student to simplify the scheduling process. The system will also analyze the schedule to check for certain conditions, such as tutor subject coverage and maximum shift times and notify the manager of any errors or discrepancies that need attention. Finally, the system will allow the manager to produce a schedule flyer by subject that lists all the times that at least one tutor is available for a specific subject. These capabilities will be available to the user through a Windows application.

# System Requirements

## Functional Requirements

R1. The system shall allow the manager to add a new student worker to the system.

R1.1. The manager shall click the Student Workers -> Manage Student Workers button on the tool strip.

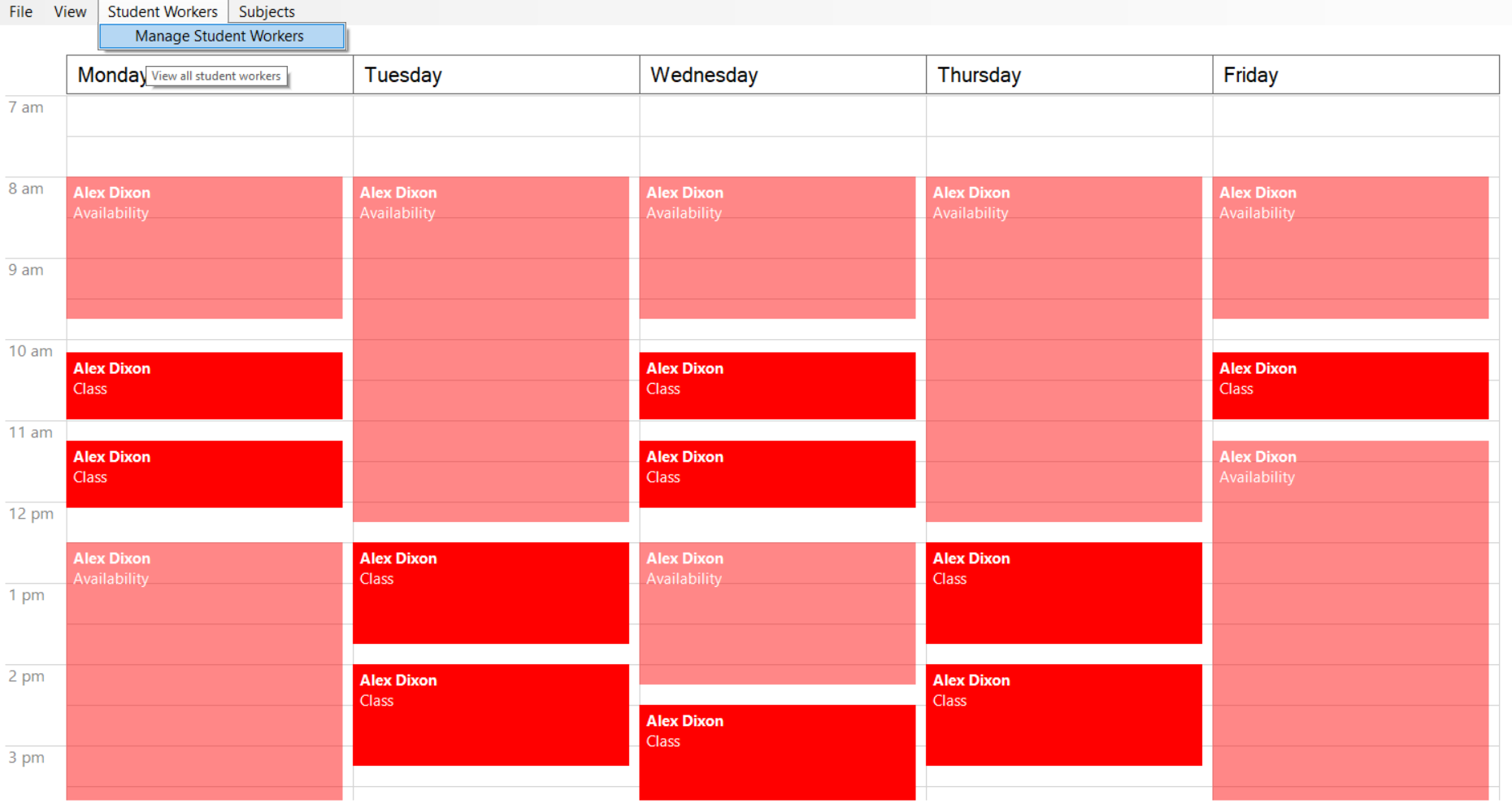


Figure 1 - Manage Student Workers Button GUI

R1.2. The system shall display the current list of student workers.

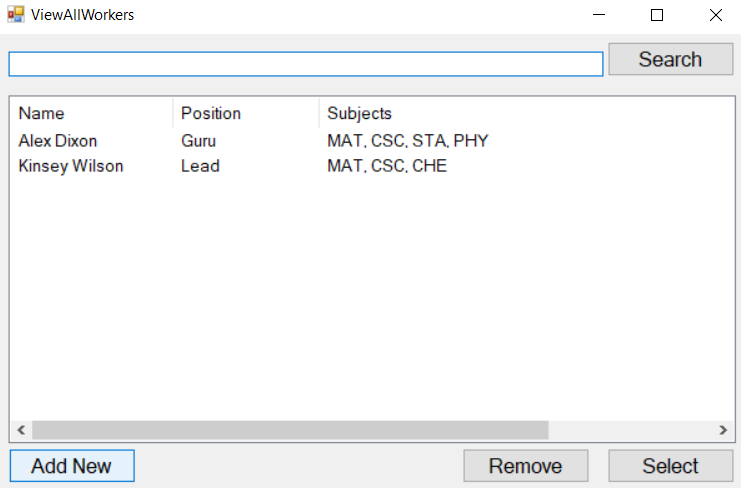


Figure 2 - Add New Student Worker Button GUI

R1.3. The manager shall click the Add New button.

R1.4. The system shall display a form to enter information about the new student worker.

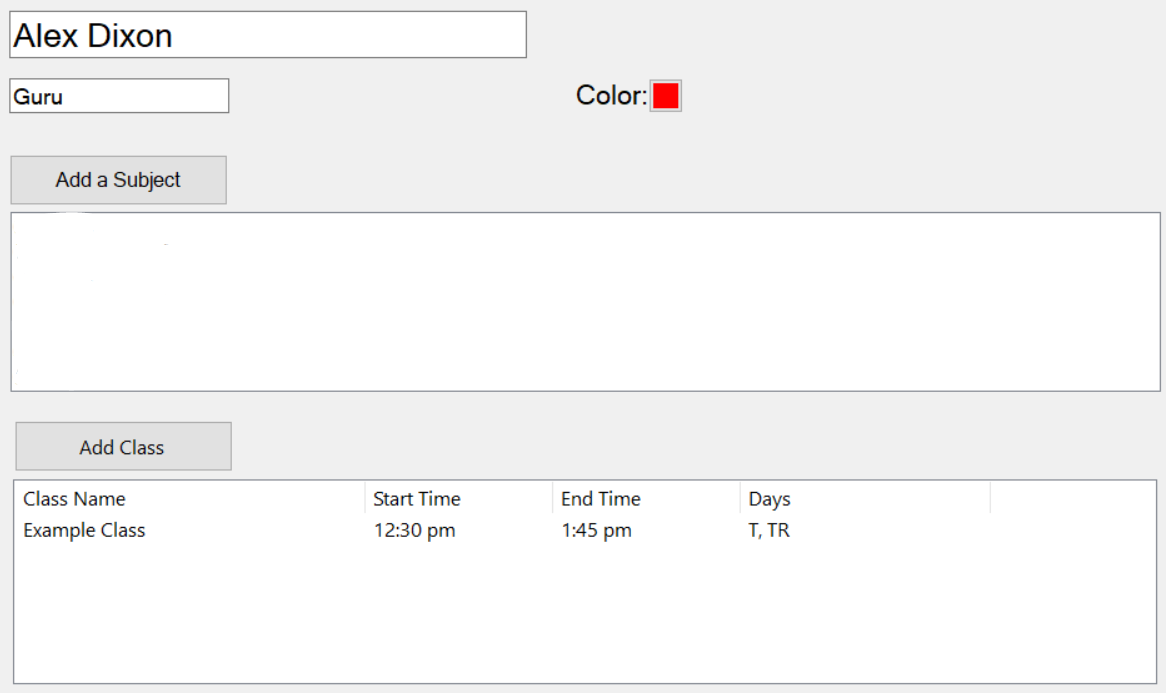


Figure 3 - New Student Worker Information GUI

R1.5. The manager shall enter the student worker information, such as name, position, subjects covered, etc., then close the popup window.

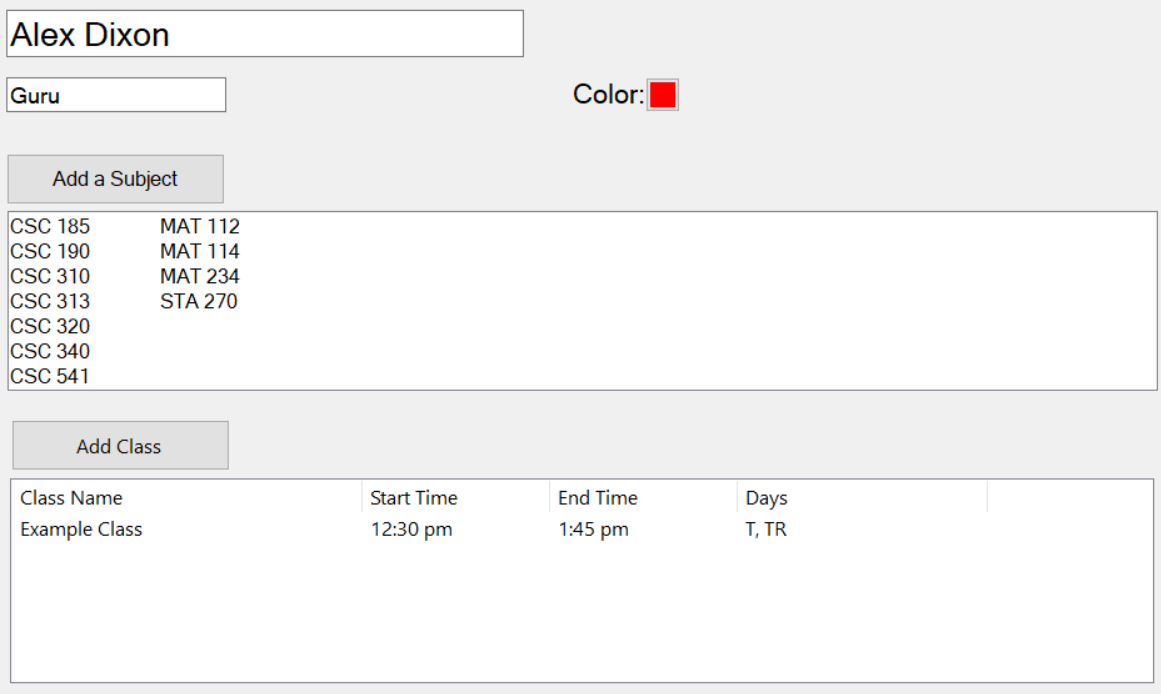


Figure 4 - New Student Worker Entered Information GUI

R1.6. The system shall save the information and return to the main schedule view.

R2. The system shall allow the manager to edit a student worker’s information.

R2.1 The manager shall click the Student Workers -> Manage Student Workers button on the main menu, as shown in R1.1.

R2.2. The system shall display the current list of student workers.

R2.3. The manager shall select a student worker from the list and click the Select button.

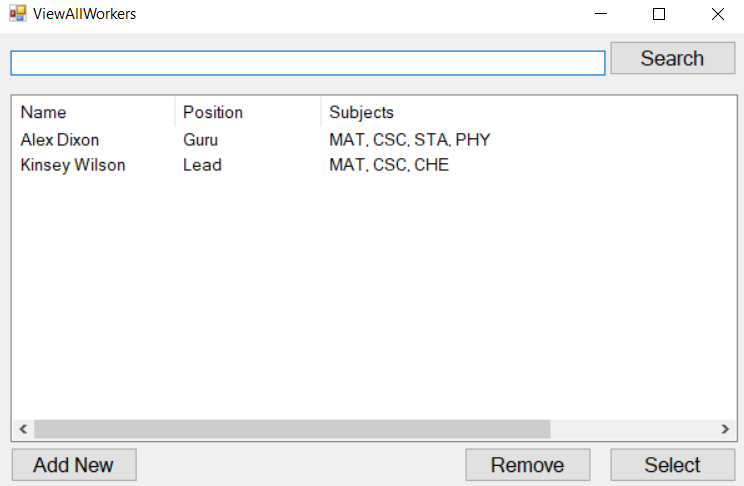


Figure 5 - View All Student Workers GUI

R2.4. The system shall display the selected student worker’s information, as shown in Figure 3.

R2.5. The manager shall update the information then close the form.

R2.6. The system shall save the updated information and return to the student workers form that will show the updated information.

R3. The system shall allow the manager to remove a student worker from the system.

R3.1. The manager shall view the list of student workers by the steps of R2.1-R2.2.  
R3.3. The manager shall select a student worker from the list and click the Remove button.  
R3.4. The system shall request confirmation for the choice to remove the selected student worker.

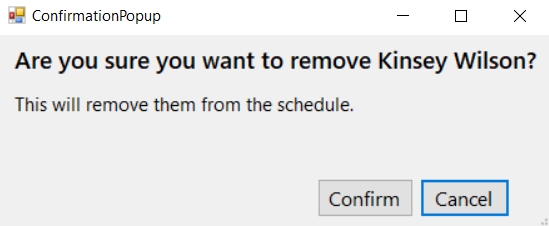


Figure 6 - Remove Student Worker Confirmation GUI

R3.5. The manager shall confirm the chosen action.  
R3.6. The system shall remove the student worker from the schedule.

R4. The system shall allow the manager to add a subject to the subject list.

R4.1. The manager shall click the Subjects -> Manage Subjects button on the main menu.

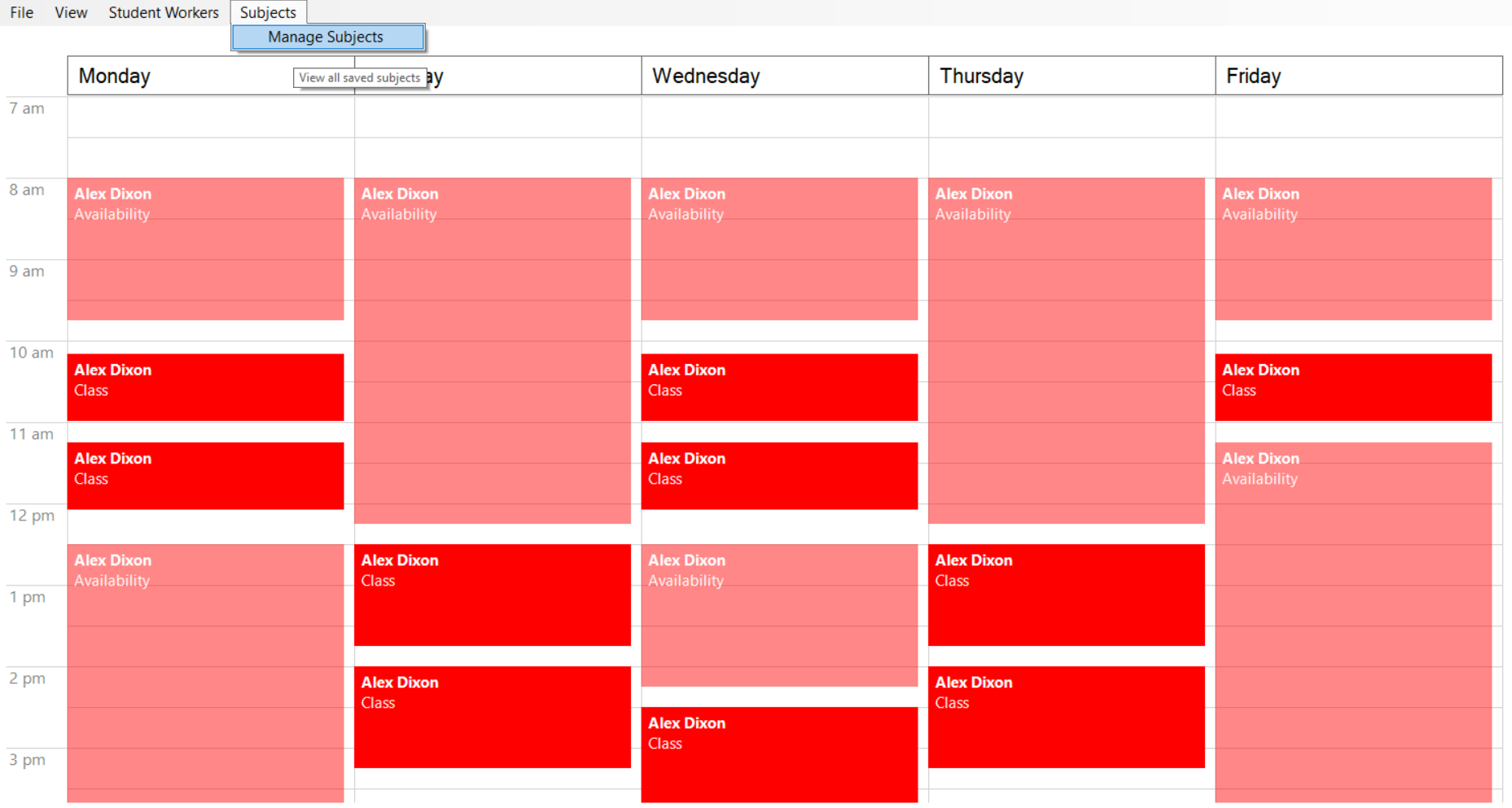


Figure 7 - Manage Subjects Button GUI

R4.2. The system shall display the current list of subjects.  
R4.3. The manager shall click the Add a Subject button.

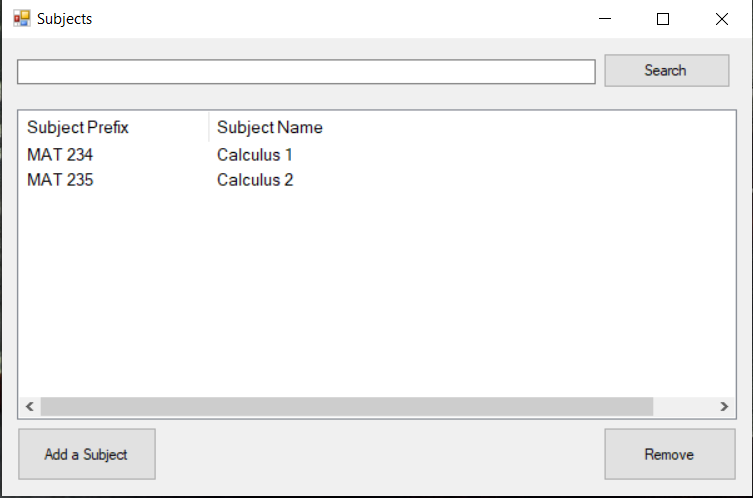


Figure 8 - View All Subjects GUI

R4.4. The system shall display the form to enter information about the subject, such as name, three-letter prefix, and number.  
R4.5. The manager shall enter the information then click the Save button.

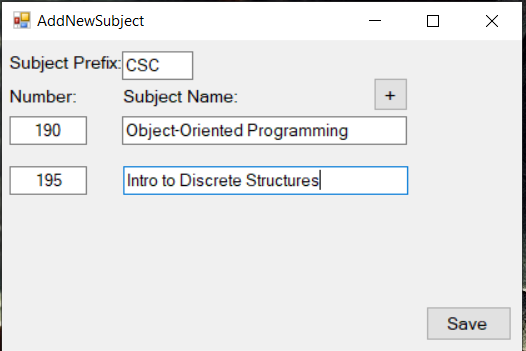


Figure 9 - New Subject Information GUI

R4.6. The system shall ask for confirmation that the manager wants to add the new subject.

R4.7. The manager shall click the Add button.

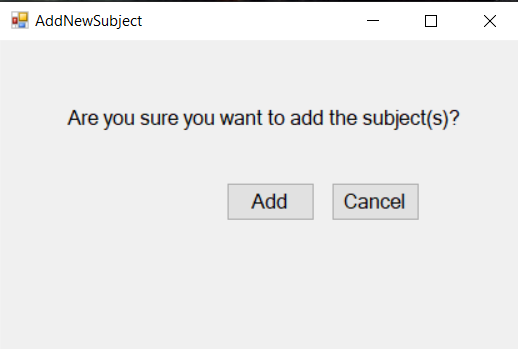


Figure 10 - Create New Subject Confirmation GUI

R4.8. The system shall add the new subject(s).

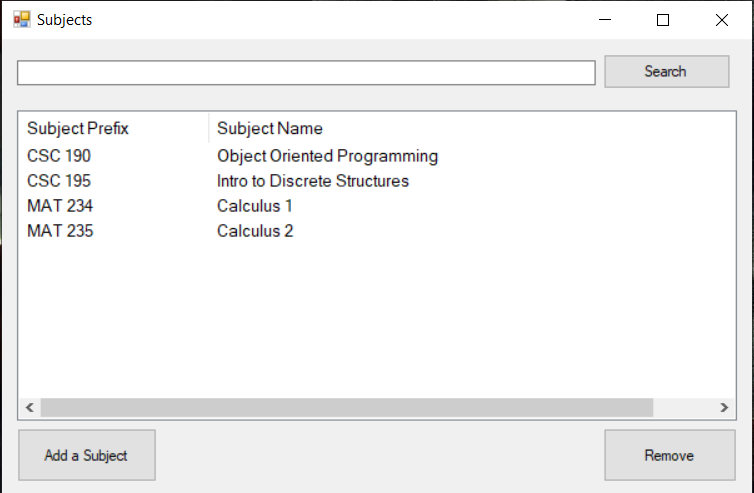


Figure 11 - All Subjects With Newly-Added GUI

R5. The system shall allow the manager to remove a subject from the subject list.

R5.1. The manager shall click the Subjects -> Manage Subjects button on the tool strip, as shown in R4.1.

R5.2. The system shall display the current list of subjects.

R5.3. The manager shall select a subject from the list then click the Remove button.

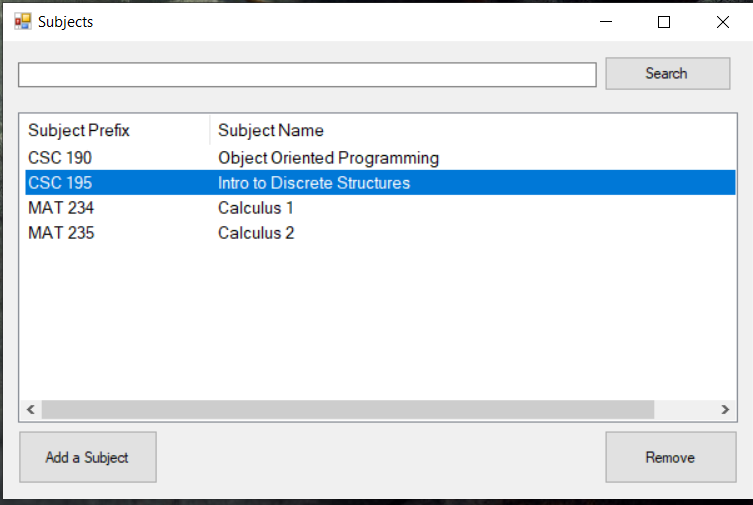


Figure 12 - Select Subject GUI

R5.4. The system shall request confirmation of the action.

R5.5. The manager shall confirm the action to remove the subject.

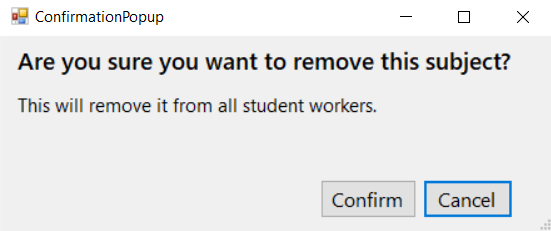


Figure 13 - Remove Subject Confirmation GUI

R5.6. The system shall remove the subject from the list.

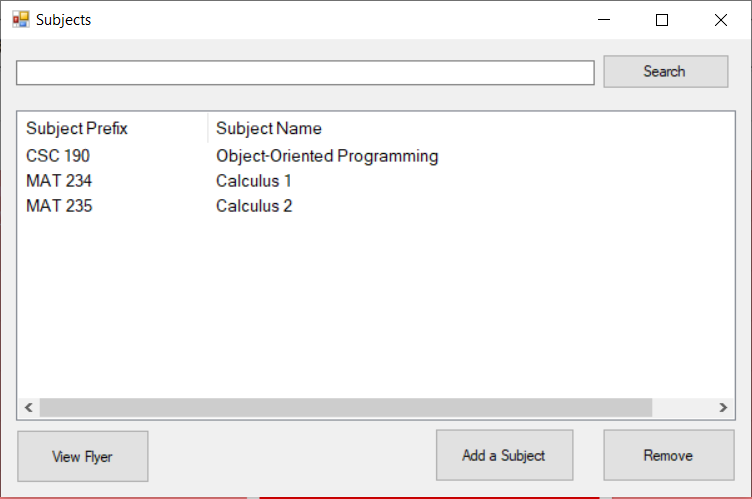


Figure 14 - All Subjects With Newly-Removed GUI

R6. The system shall allow the manager to edit a weekly schedule of class meeting times for a student worker.

R6.1. The manager shall click the Student Workers -> Manage Student Workers button on the main menu, as shown in R1.1.  
R6.2. The system shall display the current list of student workers.  
R6.3. The manager shall select a student worker from the list whose class schedule they wish to edit.

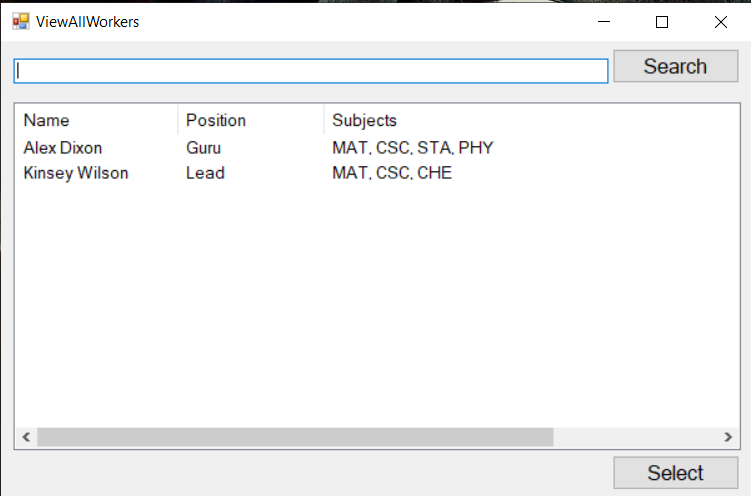


Figure 15 - View All Student Workers GUI

R6.4. The system shall display the selected student worker’s classes.

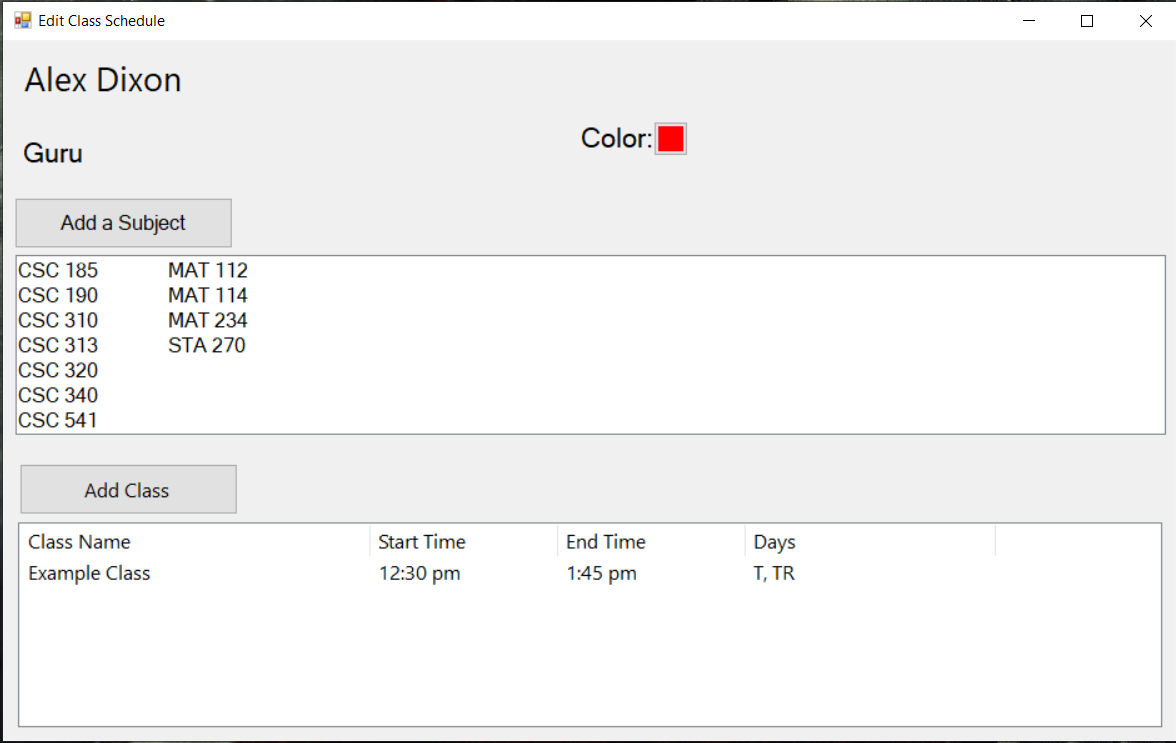


Figure 16 - Student Worker Class List GUI

R6.5. The manager shall select a class to edit or click the Add Class button to create a new class.  
R6.6. The system shall display the form to enter information for the class.

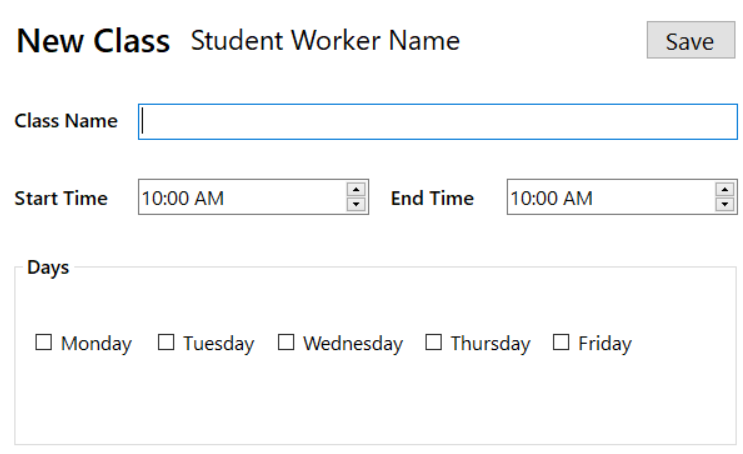


Figure 17 - Edit Class Information GUI

R6.7. The manager shall enter the information and click the Save button.  
R6.8. The system shall save the new information and return to the list of classes, which will display the updated information.

R7. The system shall allow the manager to select certain student workers by criteria such as subject tutored, job position, and name, and view a weekly or single-day schedule for those student workers.

R7.1. The manager shall click the View -> Display Student Workers button from the main menu.

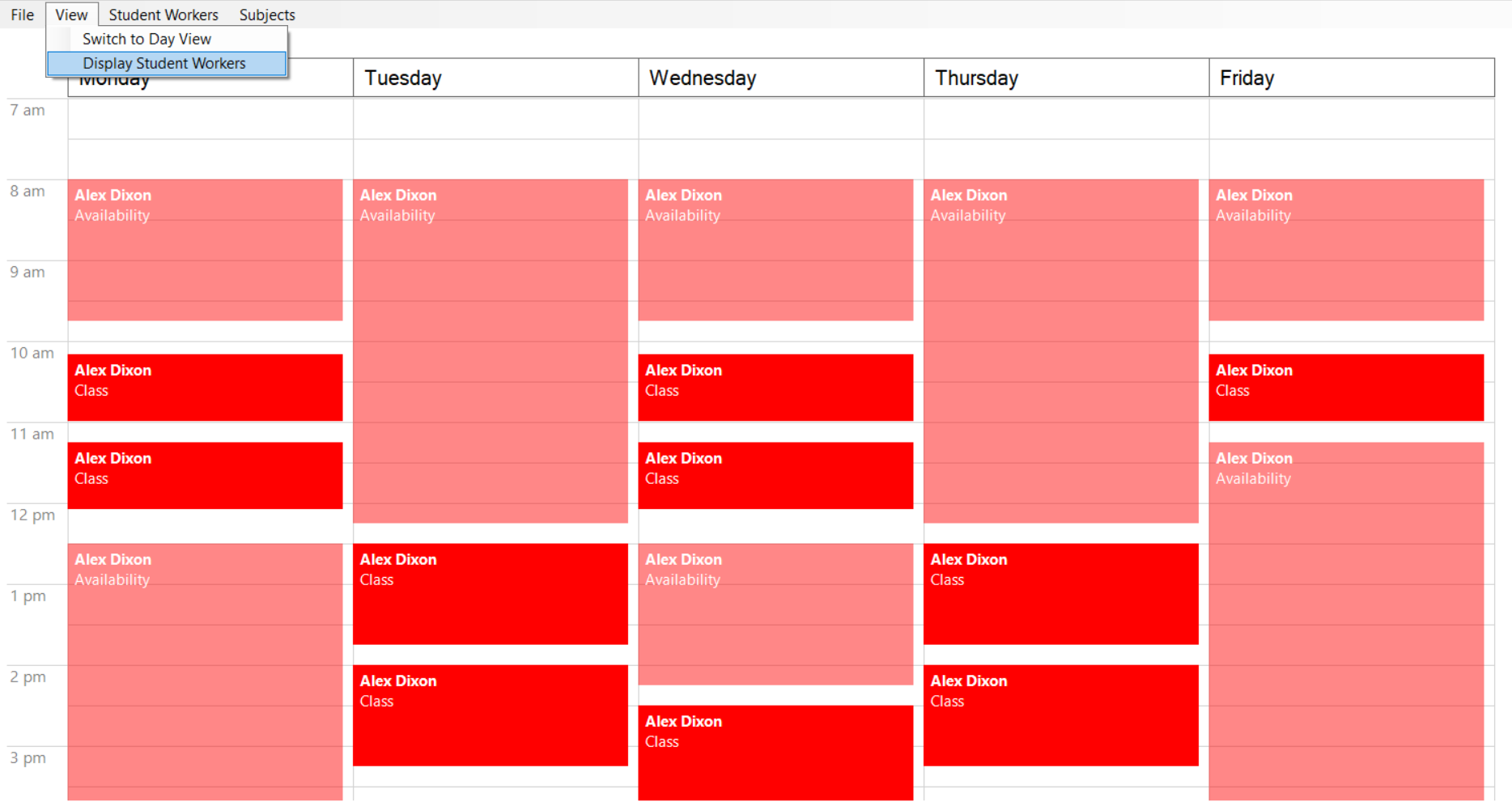


Figure 18 - Display Student Workers Button GUI

R7.2. The system shall display the list of all current student workers.  
R7.3. The manager may select a subset of student workers whose schedules they wish to view or edit.

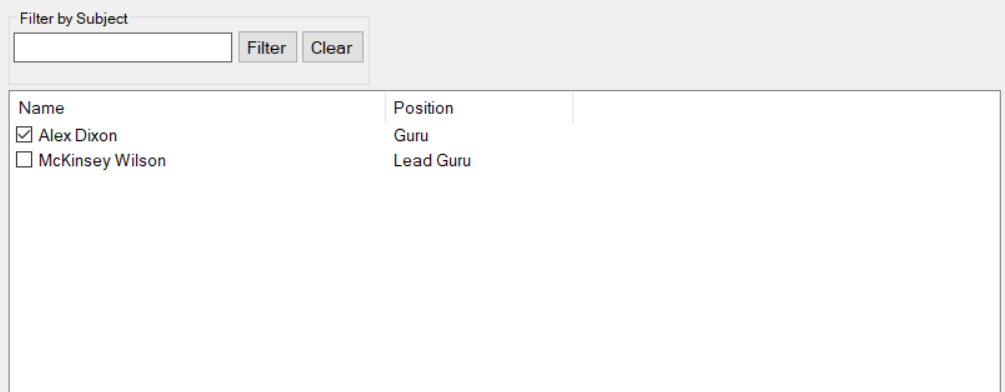


Figure 19 - Display Student Workers GUI

R7.4. The system shall display the schedule for the selected student workers.  
R7.5. The manager may select between a daily schedule or weekly schedule view.

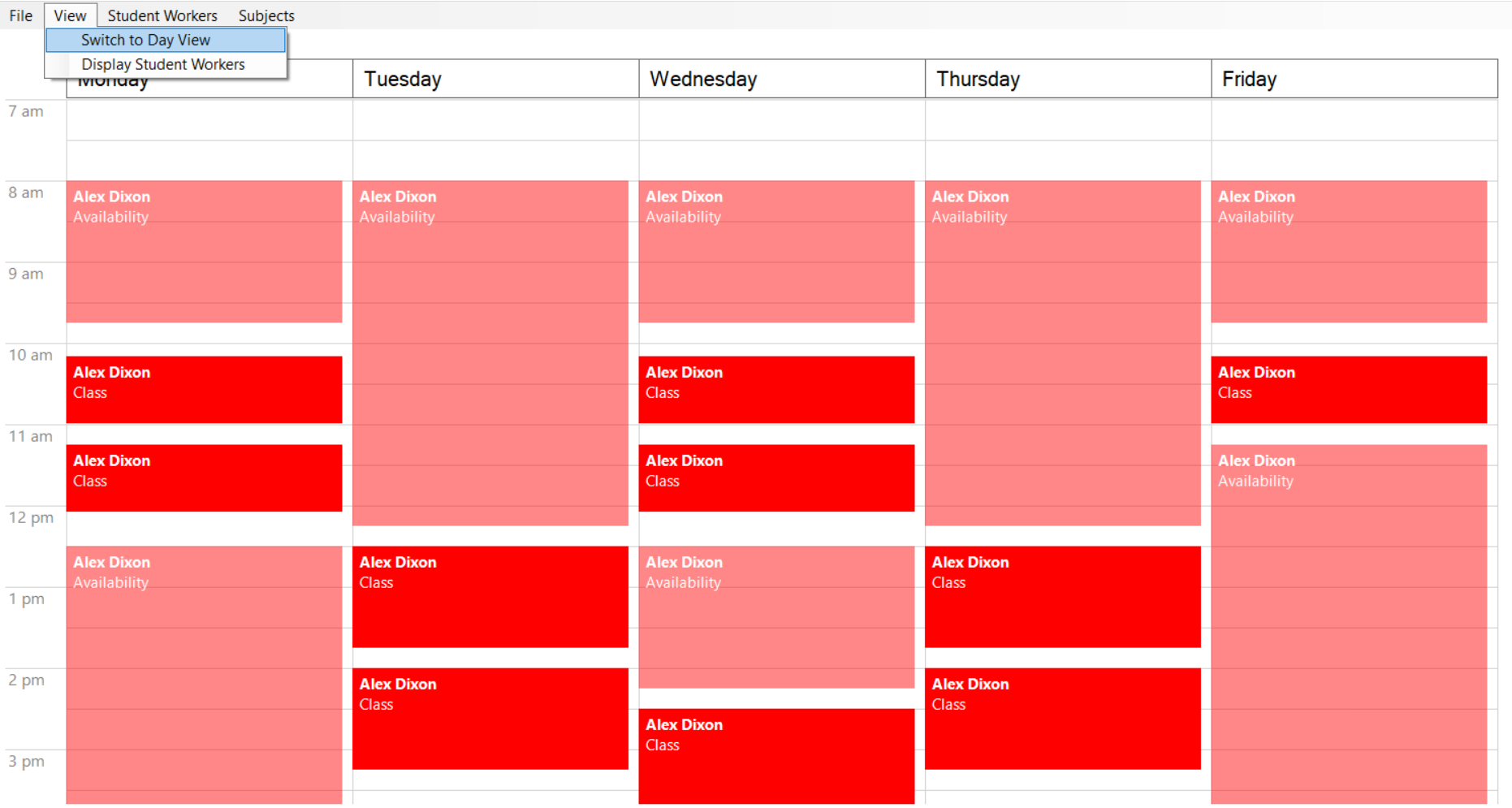


Figure 20 - Switch Views Button GUI

R7.6. The system shall display the selected schedule view.



Figure 21 - Calendar Day View GUI

R8. The system shall allow a manager to log into or log out of the system.

R8.1. The user shall enter their credentials and click the log in button.

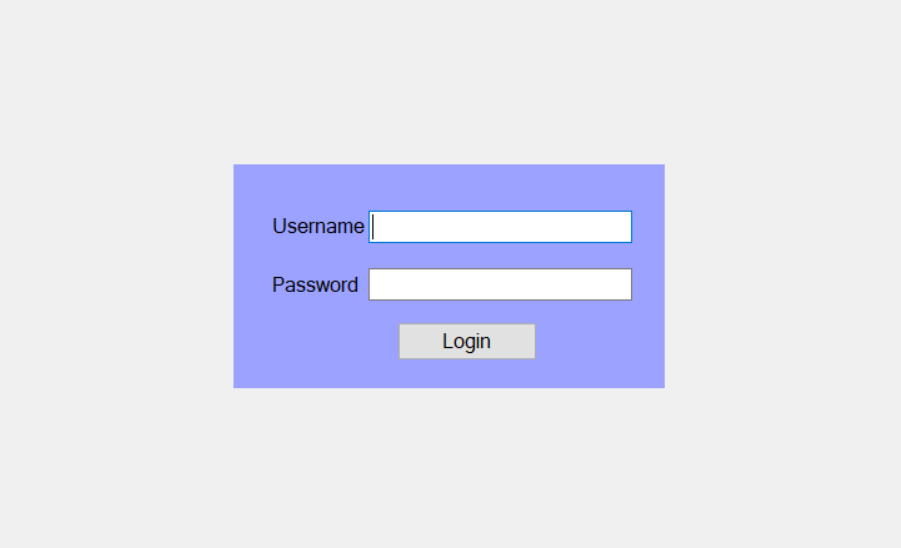


Figure 22 - Login GUI

R8.2. The system shall verify the user’s credentials.

R8.2.1. If the credentials are incorrect or invalid, the system shall display a message to inform the user.

R8.3. The system shall display the appropriate main menu depending on the type of user.

R9. The system shall provide the manager a visual representation of the times each week that a student worker is available to work based on their class schedule and the open hours of the work location.

R9.1. The manager shall select student workers whose availability schedule they wish to view by the steps of R7.  
R9.2. The system shall calculate and display the times for each weekday that a student worker is available to work.

R10. The system shall allow the manager to specify work times for student workers on the schedule.

R10.1. The manager shall click on the schedule to create a new work event.

R10.2. The system shall display a form for the manager to enter details about the work event.

R10.3. The manager shall enter in the necessary information for the work event, including start time, end time, and student worker, then click the Create button.

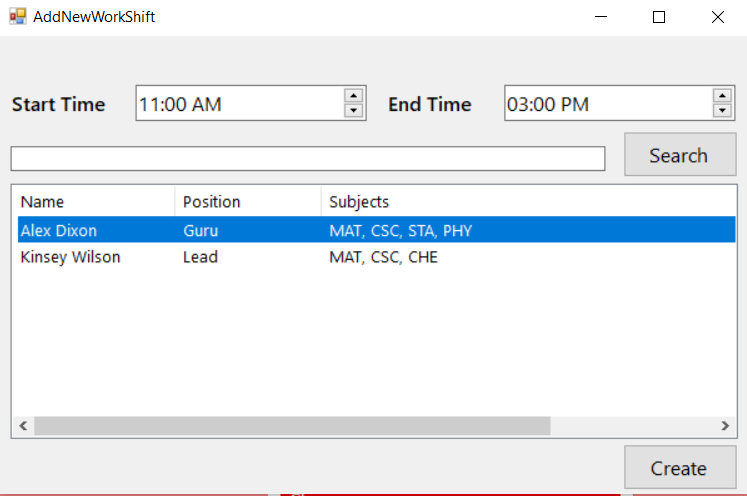


Figure 23 - New Work Shift GUI

R10.4. The system shall verify that the work event does not conflict with the chosen student worker’s class schedule.

R10.4.1. If the work event conflicts with the student worker’s class schedule, the system will display an error message to the user. The system will not create the work event and will allow the user to edit the work event.

R10.4.2. If there is no conflict between the work event and the student worker’s schedule, then the system will save the work event and display it on the schedule.

R11. The system shall allow the manager to create a document summarizing all the hours for each weekday that a student worker is available for a specific subject.

R11.1. The manager shall click the Subjects -> Manage Subjects button from the main menu.

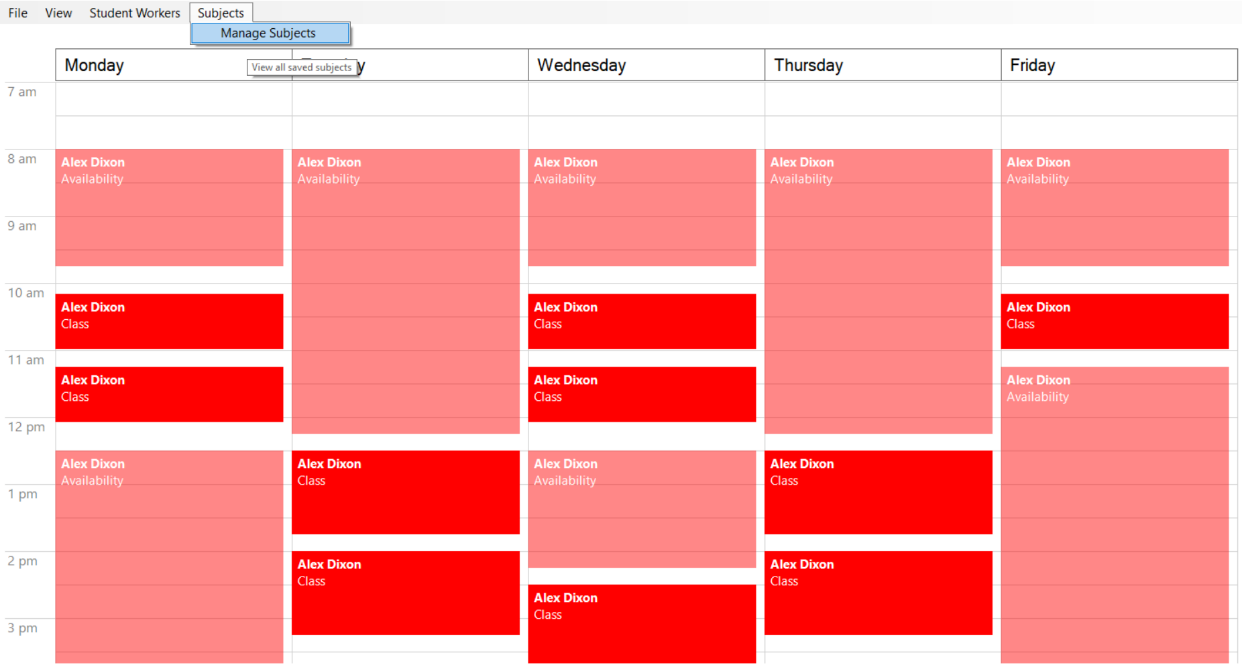


Figure 24 - Manage Subjects Button GUI

R11.2. The system shall display the current list of subjects.

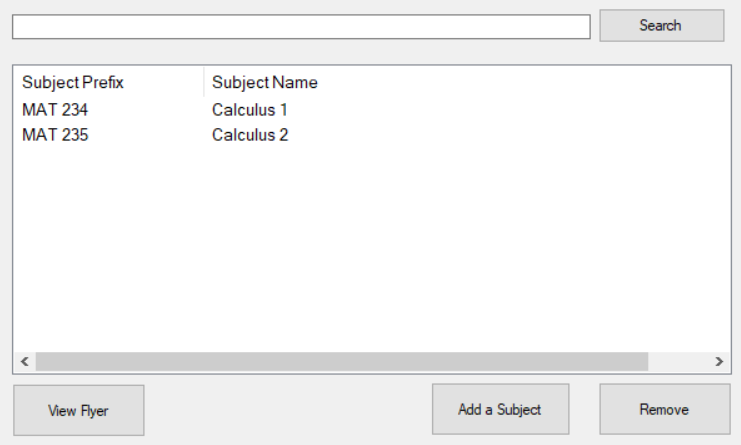


Figure 25 - Current Subject List GUI

R11.3. The manager shall select the subjects they wish to generate a subject flyer for.

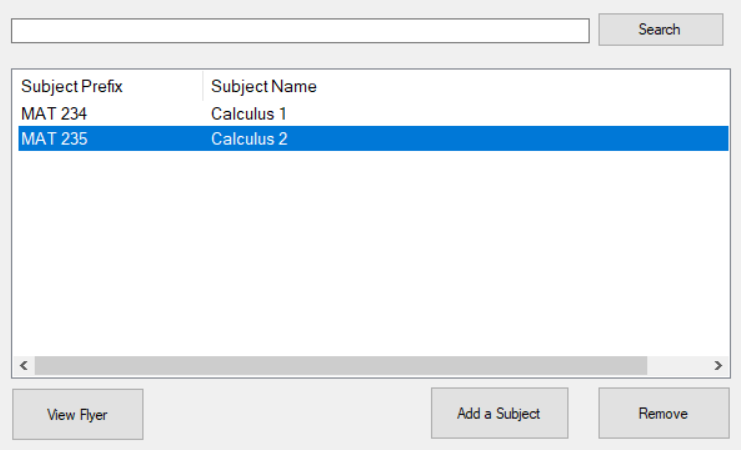


Figure 26 - Select Subject GUI

R11.4. The manager shall click the View Flyer button.

R11.5. The system shall compile and save the document summarizing the times each week that a tutor is available for the selected subject(s).

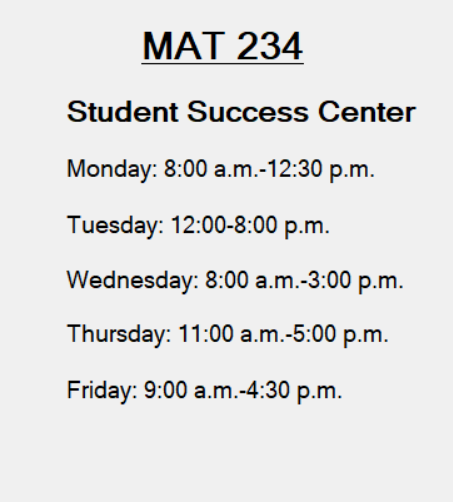


Figure 27 - Subject Flyer GUI

R12. The system shall allow the manager to verify that the schedule matches certain criteria and inform the manager of any erroneous situations that occur in the schedule.

R12.1. The manager shall edit the work schedule as previously described.  
R12.2. The system shall ensure that the events created do not cause conflicts with any of the specified criteria so the schedule is validated.

R13. The system shall allow a student worker to edit their weekly schedule of class meeting times.

R13.1. The student shall navigate to the web page using their preferred web browser and enter their student ID number.



Figure 28 - Student ID Input GUI

R13.2. The system shall display their current class schedule that has been entered.

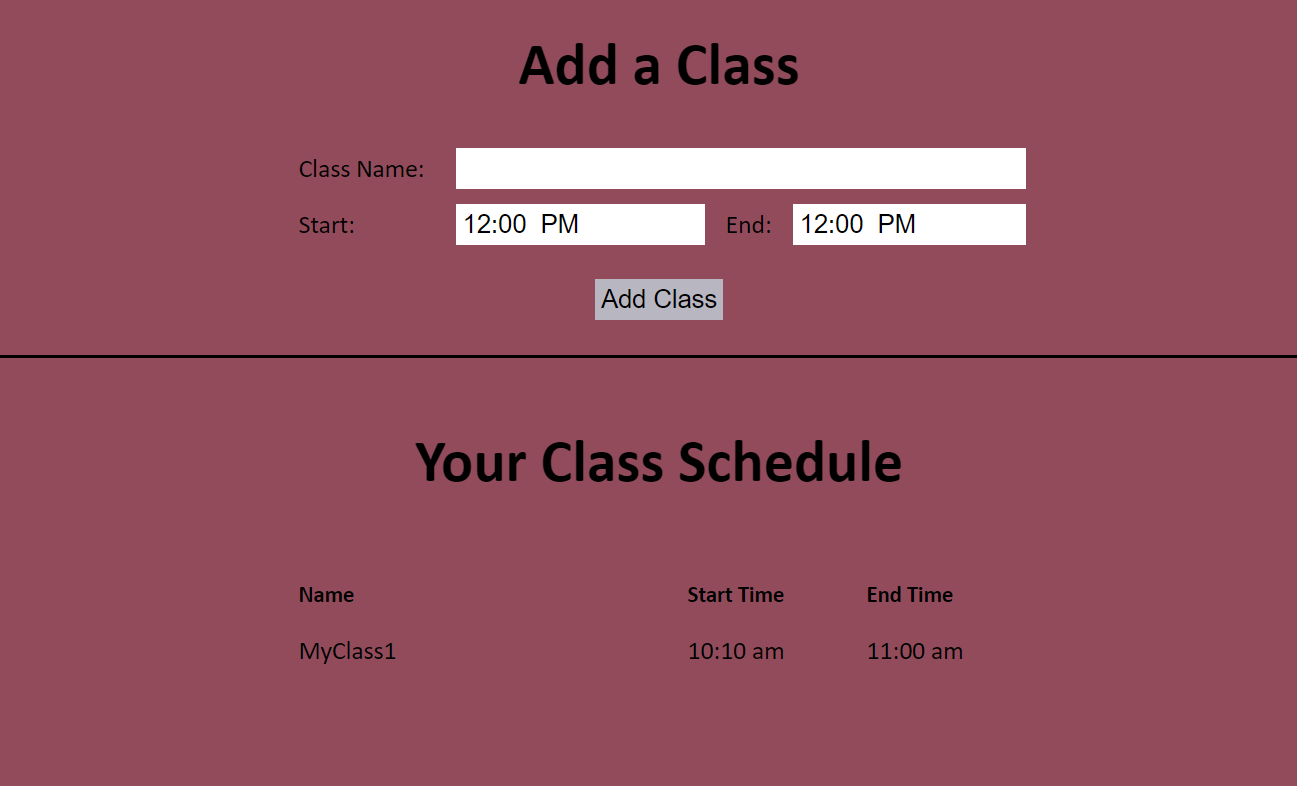


Figure 29 - Student Worker GUI

R13.3. The student worker shall enter in their new class information and click the Add Class button.

R13.4. The system shall verify the class information.

R13.5.1. If the class information was incomplete or conflicted with another class, the system will display an error message to the student. The system will not create the class in the schedule.

R13.5.2. If the class information was complete the system shall add the class to the student's schedule.

R14. The system shall allow a student worker to view their class and/or work schedule.

R14.1. The student worker shall navigate to their class schedule web page by the steps of R13.1 – R13.2.

R14.2. The system shall display the schedule with the student worker’s current class times.

## Non-Function Requirements

NR1. The system shall provide a user-friendly interface that conforms to Windows Forms standards for a consistent user experience.

NR2. The system must run on Windows 7 and later Windows operating systems.

NR3. The system must ensure the data it stores is secure from external sources or unauthorized users.

# Use Case Diagram



Figure 30 - Use Case Diagram

As seen in Figure 30, a student worker will be able to edit their own class schedule and view their schedule, including their classes and work shifts. The manager will be able log in to the system. Once logged in, the manager can view schedules of selected student workers, view student worker availability, edit a student worker’s work schedule, and check the current schedule to see if it meets all criteria. The manager can also add a new student worker to the system, remove student workers from the system, and edit a student worker’s information. Finally, the manager can add a subject to the system, remove a subject from the system, or generate a subject flyer that shows all times that a tutor is scheduled for that subject.

# Class Diagram



Figure 31 - Class Diagram

This diagram shows the classes that will interact in the system for efficient data storage and access. The Student Worker class represents student workers in the system and includes attributes for general student worker information; the class also contains operations related to a student worker including creating a new one, getting a student worker’s class and availability schedules, and editing a student worker’s information. The Subject class contains information important to a subject, including the subject’s name, number, and abbreviation. The Subject class contains operations to create a subject, remove a subject, and create a subject flyer. The Manager class contains the name of the manager and a function to verify login. The Schedule class is a component of the Student Worker class and is made up of CalendarEvents that contain information about a student worker’s class or work schedule. CalendarEvents also contain Times that consist of hours and minutes.

# Sequence Diagrams



:manager

:StudentWork

er

:DBMgr

saveStudentWorker()

result

saveStudentWorker()

result

name\_position

StudentWork

erInfoForm

«WinForm»

name\_position

createStudentWorker()

result

studentWorkerList

StudentWork

erInfoForm

«WinForm»

createStudentWorker()

result

studentWorkerList

Figure 32 - Add Student Worker Sequence Diagram

The manager will input, at minimum, the new student worker’s name and position through the StudentWorkerInfoForm. This form will call the createStudentWorker function of the StudentWorker class, which in turn will call saveStudentWorker of the DBMgr class to save the information to the database. The student worker list will then be updated to show the newly added student worker.



Figure 33 - Edit Own Class Schedule Sequence Diagram

The student worker will enter their student id to verify they are in the system and able to access their class schedule. If so, they will be redirected to their class schedule page, where they can then edit their class schedule. If not, they will be shown an error message to explain the issue.



Figure 34 - View Own Schedule Sequence Diagram

The student worker will enter their student id to verify they are in the system and able to access their class schedule. If so, they will be redirected to their class schedule page, where they can view their class schedule. If not, they will be shown an error message to explain the issue.



Figure 35 - Login Sequence Diagram

The manager will enter their username and password through the LoginForm. This will call the login function of the manager class, which in turn gets the manager object associated with the username from the DBMgr if the password is correct. The manager will then be shown the main screen. If the password is incorrect, they will be shown an incorrect password message.



Figure 36 - View Student Worker Schedules Sequence Diagram

The manager will select student workers from the main form whose schedules they wish to view. After the student worker objects, including schedules, are retrieved, the events for the schedules will be returned and displayed to the manager.



Figure 37 - View Availability Schedule Sequence Diagram

After the schedule has been displayed by the process of Figure 6, the manager may opt to view the student workers’ availability times through a component of the GUI. This will call the method buildAvailabilitySchedule of the StudentWorker class, which calculates available work times based on the student worker’s class schedule. The available times will then be displayed to the manager.



Figure 38 - Edit Work Schedule Sequence Diagram

The manager will enter the start time and end time for the new work time of the selected student worker. The mainForm GUI will call the editSchedule function of the selected student worker, which will call the saveStudentWorker function of the DBMgr to save all of the new information to persistent storage. The changes to the work schedule will then be displayed to the manager.



Figure 39 - Create Subject Flyer Sequence Diagram

The manager will select a subject from the list of subjects. The buildSubjectFlyer function of the selected subject object will generate the subject flyer and display it to the manager.



Figure 40 - Check Schedule Sequence Diagram

The manager will click a button to check that the schedule meets the necessary criteria. The checkSchedule function of the schedule object will check the criteria and return the results, which will be displayed to the manager for review.



Figure 41 - Edit Class Schedule Sequence Diagram

Similar to the process of editing the work schedule, the manager will enter the start and end time of the event for the selected student worker. In this case, the class name will also be entered. Then, the editSchedule function of the selected student worker object will update the schedule by calling the saveStudentWorker function of the DBMgr. The schedule will then be updated and shown to the manager.



Figure 42 - Edit Student Worker Info Sequence Diagram

The manager will navigate to the studentWorkerForm, which will load the list of student workers by retrieving them from the database and then displaying selected information. The manager will then select a student worker to edit and input the new information. The editInfo function of the selected student worker object will then call saveStudentWorker in the DBMgr to save the updated information and return the result to inform the manager of the successful changes.



Figure 43 - Remove Student Worker Sequence Diagram

The manager will select a student worker to remove, and the removeStudentWorker function of the StudentWorker class will call the removeStudentWorker function of the DBMgr to update the database accordingly. The changes will then be reflected in the student worker list to show the manager that the student worker was removed.



Figure 44 - Add Subject Sequence Diagram

The manager will enter the name, prefix, and number of the new subject to add using the subjectsForm. The create method of the new subject object will call the saveSubject function of the DBMgr to save the new information to the database and return the result. The manager will then be shown the updated subject list including the newly added subject.



Figure 45 - Remove Subject Sequence Diagram

The manager will select a subject to remove through the subjects form. The removeSubject function of the subject class will call the removeSubject function of the DBMgr to update the database accordingly. The subject list will then be updated to reflect the changes to the manager.

# Activity Diagrams



Figure 46 - Add Student Worker Activity Diagram

The manager will enter the name, id, and position of the new student worker through the GUI, which will pass that information to the createStudentWorker function. This function will create a new StudentWorker instance and pass the information to the DBMgr to save it to the database. If the database save fails, the error message will be returned and then displayed to the manager. If the save is successful, the success message will be returned and the full student worker info list will be displayed including the newly-added information.



Figure 47 - Add Subject Activity Diagram

The manager will enter the name, prefix, and number of the new subject through the GUI, which will pass that information to the createSubject function. This function will create a new Subject instance and pass the information to the DBMgr to save it to the database. If the database save fails, the error message will be returned and then displayed to the manager. If the save is successful, the success message will be returned and the full subject info list will be displayed including the newly-added information.



Figure 48 - Check Schedule Activity Diagram

The manager will click the button to check the schedule. The mainForm GUI will call the check schedule function that will check all the criteria of the schedule for every event on the schedule. First, it will loop through all work events to check the criteria only for work events. Then, it will loop through all events to check criteria pertaining to all events. Finally, it will display the list of conflicts in the schedule.



Figure 49 - Create Subject Flyer Activity Diagram

The manager will select a subject from the list of subjects and click the Create Subject Flyer button. The subjectsForm will call the buildSubjectFlyer function which will create the subject flyer. This function will loop through each student worker and check whether they cover the selected subject. If so, their schedule will be added to the subject flyer. Finally, the subject flyer will be displayed.



Figure 50 - Edit Class Schedule Activity Diagram

The manager will enter the student worker, class name, and start/end times for the new class event. The mainForm will call the editSchedule function to modify the schedule. This function will create a new event object and save it to the database through the DBMgr. On success, the updated schedule will be displayed; otherwise, an error message will be displayed.



Figure 51 - Edit Own Class Schedule Activity Diagram

The student worker user will enter their student id to verify their identity. The system will check their id in the database. If found, the class schedule page will be shown; otherwise, an error message will be shown. Once the class schedule page is shown, the student worker will make and submit a schedule change, and the system will update their information in the database and inform the user of the success result.



Figure 52 - Edit Student Worker Info Activity Diagram

The manager will click the button to view all student workers, and the system will retrieve all the student workers from the database and display the results in a list. The manager will select a student worker and edit their information using the student worker GUI. The system will then update these changes in the database and display the success result.



Figure 53 - Edit Work Schedule Activity Diagram

The manager will enter the student worker and start/end times for the new work event. The mainForm will call the editSchedule function to modify the schedule. This function will create a new event object and save it to the database through the DBMgr. On success, the updated schedule will be displayed; otherwise, an error message will be displayed.



Figure 54 Login Activity Diagram

The manager will enter their username and password and click the login button. The LoginForm will call the login() method from the manager class. The login() method will call the getManager() method from the DBMgr class. The function will establish a connection to the database and execute a query to determine if the user entered the correct password. If the password is incorrect, the LoginForm will display an incorrect login error to the user. If the password is correct, the system will display the mainForm.



Figure 55 Remove Student Worker Activity Diagram

The manager will select a student worker and click the remove button. The studentWorkerForm will call the removeStudentWorker() function from the StudentWorker class. The StudentWorker class will pass the studentID of the selectedStudentWorker to the removeStudentWorker() function from the DBMgr class. The function will execute a query using the studentID to delete the student worker from the database. If the deletion is not successful, the system will display an error. If the deletion is successful, the studentWorkerForm will display an updated list where the selectedStudentWorker has been removed.



Figure 56 Remove Subject Activity Diagram

The manager will select a subject and click the remove button. The subjectForm will call the removeSubject() function from the Subject class. The Subject class will pass the subjectAbbreviation and the subjectNumber of the selectedSubject to the removeSubject() function from the DBMgr class. The function will execute a query using the subjectAbbreviation and subjectNumber to delete the subject from the database. If the deletion is not successful, the system will display an error. If the deletion is successful, the subjectForm will display an updated list where the selectedSubject has been removed.



Figure 57 View Availability Schedule Activity Diagram

The manager will choose the student worker whose availability they wish to view. The mainForm will call the buildAvailabilitySchedule() function from the studentWorker class. If the student worker does not have a class schedule in the system, then the method will return. If the student worker has a class schedule, the method will create a list of dailyEvents and add the student worker’s classes into the list. The method will then loop through the dailyEvents list checking for availability before, after, and between classes. The method will add any available times into the student worker’s availability schedule. The mainForm will then display the student worker’s availability.



Figure 58 View Student Worker's Schedules Activity Diagram

The manager will select the student workers whose schedules they wish to view. The mainForm will call the getStudentWorkers() method from the StudentWorker class. The StudentWorker will call the method from the DBMgr class to fetch the student worker information from the database. The schedule for each student worker and the events in the schedule will be retrieved and displayed to the manager.



Figure 59 View Own Schedule Activity Diagram

The student worker user will enter their student id to verify their identity. The system will check their id in the database. If found, the class schedule page will be shown; otherwise, an error message will be shown.

# State Diagrams



Figure 60 TutorScheduler State Diagram

The user begins by logging into the system. If the login is successful, the system will display the main page where the user can view the schedule. If the login is unsuccessful, the user will remain on the login page. Once on the main page, the user can view the day schedule by clicking day view. They can return to week view by clicking week view. The user can view the subject list by clicking manage subjects. From there, the user can add a subject, remove a subject, or view a subject flyer. From the main page, the user can also choose to display a chosen student worker’s schedule. They can also view all student workers by choosing manage student workers. When viewing all student workers, the user can add a student worker, view and edit a student worker, or delete a student worker.

# Database Design

## ER Schema

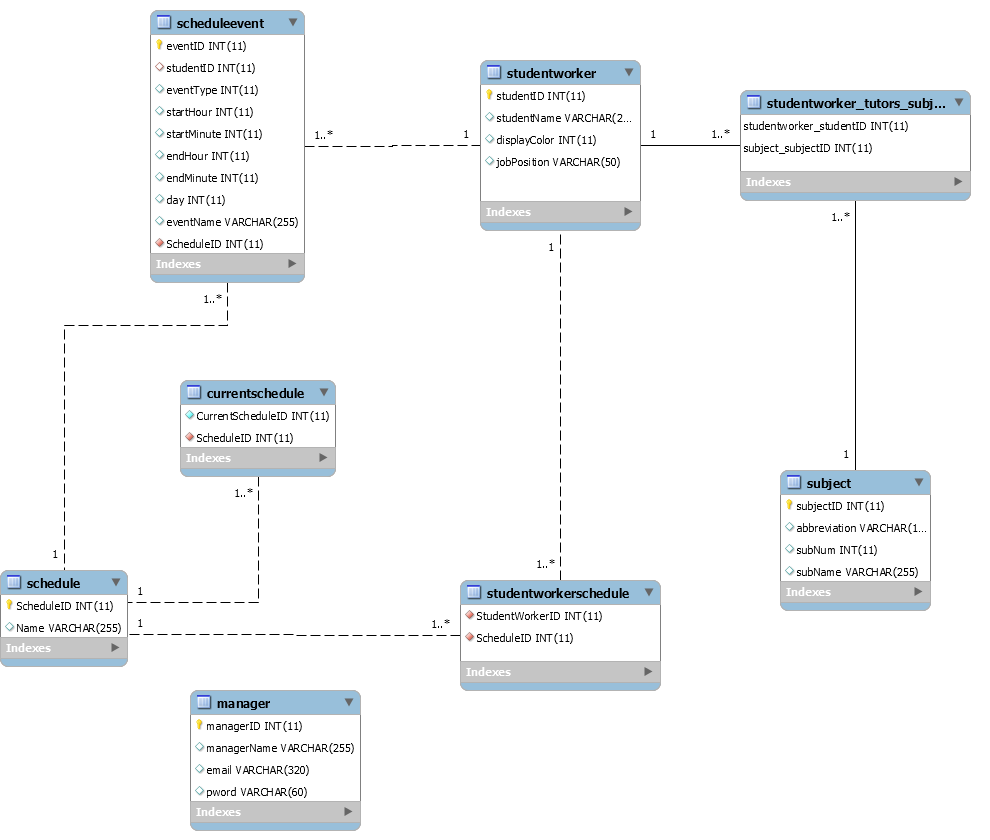


Figure 61 - Tutor Scheduler ER Diagram

The database will have eight entities: Manager, Student Worker, Schedule Events, Subject, Schedule, Current Schedule, StudentWorkerSchedule, and SubjectTutored. A manager can employ many student workers, while a student worker can only be employed by one manager. A student worker can tutor many subjects, and each subject can be tutored by many student workers. A student worker can have many schedule events, but a schedule event can only belong to one student worker.

## Table Schema

StudentWorker(studentID, studentName, displayColor,jobPosition)

Manager(managerID, email, password, managerName)

Schedule(ScheduleID, Name)

CurrentSchedule(CurrentScheduleID)

StudentWorkerSchedule(StudentWorkerID, ScheduleID)

ScheduleEvent(EventID, studentID, eventType, startHour, startMinute, endHour, endMinute, day, eventName, scheduleID)

Subject(subjectID, abbreviation, subNum, subName)

SubjectTutored(studentID, subjectID)

# Conclusion

Our object-oriented system reduces the workload of the scheduling process for tutoring managers. The intuitive UI and organized presentation of information will make this possible. It allows tutors and managers to communicate their responsibilities for class schedules and work schedules efficiently. The object-oriented design of the system will improve the organization and modularity of the system, making it easier to expand and improve if further evolution of the system is desired.

# Data Dictionary

Schedule – The student worker schedule for the Student Success Center is the same from week to week for an entire academic semester. The system will focus on creating this schedule that is used throughout the semester but can also be revised and updated if needed.

Subject – A course subject that a student worker tutors. Includes the name, course prefix, and course number.

Subject flyer – A document that shows the comprehensive hours that at least one tutor for a specific class or subject is scheduled to work. These are distributed to students to inform them of what times they can get help with specific subjects.

Work event - A work event contains the information about a work shift including the student worker to whom the shift belongs, the shift’s start time, and the shift’s end time.

Student Worker – A student tutor that has a class schedule from which the availability schedule is determined and is assigned a work schedule by the manager.

Manager – The main user of the system who creates the work schedule for all student workers.

Calendar Event – An event that is saved in the schedule; has a specific type of availability, class, or work, as well as a start time and an end time.

Time – A simple time structure containing only the hours in 24-hour format and the minutes, both stored as integers.

Position – A student worker’s job position.