Desktop Planner

Software Requirements Specification

Version 1.0

8 October 2018

Tyler Handerhan, Zhisheng Xu, Michael Long,
Chengzhi Sun, Evan McWilliams

Revision History

Date	Description	Author	Comments
<date>	<Version 1>	<Your Name>	<First Revision>

Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

Signature	Printed Name	Title	Date
	<Your Name>	Software Eng.	

Table of Contents

1. Introduction

1.1 Purpose

Because people have many tasks in their daily lives, tools to manage tasks and commitments they have are invaluable. This application will assist in keeping track of different types of commitments that many students have such as classes, exams, assignments, and appointments.

1.2 Scope

The Desktop Planner application will maintain several user specified lists that it will use to display a calendar holding exams, assignments, appointments and classes to the user. Another list will also hold any alarms associated with the calendar events. Alarms will go off at the time that they are specified to go off at and will display what event they are associated. A location can also be associated with an event and when an event is opened the location can open a browser window to google maps showing the location.

1.3 Definitions, Acronyms, and Abbreviations

GUI: Graphical User Interface
STD: State-Transition Diagram

2. General Description

2.1 Product Perspective

This application will have two parts: the desktop executable and its web extension. The Desktop executable will be the entry point of the planner application while the web portal will be used to display the scheduled event locations and other information about them.

A GUI library will be needed for creating human-friendly UI in the desktop executable for users. The UIs will trigger all the user operations and all information or results from the application to the users. In addition, it also needs a database locally in users' machines to serve as a space for both storing user's information and communicating data between the desktop executable and web extension. All the communications will be done locally in users' operation systems.

The web extension will be used to search related information about users' appointments if requested. In order to implement the portability among common operating systems, Google Chrome will be used for all Internet functions of the planner application. The desktop executable will communicate the request from the database mentioned above to the Google Chrome extension in order to get access any online content especially Google Maps.

Finally, a security library will be used to cipher the information with the database. It will not be a perfect secured fire-wall system but is a standard library that can prevent simple to medium level hackings. This security library will be cross-platform as well.

2.2 Product Functions

The planner application will be able to present users with lists of appointments and courses. Users should be able to add or remove any courses and appointments freely. For each event, the application will also check any alarms that associated with the event. In addition, the application should also be able to find related information about users' courses and their location information. Finally, it can check the time conflict among different appointments, exams, and classes.

The web extension will fetch the correct location information through Google Chrome on the Internet and transfer it to desktop executable through the database. Also, it will automatically disabled if the users do not have access to the Internet.

2.3 Constraints, Assumptions and Dependencies

The first assumption is that the majority of the users will be using the application in an academic setting, namely students, professors or school faculty members. It will not allow time conflicting events as some other planners do because appointments, exams, and classes should always be exclusive during certain their allotted time periods.

Another assumption is that the security environment of users' operating systems will be at a strong level. Since the security library for the database is not going to be a Great Wall style anti-hacking system, the planner application will always assume the operating system it runs on will have strong security firewall already installed.

One constraint of the application will be the accessibility to the Internet. Although most functionalities will still perform normally without an Internet connection, some important features such as searching for appointment locations.

Another constraint is that the application needs the operating system where it runs on to support the user's default browser. It will be troublesome if the application uses operating systems' default API or system libraries to get access to the machines' browsers because different operating systems will have totally API or system libraries and these entry points may change frequently with operating systems' updates. Figuring out each operating systems' API and updating planner application for each operating system update is just not feasible.

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 Software Interfaces

The JavaFX library will be used to connect with the front-end interface of the application in FXML, an XML-based markup language. Very little connecting of the formats needs to be done, because the JavaFX library automatically communicates with FXML files, and comes as part of the normal Java development kit.

3.2 Functional Requirements

3.2.1 List of Courses

Add and remove courses from a list that the planner stores.

3.2.2 List of Scheduled Items

Add and remove assignments, appointments, and exams from a list that the planner stores.

3.2.3 List of Alarms

Add, remove, turn on, and turn off alarms in a list on the planner.

3.2.4 Display of Courses

Similar UI to display the current list of courses and their basic details such as course names.

3.2.5 Display of Scheduled Items

Similar UI for showing the current list of assignments, appointments, and exams.

3.2.6 Display of Alarms

Similar UI for showing the current list of alarms, and shows whether the alarms are active.

3.2.7 Edit Course Details

Open course information, change, and save at any time.

3.2.8 Edit Scheduled Item Details

Open assignment, appointment, or exam info and change it at any time.

3.2.9 Edit Alarm Details

Open alarms and change them at any time.

3.2.10 Filter Scheduled Items by type

Menu option used to choose what types of items should be display in the items list or all of them at once.

3.2.11 Alarm Notifications

At the alarm's time a OS notification appears.

3.2.12 Open Scheduled Item Locations in Browser

Locations may be clicked to open the location in google maps.

3.3 Use Cases

3.3.1 View Calendar

Use Case ID:	1.0
Use Case Name:	View Calendar

Created By:	Michael Long	Last Updated By:	Michael Long
Date Created:	10/6/2018	Date Last Updated:	10/7/2018

Actor:	User
Description:	Immediately after start up and after the user has completed any operation that would change the appearance of the calendar, the GUI will update the visual representation of the new calendar.
Preconditions:	1. User must be logged in
Postconditions	1. GUI will display the calendar
Priority:	High
Frequency of Use:	Will be performed immediately after start up and whenever the user completes an operation that would change the calendar.
Normal Course of Events:	1. The calendar will add all the events it has and any courses the user has to the graphical representation of the calendar, updating any changes that occured
Alternative Courses:	None
Exceptions:	1.0.EX.1 Error occurs while accessing calendar information in the database. Program should display an error message to the user. 1.0.EX.2 Error occurs while rendering the graphical representation of the calendar. Program should display an error message.
Includes:	None
Special Requirements:	None
Assumptions:	Needs access to the GUI and all information associated with the calendar.
Notes and Issues:	None

3.3.2 Add Class

Use Case ID:	2.0
Use Case Name:	Add Class

Created By:	Michael Long	Last Updated By:	Michael Long
Date Created:	10/6/2018	Date Last Updated	10/7/2018

Actor:	User
Description:	The user will press an “add class” button in the GUI. The GUI will then collect information necessary to make a new class such as time location days of the week and subject. The GUI will then pass this information to the user information handler which will create the class in the database and update the graphical representation of the calendar.
Preconditions:	1. User must be logged in
Postconditions	1. Class will be entered into database 2. Class will be represented on the calendar
Priority:	Medium
Frequency of Use:	Will only be run when the user adds a class.
Normal Course of Events:	1. User will press the “add class” button 2. User will enter all information needed to create class 3. GUI will pass this information to the user information handler 4. User information handler will update the database with the new class 5. Calendar will update visual representation
Alternative Courses:	None
Exceptions:	2.0.EX.1 User enters incorrect information type into GUI. A message will be displayed to the user explaining the error and how to enter the correct information. 2.0.EX.2 Error occurs while creating the new class. An error message should be displayed to the user.
Includes:	1.0 View calendar
Special Requirements:	None
Assumptions:	Must have access to the user information handler and the calendar GUI.
Notes and Issues:	None

3.3.3 Remove Class

Use Case ID:	3.0
Use Case Name:	Remove Class

Created By:	Michael Long	Last Updated By:	Michael Long
Date Created:	10/6/2018	Date Last Updated:	10/7/2018

Actor:	User
Description:	The user will press a “remove class” button in the GUI. The GUI will then call the calendar handler which will remove the specified class from the database and updates the graphic representation of the calendar.
Preconditions:	1. User must be logged in 2. User must have at least one class associated with them
Postconditions	1. The class will be removed from the database 2. Calendar will update with the changes to the database
Priority:	Low
Frequency of Use:	Can only be used as many times as classes are created.
Normal Course of Events:	1. User will specify a class to be removed 2. Class identifier will be passed to the user information handler 3. User information handler will then remove the specified class from the database 4. Calendar will update visual representation
Alternative Courses:	None
Exceptions:	3.0.EX.1 Class was not found in the database. The user should be given a notification that the class was not found. 3.0.EX.2 Error occurs while removing the class from the database. The user should be notified with an error message.
Includes:	1.0 View Calendar
Special Requirements:	None
Assumptions:	Needs access to the user information handler and the calendar GUI.
Notes and Issues:	None

3.3.4 Edit Class

Use Case ID:	4.0
Use Case Name:	Edit Class

Created By:	Michael Long	Last Updated By:	Michael Long
Date Created:	10/6/2018	Date Last Updated:	10/7/2018

Actor:	User
Description:	The user will press an “edit class” button in the GUI. The GUI will then collect any information that the user wants to update. It will then call the user information handler to update the class information in the database and update the graphical representation of the calendar.
Preconditions:	1. User must be logged in 2. User must have at least one class associated with them
Postconditions:	1. Class must be updated with the new information from the user 2. Calendar will update with the changes to the database
Priority:	Low
Frequency of Use:	May not be used at all but can only be called when at least one class is associated with the user.
Normal Course of Events:	1. User specifies the class that they want to update 2. User will enter any information that they want to change 3. New information and class identifier are passed to the user information handler 4. User information handler will update the specified information 5. Calendar will update visual representation
Alternative Courses:	None
Exceptions:	4.0.EX.1 Specified class not found in the database. User is notified via an error message.

	4.0.EX.2 No classes in database. User is notified via an error message. 4.0.EX.3 Error occurs updating the information in the database. User is notified via an error message.
Includes	1.0 View Calendar
Special Requirements:	None
Assumptions:	Must have access to the user information handler and the calendar GUI.
Notes and Issues:	None

3.3.5 Add Appointment

Use Case ID:	5.0
Use Case Name:	Add Appointment

Created By:	Michael Long	Last Updated By:	Michael Long
Date Created:	10/6/2018	Date Last Updated:	10/7/2018

Actor:	User
Description:	The user will press an “add appointment” button in the GUI. The GUI will then collect information necessary to make a new appointment such as time, location and a description of the appointment. This information will then be passed to the calendar handler which will use it to create a new appointment in the database and update the graphical representation of the calendar
Preconditions:	1. User must be logged in
Postconditions	1. Appointment will be entered into database 2. Appointment will be represented on the calendar
Priority:	Medium
Frequency of Use:	Will only be run when the user adds an appointment
Normal Course of Events:	1. User will press the “add appointment” button

	2. User will enter all information needed to create the appointment 3. GUI will pass this information to the calendar handler 4. Calendar handler will update the database with the new appointment 5. Calendar will update visual representation
Alternative Courses:	None
Exceptions:	5.0.EX.1 User enters incorrect information type into GUI. A message will be displayed to the user explaining the error and how to enter the correct information. 5.0.EX.2 Error occurs while creating the new appointment. An error message should be displayed to the user.
Includes:	1.0 View Calendar
Special Requirements:	None
Assumptions:	Must have access to the calendar handler and the calendar GUI.
Notes and Issues:	None

3.3.6 Remove Appointment

Use Case ID:	6.0
Use Case Name:	Remove Appointment

Created By:	Michael Long	LastUpdated By:	Michael Long
Date Created:	10/6/2018	Date Last Updated:	10/7/2018

Actor:	User
Description:	The user will press a “remove appointment” button in the GUI. The GUI will then call the calendar handler which will remove the specified appointment from the database and update the graphical representation of the calendar.
Preconditions:	1. User must be logged in 2. User must have at least one appointment associated with

	their calendar
Postconditions	1. The appointment will be removed from the database 2. Calendar will update with the changes to the database
Priority:	Low
Frequency of Use:	Can only be used as many times as appointments are created.
Normal Course of Events:	1. User will specify a appointment to be removed 2. Appointment identifier will be passed to the calendar handler 3. Calendar handler will then remove the specified appointment from the database 4. Calendar will update visual representation
Alternative Courses:	None
Exceptions:	6.0.EX.1 Appointment was not found in the database. The user should be given a notification that the appointment was not found. 6.0.EX.2 Error occurs while removing the appointment from the database. The user should be notified with an error message.
Includes	1.0 View Calendar
Special Requirements:	None
Assumptions:	Needs access to the calendar handler and the calendar GUI.
Notes and Issues:	None

3.3.7 Edit Appointment

Use Case ID:	7.0
Use Case Name:	Edit Appointment

Created By:	Michael Long	Last Updated By:	Michael Long
Date Created:	10/6/2018	Date Last Updated:	10/7/2018

Actor:	User

Description:	The user will press an “edit appointment” button in the GUI. The GUI will then collect an information that the user wants to update and call the calendar handler to update the appointment information in the database and update the graphical representation of the calendar.
Preconditions:	1. User must be logged in 2. User must have at least one appointment associated with them
Postconditions	1. Appointment must be updated with the new information from the user 2. Calendar will update with the changes to the database
Priority:	Low
Frequency of Use:	May not be used at all but can only be called when at least one appointment is associated with the user’s calendar.
Normal Course of Events:	1. User specifies the appointment that they want to update 2. User will enter any information that they want to change 3. New information and appointment identifier are passed to the calendar handler 4. Calendar handler will update the specified information 5. Calendar will update visual representation
Alternative Courses:	None
Exceptions:	7.0.EX.1 Specified appointment not found in the database. User is notified via an error message. 7.0.EX.2 No appointments in database. User is notified via an error message. 7.0.EX.3 Error occurs updating the information in the database. User is notified via an error message.
Includes	1.0 View Calendar
Special Requirements:	None
Assumptions:	Must have access to the calendar handler and the calendar GUI.
Notes and Issues:	None

3.3.8 Add Exam

Use Case ID:	8.0
Use Case Name:	Add Exam

Created By:	Michael Long	Last Updated By:	Michael Long
Date Created:	10/6/2018	Date Last Updated:	10/7/2018

Actor:	User
Description:	The user will press an “add exam” button in the GUI. The GUI will then collect collect information necessary to make a new event such as time, location and a description of the exam. This information will then be passed to the calendar handler which will use it to create a new exam in the database and update the graphical representation of the calendar.
Preconditions:	1. User must be logged in
Postconditions	1. Appointment will be entered into database 2. Appointment will be represented on the calendar
Priority:	Medium
Frequency of Use:	Will only be run when the user adds an exam.
Normal Course of Events:	1. User will press the “add appointment” button 2. User will enter all information needed to create the appointment 3. GUI will pass this information to the calendar handler 4. Calendar handler will update the database with the new appointment 5. Calendar will update visual representation
Alternative Courses:	None
Exceptions:	8.0.EX.1 User enters incorrect information type into GUI. A message will be displayed to the user explaining the error and how to enter the correct information. 8.0.EX.2 Error occurs while creating the new exam. An error message should be displayed to the user.
Includes	1.0 View Calendar

Special Requirements:	None
Assumptions:	Must have access to the calendar handler and the calendar GUI.
Notes and Issues:	None

3.3.9 Remove Exam

Use Case ID:	9.0
Use Case Name:	Remove Exam

Created By:	Michael Long	LastUpdated By:	Michael Long
Date Created:	10/6/2018	Date Last Updated	10/7/2018

Actor:	User
Description:	The user will press a “remove exam” button in the GUI. The GUI will then call the calendar handler which will remove the specified exam from the database and update the graphical representation of the calendar.
Preconditions:	1. User must be logged in 2. User must have at least one exam associated with their calendar
Postconditions	1. The exam will be removed from the database 2. Calendar will update with the changes to the database
Priority:	Low
Frequency of Use:	Can only be used as many times as appointments are created.
Normal Course of Events:	1. User will specify a exam to be removed 2. Exam identifier will be passed to the calendar handler 3. Calendar handler will then remove the specified exam from the database 4. Calendar will update visual representation
Alternative Courses:	None
Exceptions:	9.0.EX.1 Exam was not found in the database. The user should be

	given a notification that the exam was not found 9.0.EX.2 Error occurs while removing the exam from the database. The user should be notified with an error message.
Includes:	1.0 View Calendar
Special Requirements:	None
Assumptions:	Needs access to the calendar handler and the calendar GUI.
Notes and Issues:	None

3.3.10 Edit Exam

Use Case ID:	10.0
Use Case Name:	Edit Exam

Created By:	Michael Long	Last Updated By:	Michael Long
Date Created:	10/6/2018	Date Last Updated	10/7/2018

Actor:	User
Description:	The user will press an “edit exam” button in the GUI. The GUI will then collect an information that the user wants to update and call the calendar handler to update the exam information in the database and update the graphical representation of the calendar.
Preconditions:	1. User must be logged in 2. User must have at least one exam associated with them
Postconditions	1. Exam must be updated with the new information from the user 2. Calendar will update with the changes to the database
Priority:	Low
Frequency of Use:	May not be used at all but can only be called when at least one exam is associated with the user’s calendar.
Normal Course of Events:	1. User specifies the exam that they want to update 2. User will enter any information that they want to change

	3. New information and exam identifier are passed to the calendar handler 4. Calendar handler will update the specified information 5. Calendar will update visual representation
Alternative Courses:	None
Exceptions:	10.0.EX.1 Specified exam not found in the database. User is notified via an error message. 10.0.EX.2 No exams in database. User is notified via an error message. 10.0.EX.3 Error occurs updating the information in the database. User is notified via an error message.
Includes:	1.0 View Calendar
Special Requirements:	None
Assumptions:	Must have access to the calendar handler and the calendar GUI.
Notes and Issues:	None

3.3.11 Add Assignment

3.3.12 Remove Assignment

3.3.13 Edit Assignment

3.3.14 Add Alarm to Exam

3.3.13 Remove Alarm from Exam

3.3.14 Edit Exam Alarm

3.3.15 Add Alarm to Appointment

3.3.16 Remove Alarm from Appointment

3.3.17 Edit Appointment Alarm

3.3.18 First Startup

3.3.19 Startup

3.3.20 Edit Password

3.3.21 Open Location in Google Maps

3.3.22 Filter Events

3.3.23 Display Alarm List

3.3.24 Disable Alarm

3.3.25 Display Scheduled Item List

3.3.26 Display Courses List

3.3.27 Remove User

3.3.28 Exit Program

3.3.29 Modify Profile

3.4 Classes / Objects

3.4.1 UserProfile

Concrete
Superclass: Object
Subclasses: none

3.4.1.1 Attributes:

String Username
String Password
Calendar calendar

3.4.1.2 Methods:

viewCalendar()
setPassword()
setUsername()
validate()

3.4.2 CalendarGUI

Concrete
Superclass: Object
Subclass: none

3.4.2.1 Attributes:

Calendar calendar
Button modifyEvent
Button removeEvent
Button modifyProfile
Button Exit

3.4.2.2 Methods:

clickCreateEvent()
clickModifyEvent()
clickRemoveEvent()
clickModifyProfile() (display GUI to change username or password)

clickExitSave()

3.4.3 Calendar

Concrete
Superclass: Object
Subclass: none

3.4.3.1 Attributes:

ArrayList<Event> events

3.4.3.2 Methods:

createEvent()
modifyEvent()
removeEvent()
displayProfileModify()
exitSave()

3.4.4 ProfileModifyGUI

Superclass: Object
Subclass: none

3.4.4.1 Attributes:

UserProfile user
TextPrompt username
TextPrompt password
TextPrompt newUserName
TextPrompt newPassword
Button submitChanges

3.4.4.2 Methods:

clickSubmitChanges()
validateUser()

3.4.5 Event

Superclass: Object
Subclass: Exam, Assignment, Appointment

3.4.5.1 Attributes:

boolean alarmOn
String name
String date
String time
Alarm alarm

3.4.5.2 Methods:

getName()
setName()
getDate()
setDate()
getTime()
setTime()
setAlarm()
removeAlarm()
hasAlarm()

3.4.6 Class

Concrete
Superclass: Object
Subclass: none

3.4.6.1 Attributes:

boolean alarmOn
String name
String date
String time
String location
Alarm alarm
String[] daysOfWeek
String lastDay

3.4.6.2 Methods:

getName()
setName()
getDate()
setDate()
makeReocurring()
getTime()
setTime()
getLocation()
setLocation()
setAlarm()
removeAlarm()
hasAlarm()
setDaysOfWeek()
getDaysOfWeek()

3.4.7 Exam

Concrete
Superclass: Event
Subclass: none

3.4.7.1 Attributes:

boolean alarmOn
String name
String date
String time
String location
Alarm alarm
String class
String topic

3.4.7.2 Methods:

getName()
setName()
getDate()
setDate()
getTime()
getLocation()
setLocation()
setTime()
setAlarm()
removeAlarm()
hasAlarm()
setClass()
getClass()
setTopic()
getTopic()

3.4.8 Assignment

Concrete
Superclass: Event
Subclass: none

3.4.8.1 Attributes:

boolean alarmOn
String name
String date
String time
Alarm alarm
String assignedDate

String class
String description

3.4.8.2 Methods:

getName()
setName()
getDate()
setDate()
getTime()
setTime()
setAlarm()
removeAlarm()
hasAlarm()
setAssignedDate()
getAssignedDate()
setClass()
getClass()
setDescription()
getDescription()

3.4.9 Appointment

Concrete
Superclass: Event
Subclass: none

3.4.9.1 Attributes:

boolean alarmOn
String name
String date
String time
Alarm alarm
String location
String withWho

3.4.9.2 Methods:

getName()
setName()
getDate()
setDate()
getTime()
setTime()
setAlarm()
removeAlarm()
hasAlarm()
setLocation()

setWhoWith()
getLocation()
getWhoWith()

3.4.10 Alarm

Concrete
Superclass: Object
Subclass: none

3.4.10.1 Attributes:

TimeStamp alarmTime
boolean on

3.4.10.2 Methods:

setAlarm()
disable()
makeNoise()

3.4.11 LoginGUI

Concrete
Superclass: Object
Subclass: none

3.4.11.1 Attributes:

LoginPage backEndLogIn
TextPrompt userNameAttempt
TextPrompt passwordAttempt

3.4.11.2 Methods:

submit()

3.4.12 LogInPage

Concrete
Superclass: Object
Subclass: none

3.4.12.1 Attributes:

String username
String password
boolean hasAccount

3.4.12.2 Methods:

enterUserName()
enterPassword()
hasAccount()
createAccount()
validate()
getUserProfile()

3.4.13 SimpleUsersDB

Concrete
Superclass: Object
Subclass: none

3.4.13.1 Attributes:

ArrayList<UserProfile> userInfo
File savedUserInfo

3.4.13.2 Methods:

addUser()
removeUser()
checkUser()
readInfoFile()
saveUserInfo()

3.5 Non-Functional Requirements

3.5.1 Interface Is Easy to Use and Intuitive

The UI should require as few steps as possible for the user to perform in order to perform any specific operation.

3.5.2 Scheduled Items are Displayed in Order, from Soonest Time to Latest

The dates associated with each item will determine how easily the user will be able to see them.

3.5.3 Proper Implementation of Data Structures

There should be no errors in data management

3.5.4 Operations Complete in Reasonably Short Time

In order to keep the application usable, items should be able to be entered as soon and as quickly as possible

3.5.5 Alarms Go Off at the Correct Times

Alarms are predictable and useful

3.5.6 Item Edits Will Be Saved and Updated on the Display

When the user is finished editing any item, its new data is shown on the display

3.5.7 Appointments and Exams Cannot be Scheduled at Conflicting Times

It is assumed by the program that appointments and exams cannot be attended simultaneously, so the user may not schedule them at the same time

3.5.8 Assignments Can be Scheduled at Conflicting Times

If an assignment may be turned in early or submitted online, it should be allowed to be due at the same time as another assignment

3.5.9 Item Locations Open in User's Default Browser

The program will attempt to use the user's default browser on their computer to open location links

3.6 Logical Database Requirements

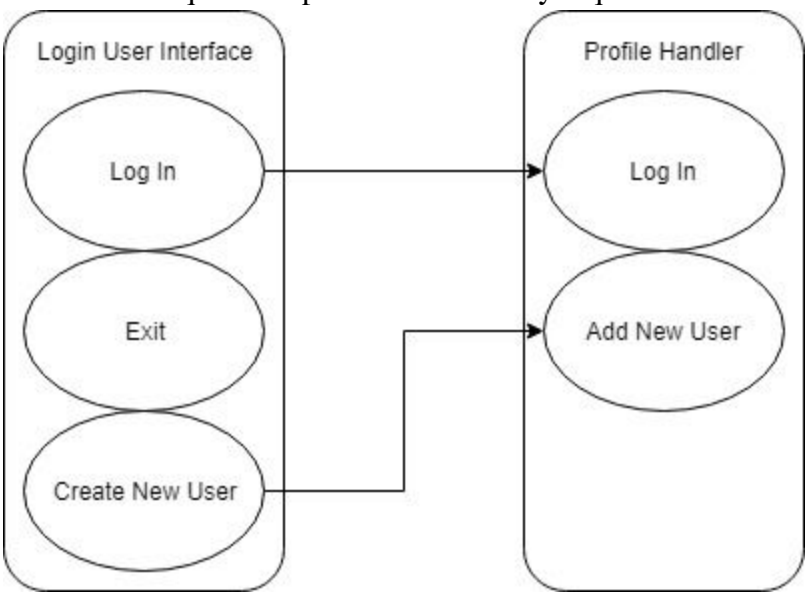
The database for this desktop planner will not need to be extensive, because the amount of necessary data storage is relatively small. As a result, the data used by the planner can simply be stored in text files. There will be one file separately for storing courses, and each type of scheduled item may have its own storage file. The data will be meaningfully formatted in the file such that it can be read and parsed later, however that is the only necessary operation to be applied to the data.

4. Analysis Models

4.1 Use Case Diagrams

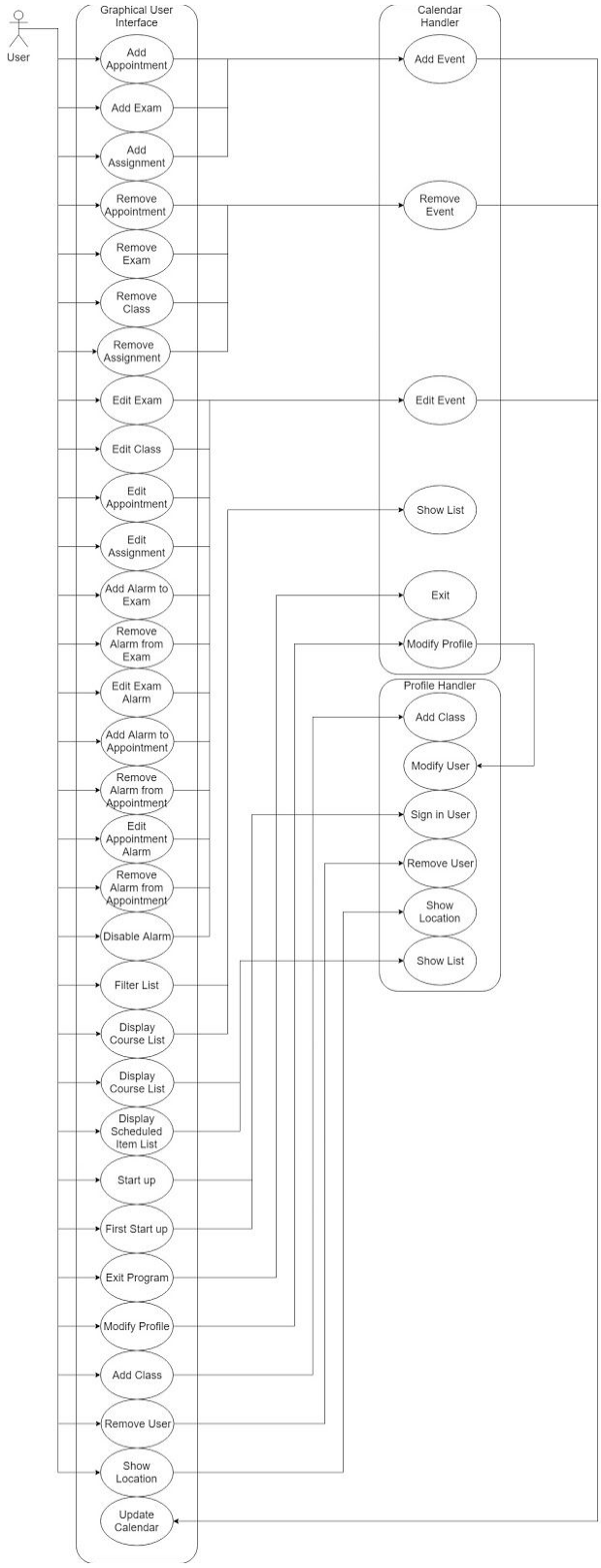
4.1.1 Security Use Case Diagram

This use case diagram displays the user interactions with the security system and what handlers are required to process the security requests



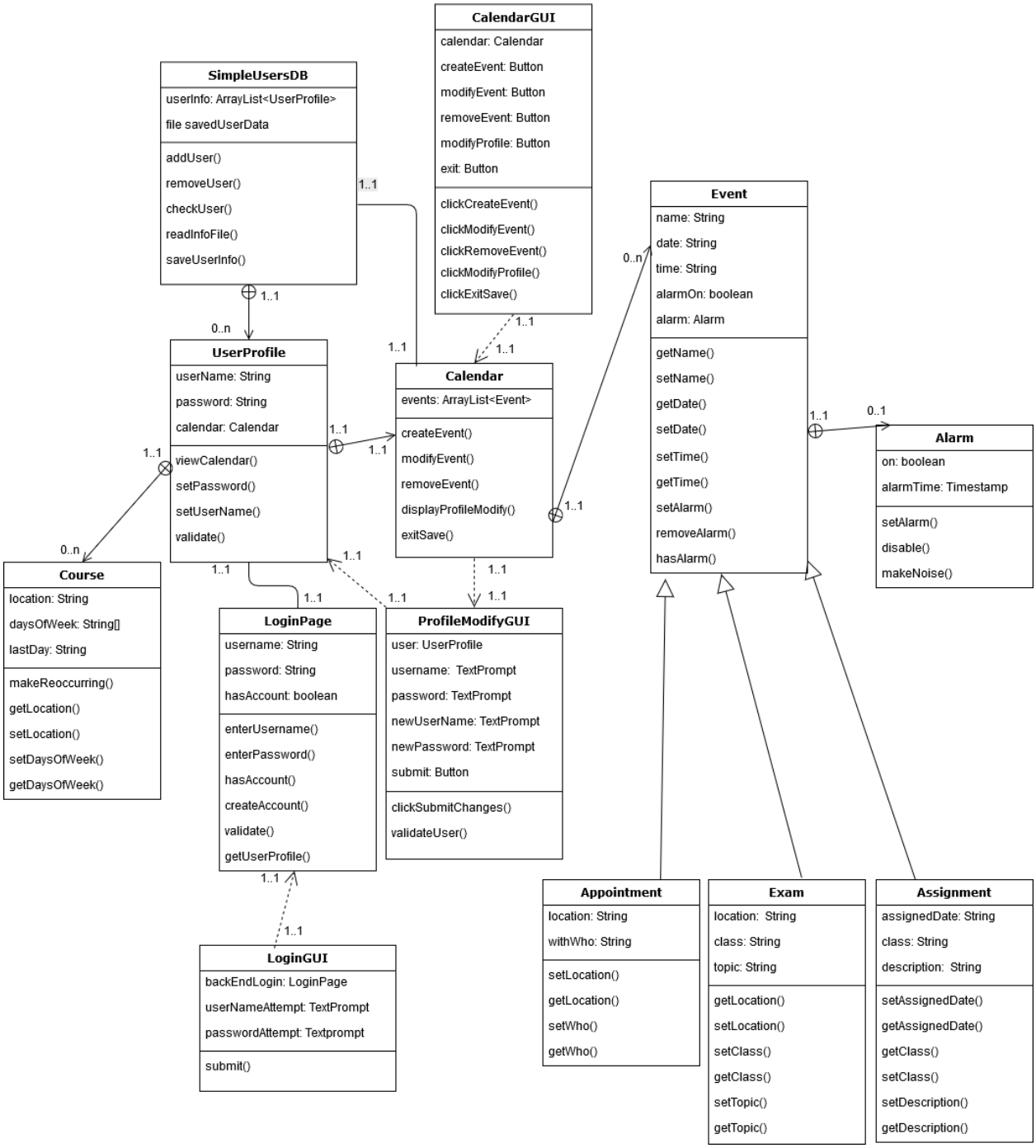
4.1.2 General Use Case Diagram

See Section 3.3 for elaboration of each use case



4.2 Class Diagram

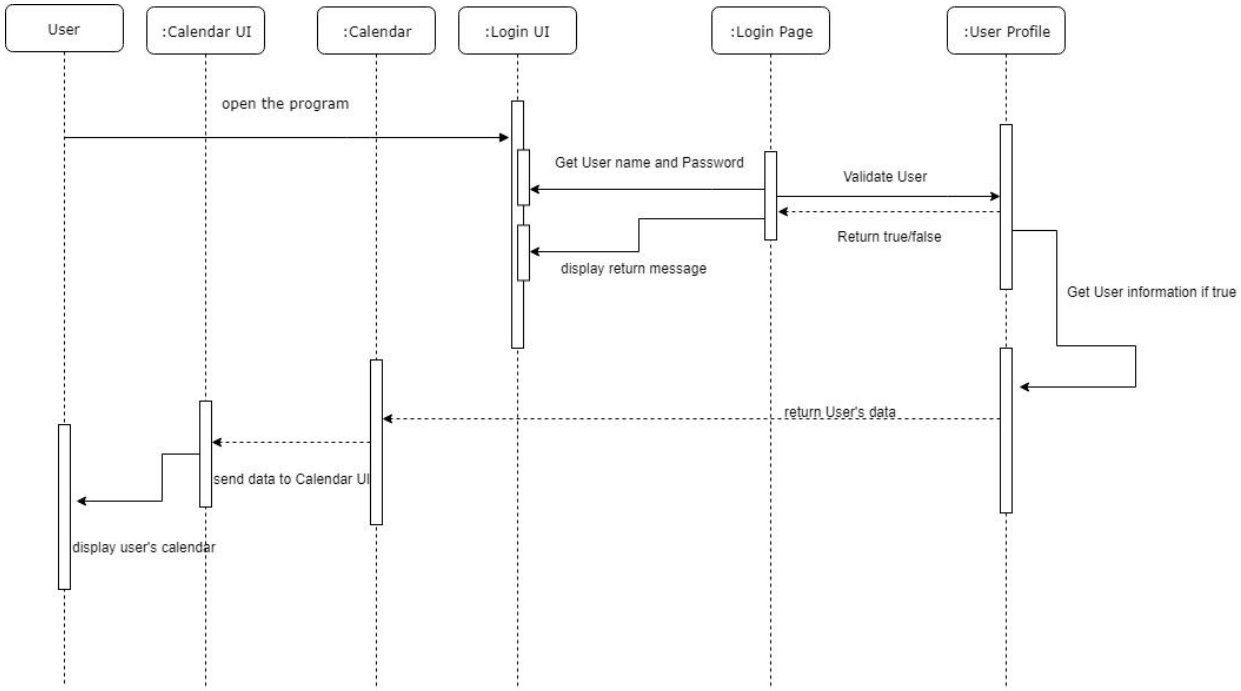
Class attributes and Methods are detailed more clearly in section 3.4.



4.3 Sequence Diagrams

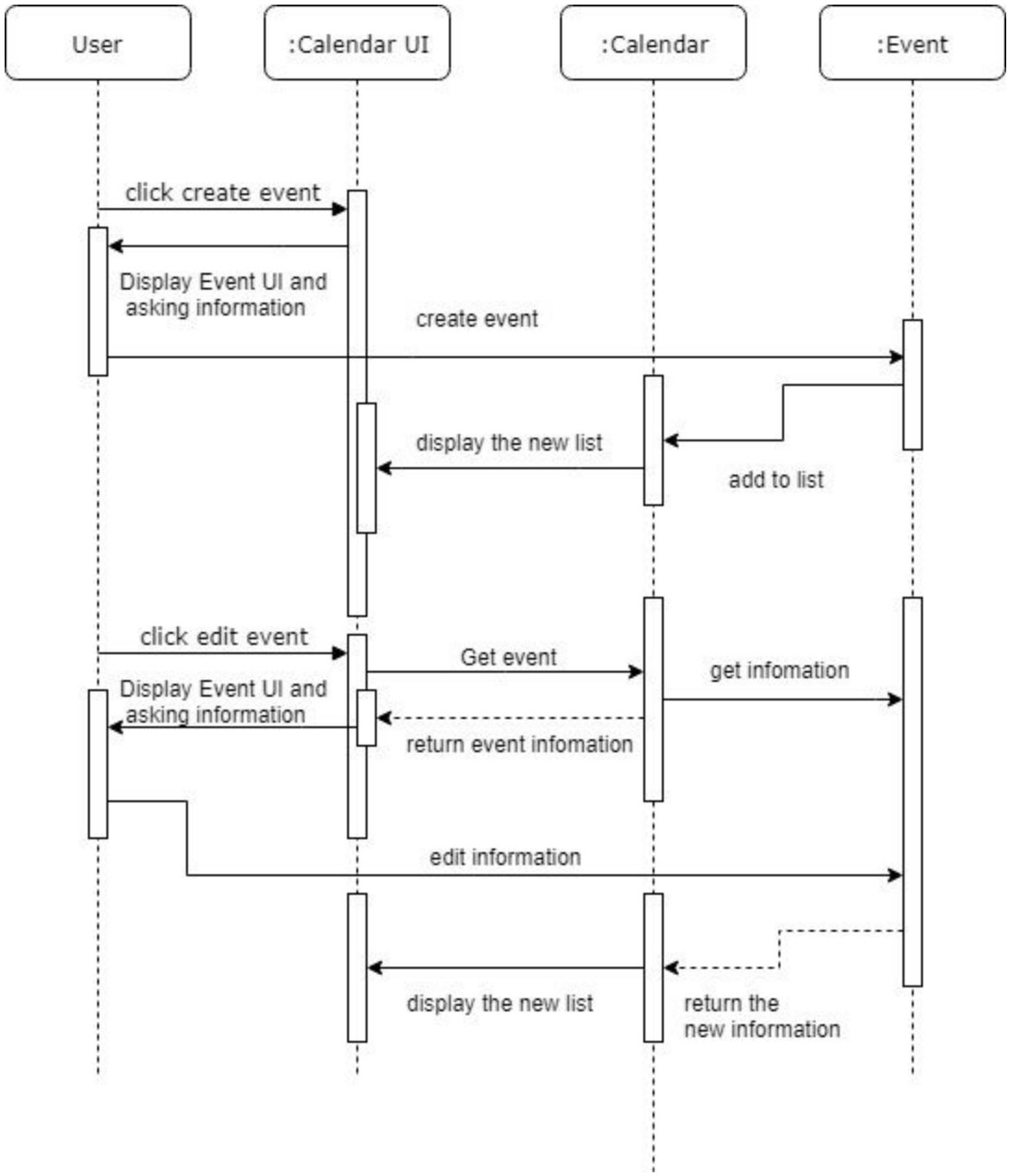
4.3.1 Login Feature(security feature)

When running the program, the user enters their password and username. The GUI pass the information to the user profile to validate the account. The program will then return a message if the account is not found. If account is valid, the program will get the data from user profile and display the calendar.



4.3.2 Create/edit event

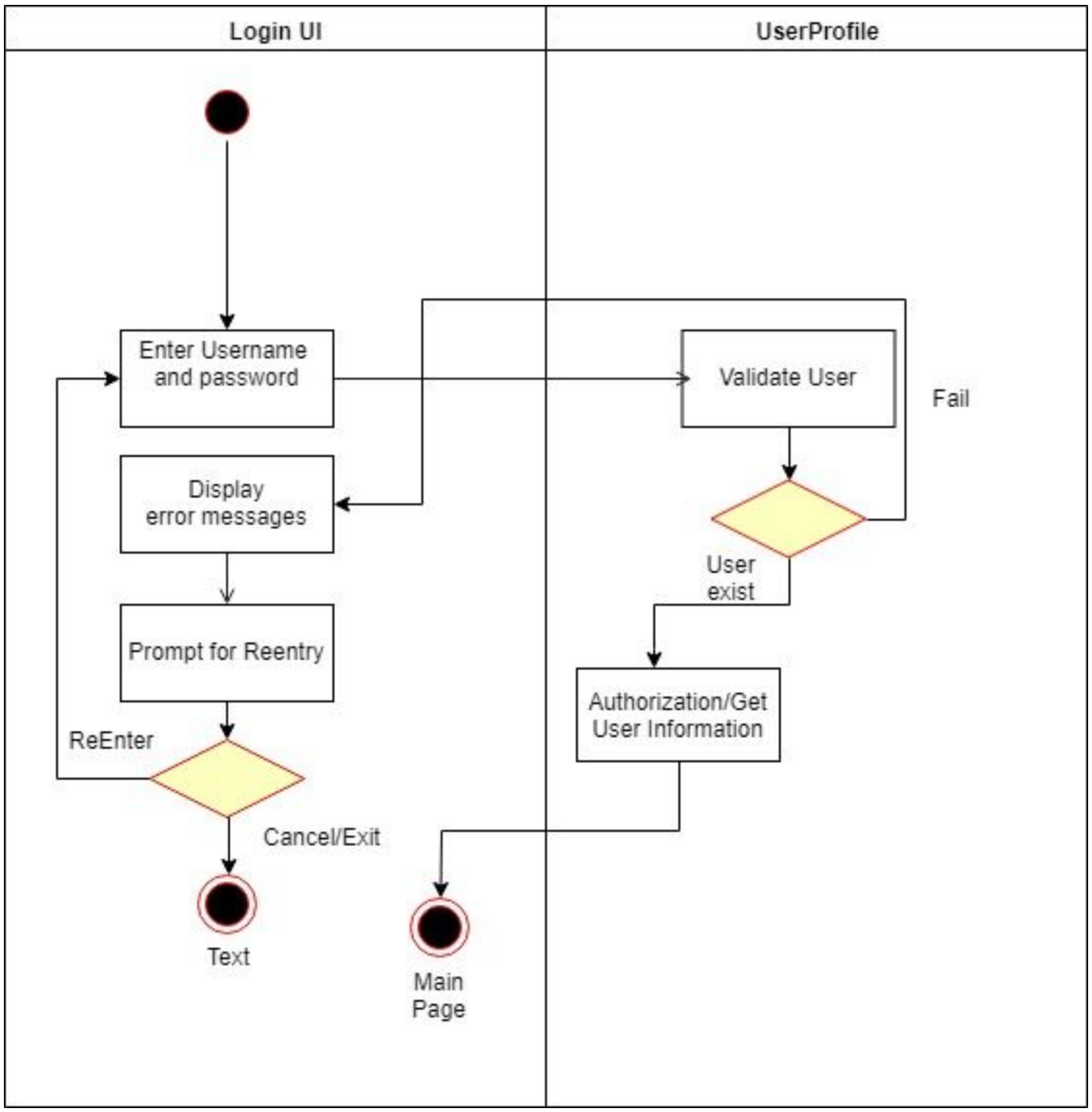
Once the user creates or edits an event, the program will display an UI to collect the information from the user required to create or edit the event. After creating event, the program add it to the calendar and display the new calendar to user.



4.4 Activity Diagrams

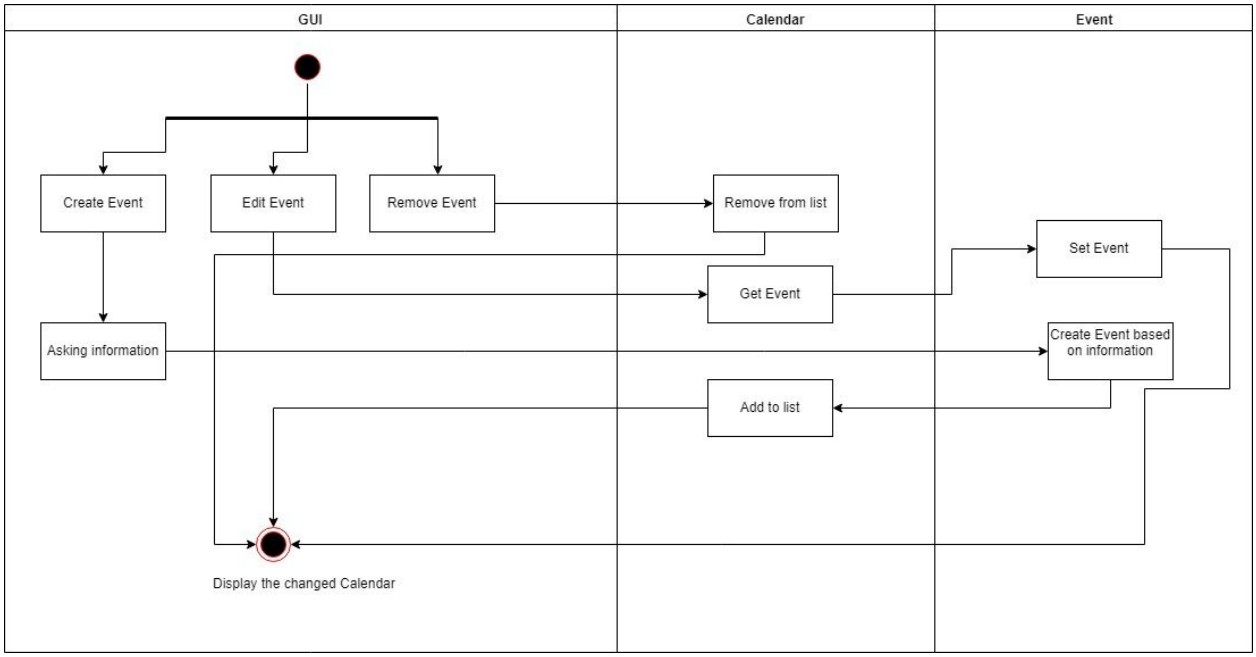
4.4.1 Login feature(security feature)

When running the program, the user will enter their password and username. The GUI will then pass the information to the user profile to validate the account. If the account is valid, the user profile will get the information from disk and have the GUI jump into the main page(calendar). Otherwise, provide user a chance to re-enter a valid account.



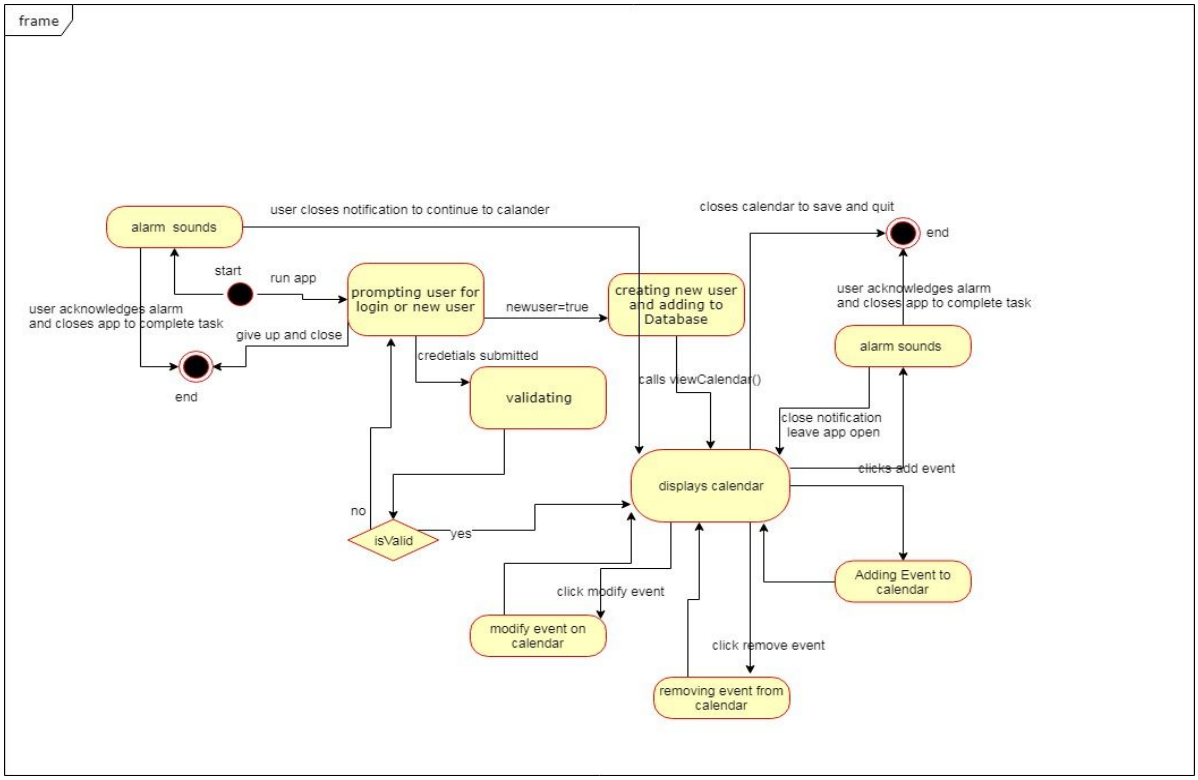
4.4.2 Create/edit/remove event

Once the user creates, edits, or removes an event. The program sends the request to the calendar and event objects and displays an UI for user to edit or create an event. After the user fills out the information from UI it will either change the event information or create a new event and add event to the calendar. If the user chose to remove the event, the event will be removed from the database. At the end, it send the new list to calendar UI and display the new calendar.



4.5 State-Transition Diagram (STD)

For our planner, when the app is not running there are two ways that it can be started. It can either be started either by a user physically running the app or if an alarm timer has already been set by a user for an event. In the case of the alarm, that alarm notification can open up the app or the user can choose to just close the notification and the app will close. In either case, the program switches to a log-in GUI state and the user can either enter their credentials or create a new account. After submitting this the program switches to its calendar view state where a user can look at all of their event details. From this page they can use the interface to create, modify or remove events as they please. Once their done, the user logs out and all information is stored in the simple database. All states of this application are depicted in the diagram below.



5. Change Management Process

All members of the group can update the SRS as project scope or requirements change. Changes must be approved by all members for approval before they become final. Any changes must be logged in the revision history.