# **Team Project: Personal Calendar with Meeting Coordinator**

# **CSC 340 Ethics & Software Engineering**

# **Ivan Olivas**

# **May 6, 2021**

Table of Tables

1. Table of Contents 3
2. Table of Figures 4

Table of Contents

[I. Introduction 3](#_Toc71127845)

[A. Problem Statement 3](#_Toc71127846)

[B. System Proposal 5](#_Toc71127847)

[II. System Description 5](#_Toc71127848)

[III. System Requirements 5](#_Toc71127849)

[A. Functional Requirements 5](#_Toc71127850)

[B. Non-functional Requirements 9](#_Toc71127851)

[IV. Use Case Diagram 10](#_Toc71127852)

[V. Class Diagram 11](#_Toc71127853)

[VI. Sequence Diagrams 12](#_Toc71127854)

[VII. Activity Diagrams 20](#_Toc71127855)

[VIII. State Diagrams 27](#_Toc71127856)

[IX. Database Design 31](#_Toc71127857)

[A. ER Diagram 31](#_Toc71127858)

[B. Table Schema 32](#_Toc71127859)

[X. Conclusion 33](#_Toc71127860)

[XI. Data Dictionary 34](#_Toc71127861)

Table of Figures

[Figure 1 - Use Case Diagram 10](file:///E:\340\TeamReport_Final.docx#_Toc71214815)

[Figure 2 - Class Diagram of Calendar System 11](file:///E:\340\TeamReport_Final.docx#_Toc71214816)

[Figure 3 - Sequence Diagram for Signing In the system 12](file:///E:\340\TeamReport_Final.docx#_Toc71214817)

[Figure 4 - Sequence Diagram for signing out of the system 13](file:///E:\340\TeamReport_Final.docx#_Toc71214818)

[Figure 5 - Sequence Diagram for viewing an event 14](file:///E:\340\TeamReport_Final.docx#_Toc71214819)

[Figure 6 - Sequence Diagram for viewing events based on the month 15](file:///E:\340\TeamReport_Final.docx#_Toc71214820)

[Figure 7- Sequence Diagram for adding an event 16](file:///E:\340\TeamReport_Final.docx#_Toc71214821)

[Figure 8 - Sequence Diagram for deleting an event 17](file:///E:\340\TeamReport_Final.docx#_Toc71214822)

[Figure 9- Sequence Diagram for editing an event 18](file:///E:\340\TeamReport_Final.docx#_Toc71214823)

[Figure 10 - Sequence Diagram for coordinating an event from the manager 19](file:///E:\340\TeamReport_Final.docx#_Toc71214824)

[Figure 11 Activity Diagram for signing in the system 20](file:///E:\340\TeamReport_Final.docx#_Toc71214825)

[Figure 12 - Activity Diagram for signing out of the system 21](file:///E:\340\TeamReport_Final.docx#_Toc71214826)

[Figure 13 - Activity Diagram for viewing an event 21](file:///E:\340\TeamReport_Final.docx#_Toc71214827)

[Figure 14 - Activity Diagram for viewing events based on the month 22](file:///E:\340\TeamReport_Final.docx#_Toc71214828)

[Figure 15- Activity Diagram for adding an event 23](file:///E:\340\TeamReport_Final.docx#_Toc71214829)

[Figure 16 - Activity Diagram for deleting an event 24](file:///E:\340\TeamReport_Final.docx#_Toc71214830)

[Figure 17 - Activity Diagram for editing an event 25](file:///E:\340\TeamReport_Final.docx#_Toc71214831)

[Figure 18 - Activity Diagram for coordinating a meeting from a manager 26](file:///E:\340\TeamReport_Final.docx#_Toc71214832)

[Figure 19 State diagram for signing in and adding an event 27](file:///E:\340\TeamReport_Final.docx#_Toc71214833)

[Figure 20 - State Diagram for deleting and editing an event 28](file:///E:\340\TeamReport_Final.docx#_Toc71214834)

[Figure 21- State Diagram for viewing an event, viewing monthly events, and for signing out of the system 29](file:///E:\340\TeamReport_Final.docx#_Toc71214835)

[Figure 22- State diagram for coordinating an event 30](file:///E:\340\TeamReport_Final.docx#_Toc71214836)

[Figure 23 - ER Diagram for database 31](#_Toc71214837)

# Introduction

## Problem Statement

A client from company needs a simple personal calendar system for his employees to use personally or also for work. Currently, they do not have a calendar system in place that allows employees to schedule and manage events. They greatly need a software system to serve as a calendar to schedule any events or meetings they might have. In addition, the manager would like the ability to schedule an event or meeting based on the availability of all the employees. It takes too much time right now to find the best time for everyone in the team to meet so having a system that finds the best time for each individual would save a lot of time and allow everyone to focus on their tasks.

## System Proposal

We propose a software system to help the company with the desired functionality of the system. A software system would be the best approach to provide all the needs of the client.

# System Description

The software system will be a simple calendar system for personal use. The system shall provide the following services to a user:

* Sign in & sign out of the system

• Add an event to the calendar, including conflict check of two events.

• Delete an event from the calendar.

• Edit an event on the calendar.

• View an event.

• View a monthly-based event list.

Beside the above basic services to a personal use of the system, the system shall allow a manager to coordinate a meeting for his/her teammates. To do this, the manager first chooses a date for holding a meeting, the duration of the meeting, and the members to attend the meeting. The system will then access the members’ calendars and find possible time slots for holding the meeting. The system will display a list of possible time slots to the manager. The manager will choose one from the list. The meeting event will be added to every participant’s calendar.

# System Requirements

## Functional Requirements

1. **The system shall allow a member to add an event to the calendar.**
   1. The system shall allow a member to select “Add Event” button.
   2. The system shall display a form regarding event information for the member to complete.
      1. The system shall get the current selected day, month, and year based on the member’s selection.
         1. The system shall check if the member date selection is in the past.
            1. If the date selection is in the past, the system shall display an error message saying, “Event can only be added for current or future dates.” Return to step 1.2.1.
            2. If the date selection is the current or future date, continue to step 1.2.2.
      2. The system shall ask the member for an event name.
         1. The member shall enter an event name.
      3. The system shall ask the member for the start time of the event.
         1. The member shall enter a start time of the event.
      4. The system shall ask the member for end time of the event.
         1. The member shall enter the end time of the event.
      5. The system shall ask the member for a description of the event.
         1. The member shall enter a description of the event.
      6. The system shall ask the member for the location of the event.
         1. The member shall enter the location of the event.
      7. The system shall request for the member’s confirmation to add the event.
         1. If the member confirms the events, continue to step 1.2.8.
         2. If the member cancels the event addition, return to step 1.1.
      8. The system shall check if there is a conflict between two events.
         * 1. If the system finds the time entered overlaps with another event, the system shall display an error message to the member.
           2. If there are no overlapping events, the system shall add the event to the member’s calendar data and continue to step 1.3.
   3. The system shall make a connection to the database.
   4. The system shall update and store the event to the database.
   5. The system shall display a successful message to the member.
   6. The system shall close the connection to the database.
2. **The system shall allow a member to delete an event from the calendar.** 
   1. The system shall allow the member to select “Delete Event” button.
   2. The system shall display a list of events to the member that can be selected for deletion.
      1. The system shall retrieve the events from the database based on the selected day.
      2. The system shall format and display the list of events to the member.
         1. If the database does not contain any events, the system shall display a message saying, “No events for this day” to the member.
         2. If the database is not empty, the system shall display the events to the member.
   3. The member shall select an event to delete.
   4. The system shall request confirmation from the member to delete an event.
   5. The member shall confirm the action of “delete event”.
      1. If the member agrees to delete the event. The system shall delete the event from the list and continue to step 2.6.
      2. If the member does not agree to delete the event. The system shall not delete the event and return to step 2.3.
   6. The system shall make a connection to the database.
   7. The system shall delete the event from the database.
   8. The system shall save changes made to the database.
   9. The system shall display a successful message of deletion to the member.
   10. The system shall close the connection to the database.
3. **The system shall allow a member to edit an event from the calendar.**
   1. The system shall allow the member to select “Edit Event” button.
   2. The system shall display a list of events to the member that can be selected to edit.
      1. The system shall retrieve the events from the database based on the selected day.
      2. The system shall format and display the list of events to the member.
         1. If the database does not contain any event, the system shall display a message “No events for this day” to the member.
         2. If the database is not empty, the system shall display the event to the member.
   3. The member shall select an event to edit.
   4. The system shall allow the member to edit the event name, start time of the event, end time of the event, description, and location of the event.
   5. The member shall enter their desired changes.
   6. The member shall save the changes made.
   7. The system shall make a connection to the database.
   8. The system shall update the event in the database.
   9. The system shall display a successful message to the member.
   10. The system shall close the connection to the database.
4. **The system shall allow a member to view an event from the calendar.** 
   1. The system shall allow the member to select “View Event” button.
   2. The system shall display a list of events to the member that are scheduled for that day.
      1. The system shall retrieve the events from the database based on the selected day.
      2. The system shall format and display the list of events to the member.
         1. If the database does not contain any events, the system shall display a message saying, “Sorry, there are no events for this day” to the member.
         2. If the database is not empty, the system shall display the events to the member.
   3. The member shall select one event at a time from the list to display all the details of that event.
      1. The system retrieves all the details of the event selected.
      2. The system shall display all the details of the event selected.
   4. The system shall close the connection to the database.
5. **The system shall allow a member to view a monthly-based events list.**
   1. The system shall allow the member to select “View Monthly-based Event List” button.
   2. The system shall recognize the current month selected from the calendar.
   3. The system shall display a list of events to the member that are scheduled for that month.
      1. The system shall retrieve the events from the database based on the selected month.
      2. The system shall format and display the list of events within the month to the member.
         1. If the database for that month does not contain any events, the system shall display a message saying, “Sorry, there are no events for this month” to the member.
         2. If the database is not empty, the system shall display the events to the member.
      3. The system shall close the connection to the database.
6. **The system shall allow a user to sign into the system to access their calendar.**
   1. The user will be prompted to input their username and password.
   2. The user shall enter their username and password.
   3. The system shall validate the user’s username and password.
      1. The system shall check if the username and password is in the database.
      2. If the log-In combination is found in the database, the customer will be presented their calendar and menu.
      3. If log-In combination is not found in database, an error message (Invalid Log-In) shall be displayed. The system will then return to Step 6.1.
7. **The system shall allow a member to sign out from their calendar system.**
   1. From the main calendar menu, the system shall display an “Exit” Button.
   2. The member shall select “Exit” in the menu.
   3. The system shall display the log-In page to the user.
8. **The system shall allow a manager to coordinate a meeting.**
   1. From the main calendar menu, the system shall display a “Coordinate” button.
   2. The manager shall select “Coordinate” in the menu.
   3. The system shall display a form regarding event information for the manager to complete.
      1. The system shall get the current selected day, month, and year based on the manager’s selection.
         1. The system shall check if the manager date selection is in the past.
            1. If the date selection is in the past, the system shall display an error message saying, “Event can only be added for current or future dates.” Return to step 8.3.1.
            2. If the date selection is the current or future date, continue to step 8.3.2.
      2. The system shall ask the manager for an event name.
         1. The manager shall enter an event name.
      3. The system shall ask the manager for a description of the event.
         1. The manager shall enter a description of the event.
      4. The system shall ask the manager for the location of the event.
         1. The manager shall enter the location of the event.
      5. The system shall ask the manager for the duration of the event.
         1. The manager shall enter the duration of the event.
      6. The system shall ask the manager for the members that should attend the meeting.
         1. The system shall retrieve the members in the database system.
         2. The system shall display the members list.
         3. The manager shall select the members to attend the meeting using first and last name.
      7. The system shall get the manager’s confirmation to search for member’s availability for meeting within that day.
         1. If the manager confirms the event, continue to step 8.4.
         2. If the manager cancels the event addition, return to step 8.3.
   4. The system shall find the available times for all the event attendees to meet.
   5. The system shall display all the available times found for attendees to meet.
   6. The manager shall select a time for the event.
      1. The system shall get the manager’s confirmation for the selected to create the event.
         1. If the manager confirms the time, continue to step 8.7.
         2. If the manager cancels the event time, return to step 8.5.
   7. The system shall make a connection to the database.
   8. The system shall create the event.
   9. The system shall add the event to the manager and attendee’s calendar.
   10. The system shall store the event to the database.
   11. The system shall display a successful message to the manager.
   12. The system shall close the connection to the database.

## Non-functional Requirements

NR1. The system will not allow the scheduling of an event in the past.

NR2. The system will not allow the scheduling of two events at the same times or overlapping of two events.

NR3. Each username must be unique.

NR4. Members and managers will be inserted manually into the system.

NR5. Manager status will be set manually.

NR6. Only a manager will have access to the coordinate functionality.

NR7. The calendar system can only serve one member/user/manager at a time.

# Use Case Diagram

**Diagram, schematic

Description automatically generated**

Figure - Use Case Diagram

This is a diagram to show the functionality of the system with use cases, including logging in and out of the system, along with being able to add, edit, and delete events. In addition, there will be two kinds of views for selection. You may select to view events of a certain day or view events of a certain month. A manager will have extra functionality by being able to coordinate a meeting and schedule it on other member’s calendar systems.

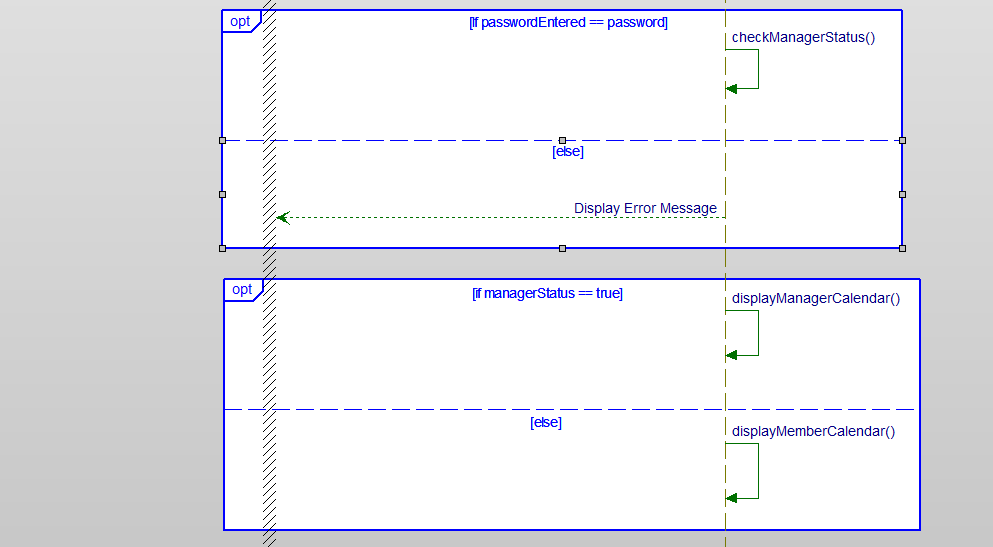
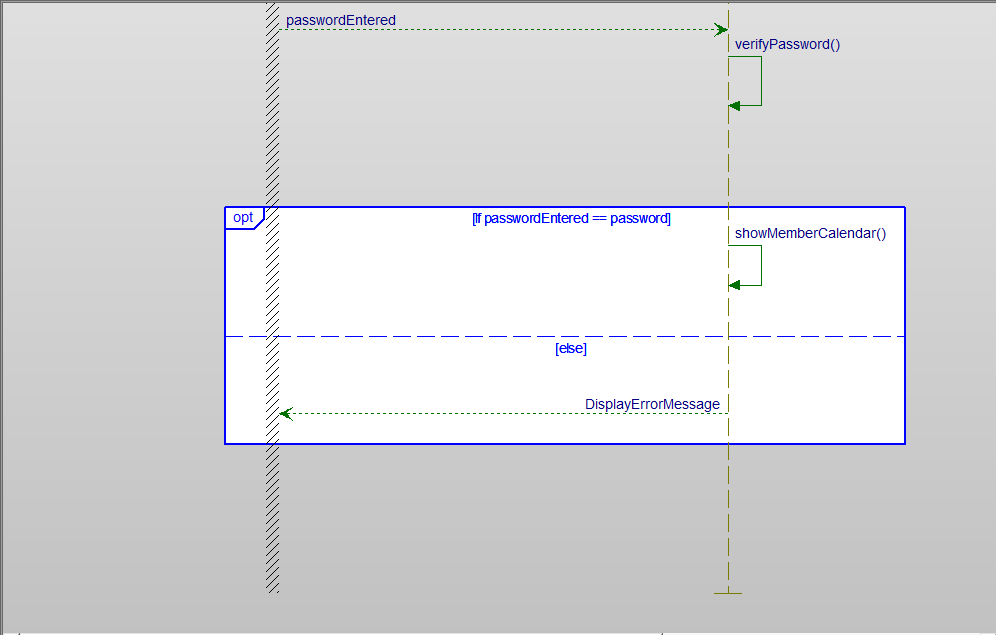
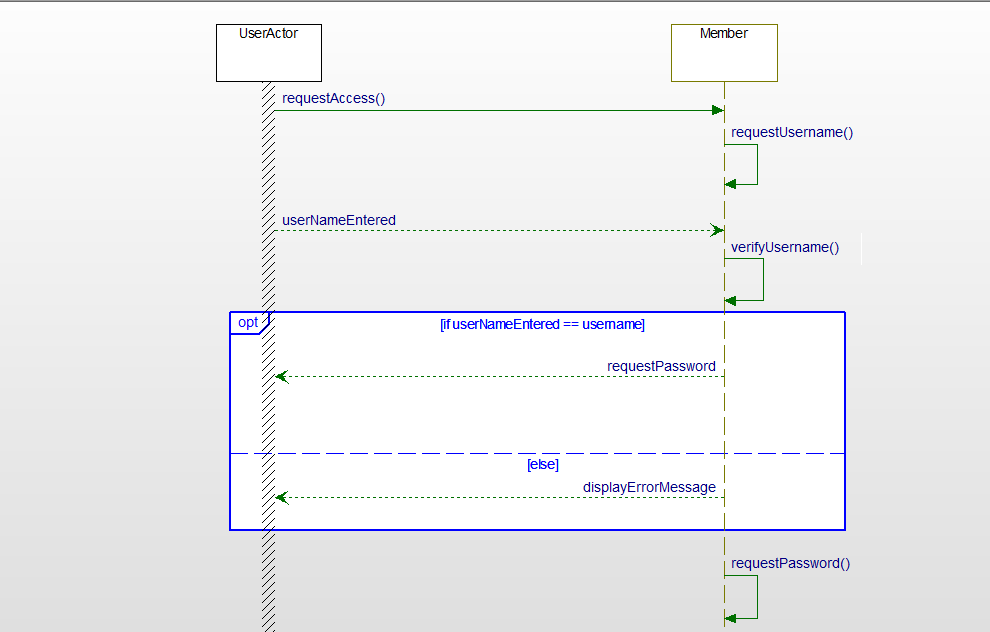
# **A picture containing text Description automatically generated**Class Diagram

Figure - Class Diagram of Calendar System

This is a diagram to show the related attributes and operations within each class that the system will use to perform the functionalities needed. The managerEvent will serve as a sub-entity using all the attributes from event with attendees as the additional attribute.

# Sequence Diagrams

Figure - Sequence Diagram for Signing In the system



When signing in the system shall have error handling to check if the correct login was entered and if the user is a manager or regular member in order to display the correct menu.

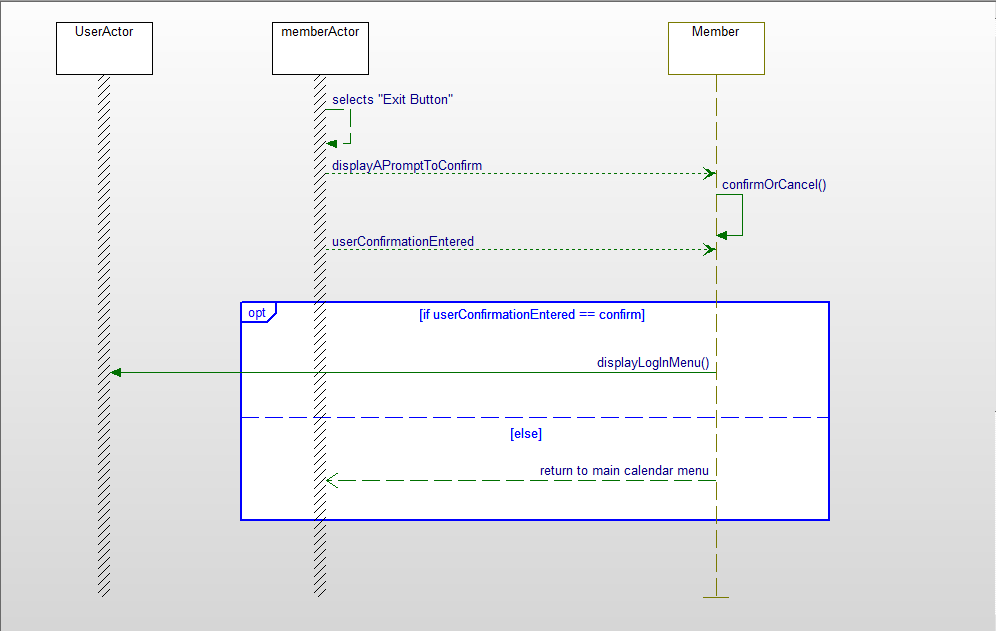
When a member signs out the screen will return to the login menu.

Figure 4 - Sequence Diagram for signing out of the system

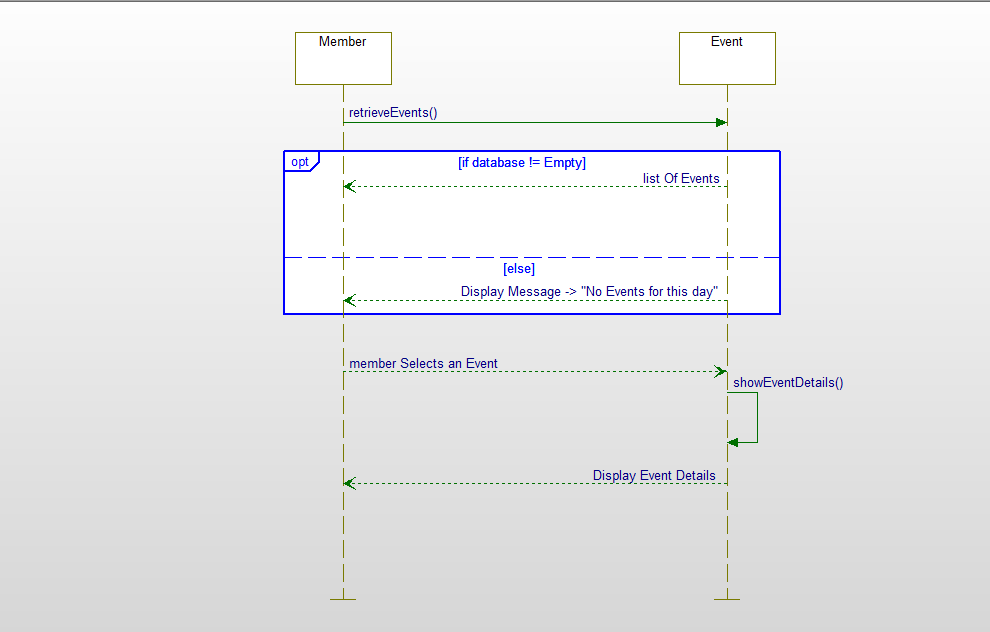


Figure 5 - Sequence Diagram for viewing an event

In order to view an event, the system shall check if the database is empty and display the event or an error message. The user will have a list of events to selected from for the day selected. Once selection occurs the panel will display all the information regarding the event to the member or manager.

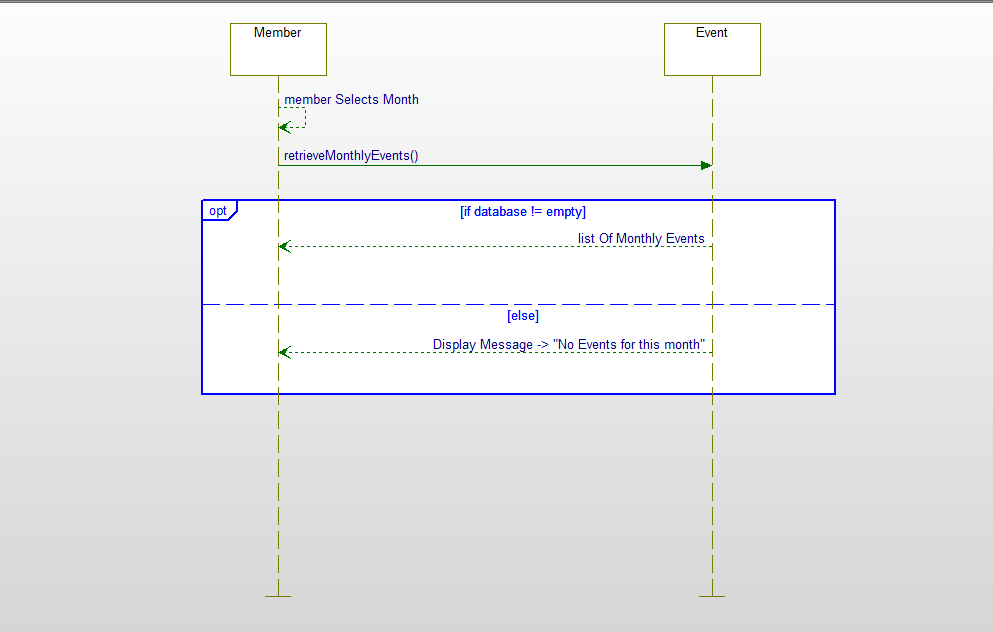
The system shall search the database of events on a month specified by the user and return that list or a message if no events were found.

Figure 6 - Sequence Diagram for viewing events based on the month

The system will check if the date the member or manager enters is in the present or future and if there are any overlapping event in order to add the event to the system and database.

Figure 7- Sequence Diagram for adding an event

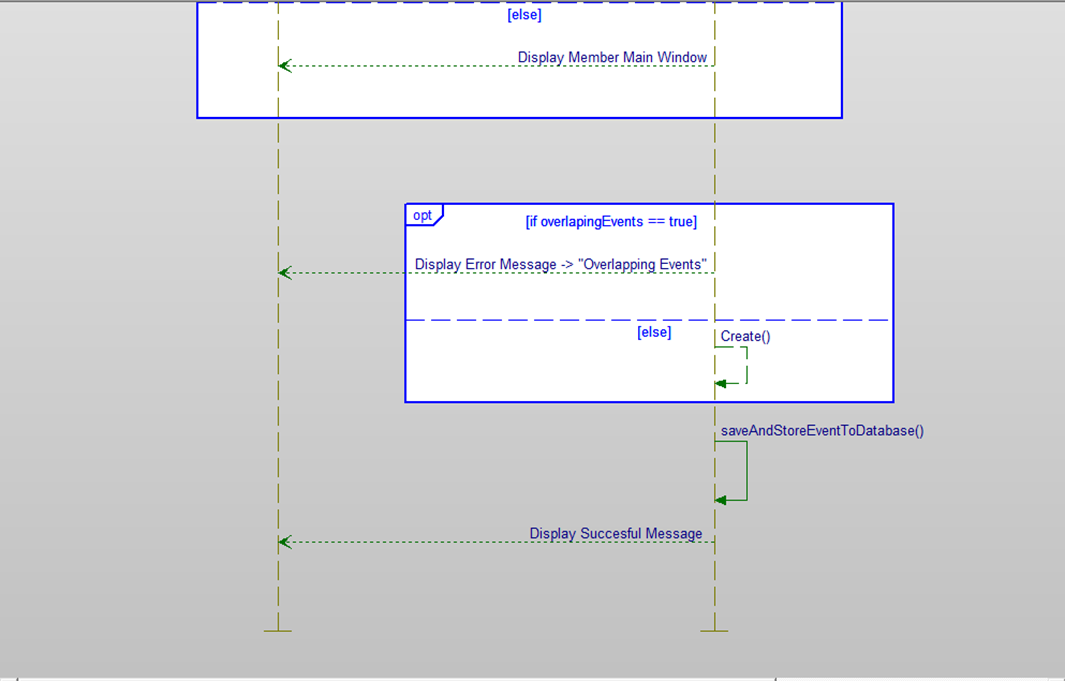
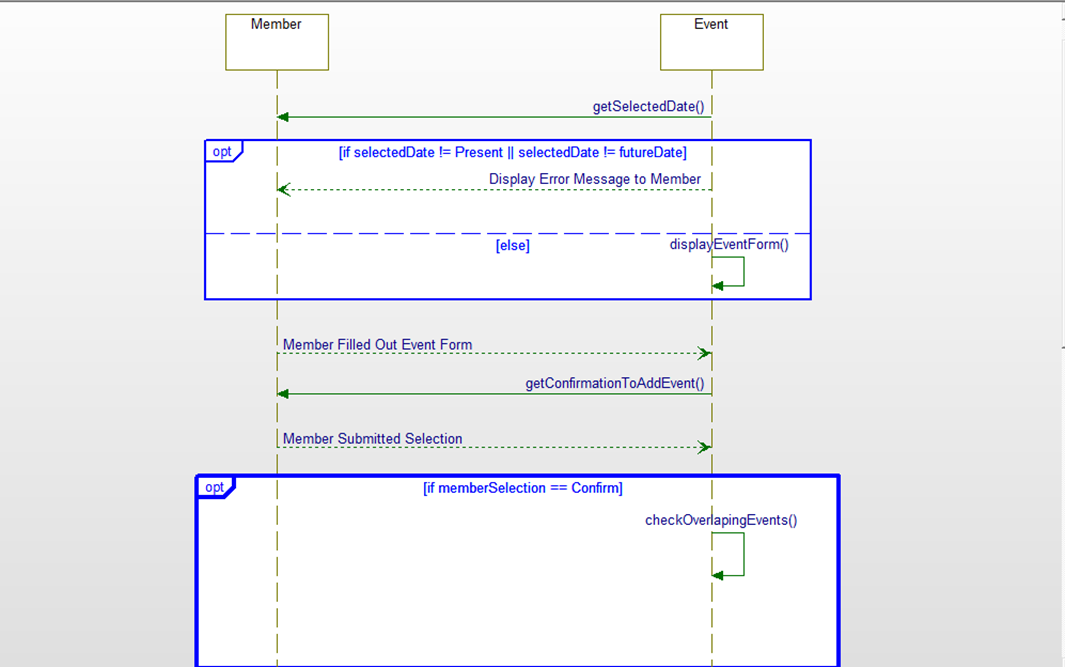
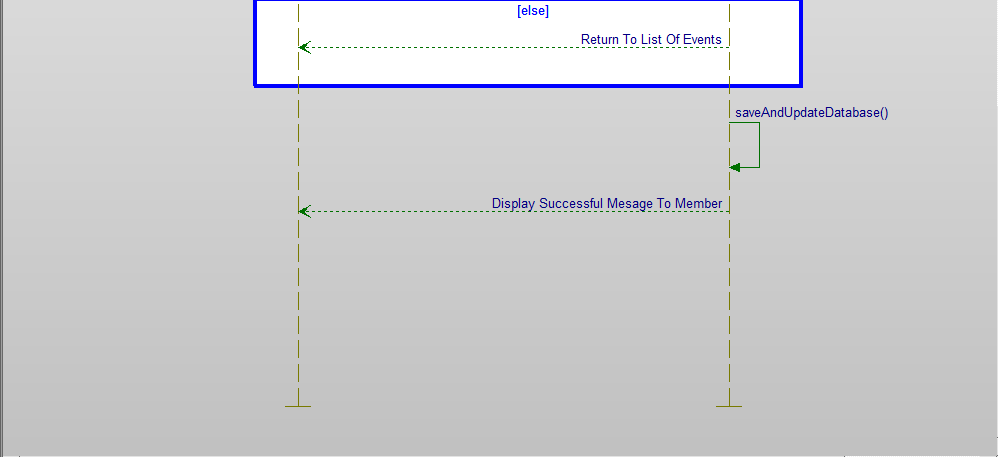
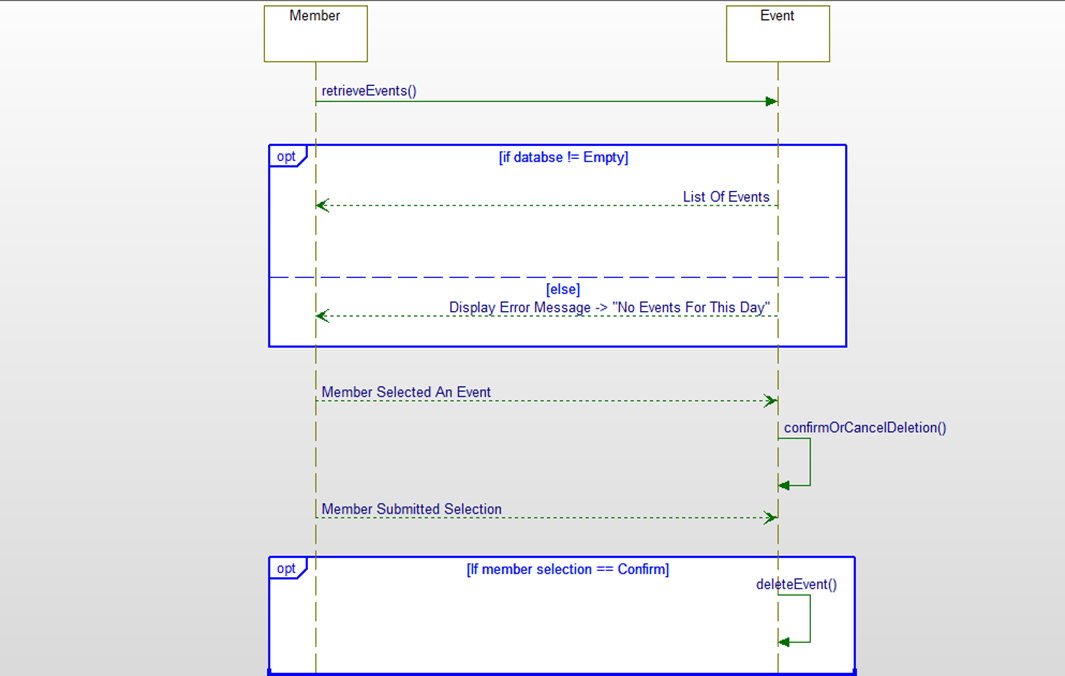


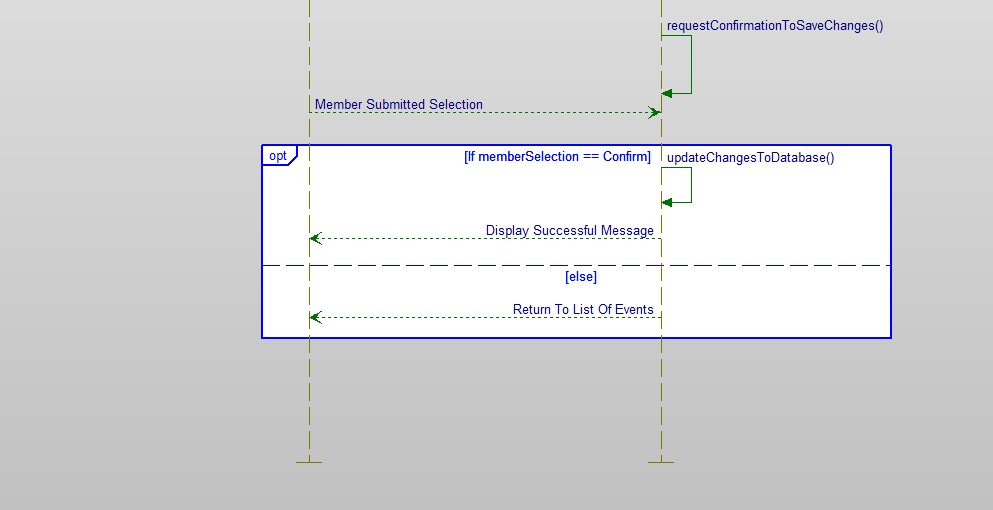
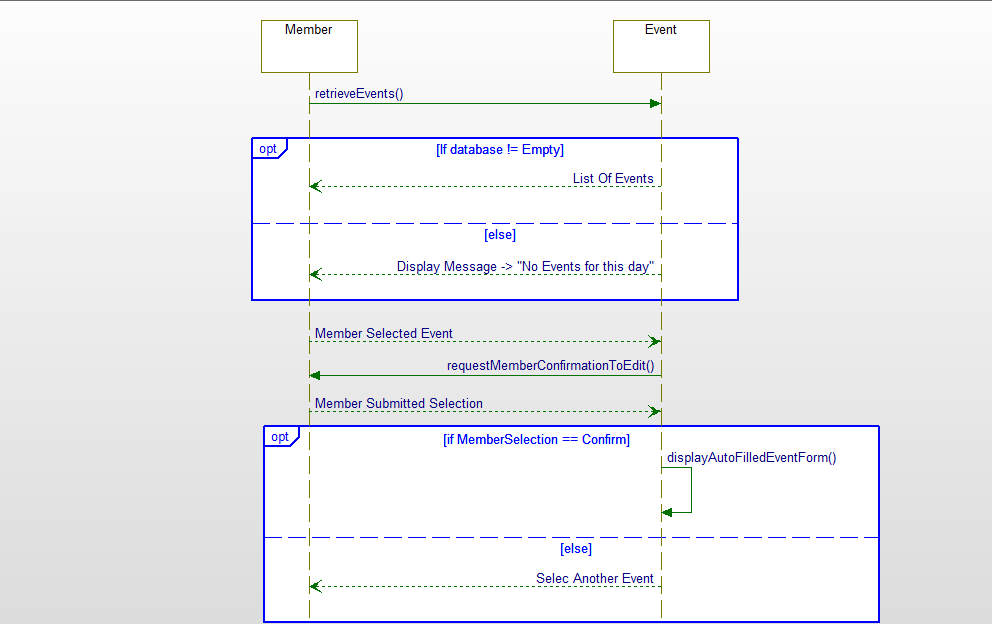
Figure 8 - Sequence Diagram for deleting an event



The system will need a confirmation from the manager or member to delete the event in the system or database. The system will save that to the database.

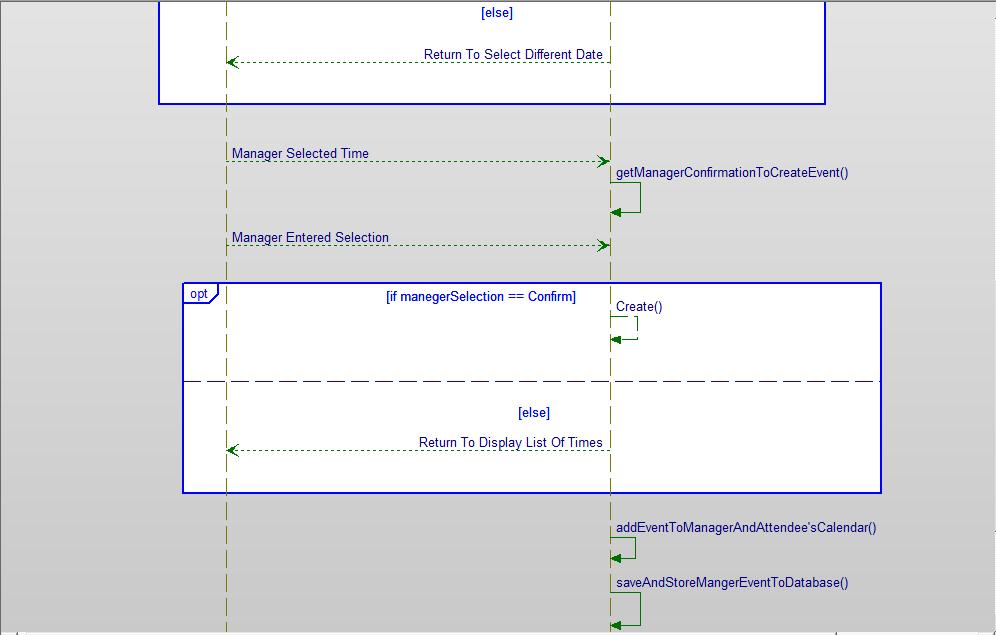
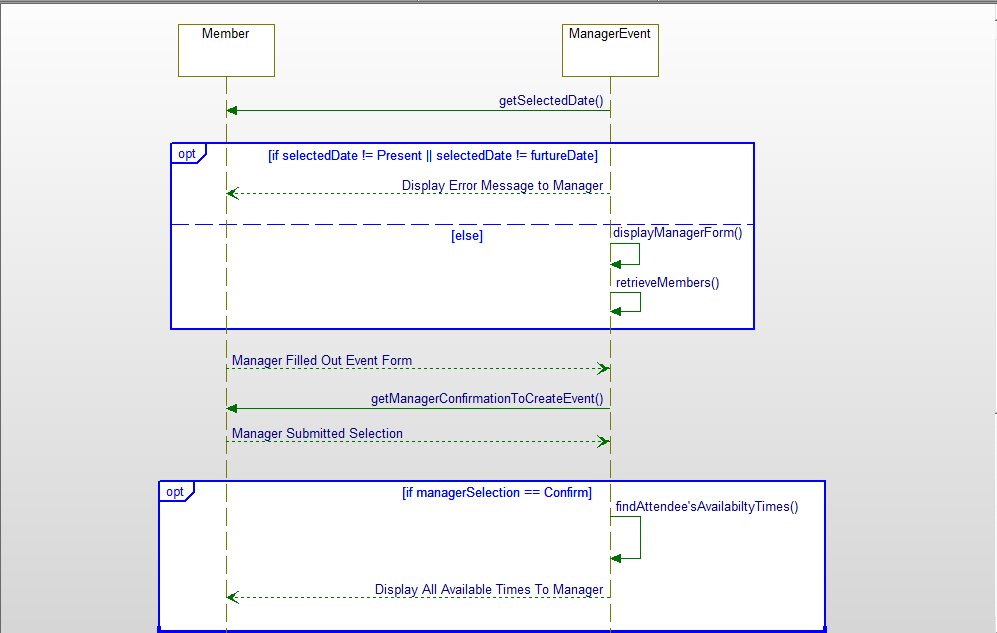
The system shall allow the user to change any attributes regarding the event they selected, and the system shall save it to the database.

Figure 9- Sequence Diagram for editing an event



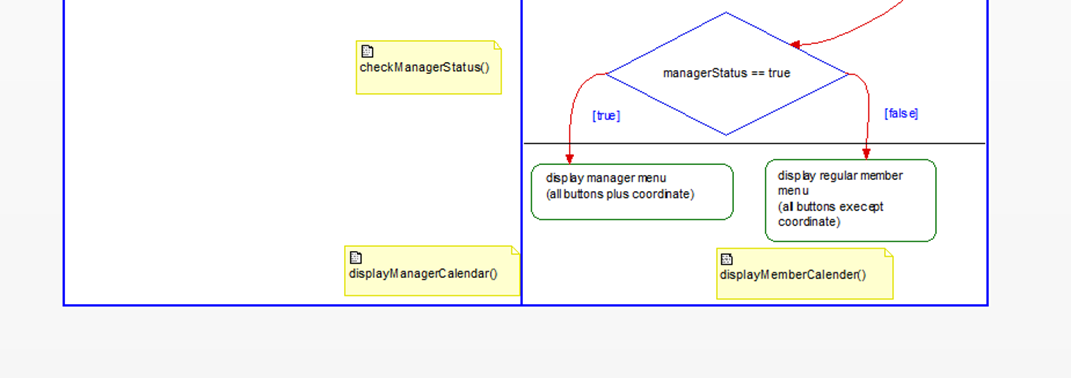
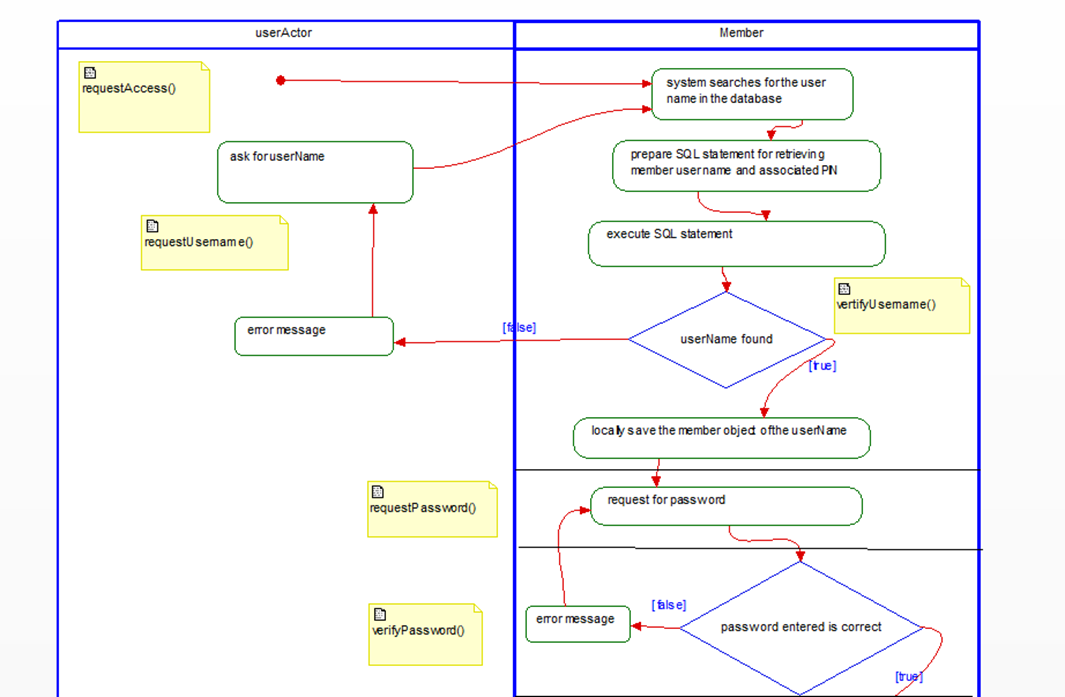
The system shall allow a manager to coordinate a meeting by allowing he /she to select a desired date for the meeting. In addition, based on the attendees the manager selects to have at the meeting the system will find the best times for everyone, and give the manager the options to select the time based on what the system found.

Figure 10 - Sequence Diagram for coordinating an event from the manager



# Activity Diagrams

Figure 11 Activity Diagram for signing in the system



For signing in, the system will determine various conditional statements, not only to sign into the system but to also provide the correct functionality. A manager should be given the option to coordinate a meeting while a regular member should not have the capability.

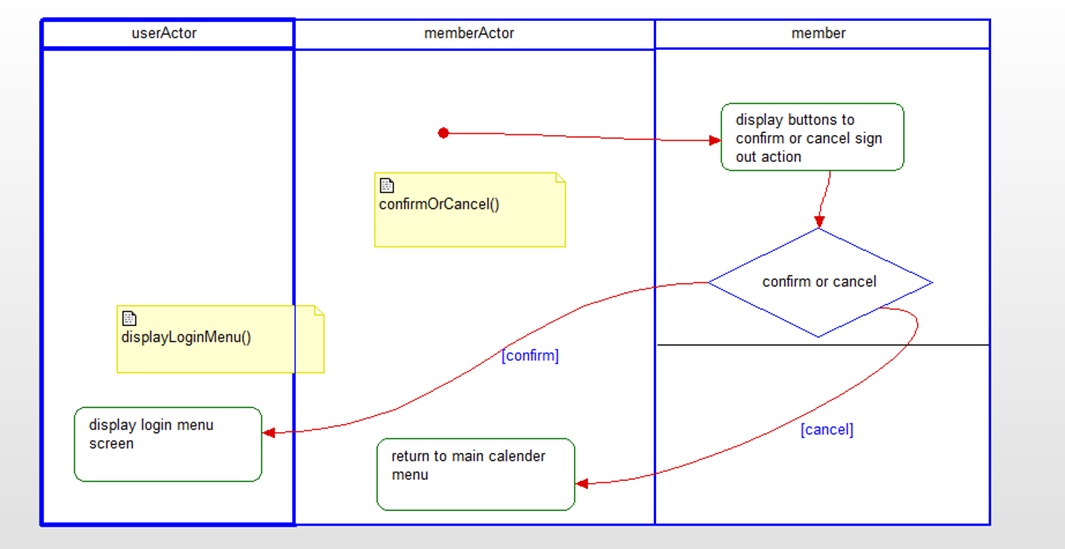
The manager or member must confirm signing out in order to be taken back to the sign in menu.

Figure 12 - Activity Diagram for signing out of the system

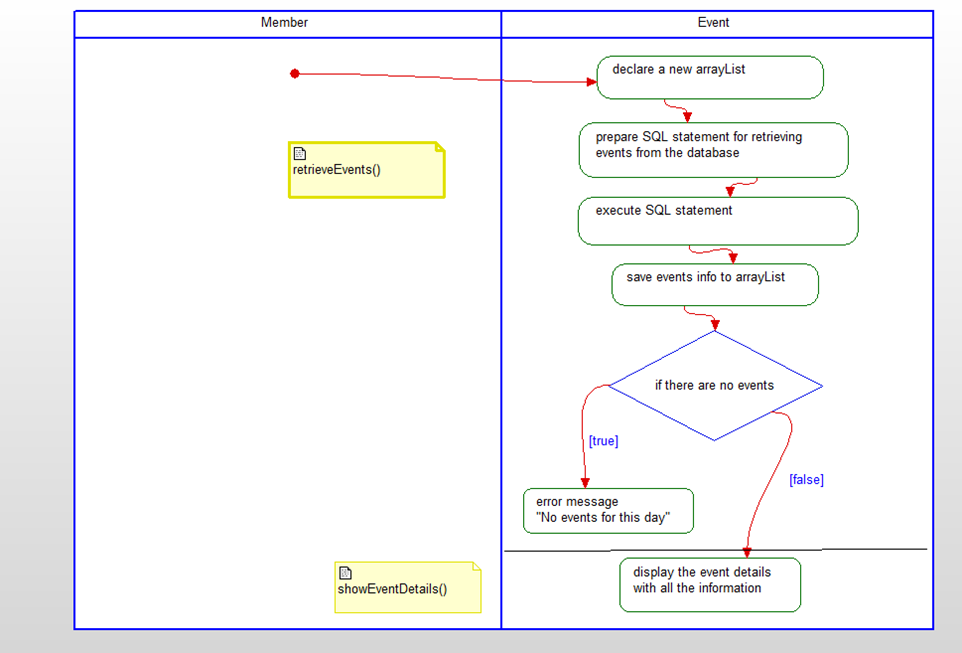


Figure 13 - Activity Diagram for viewing an event

The use case for viewing an event will primarily use two methods to provide the functionality. It should retrieve the events based on the day selected and show details based on the selection.

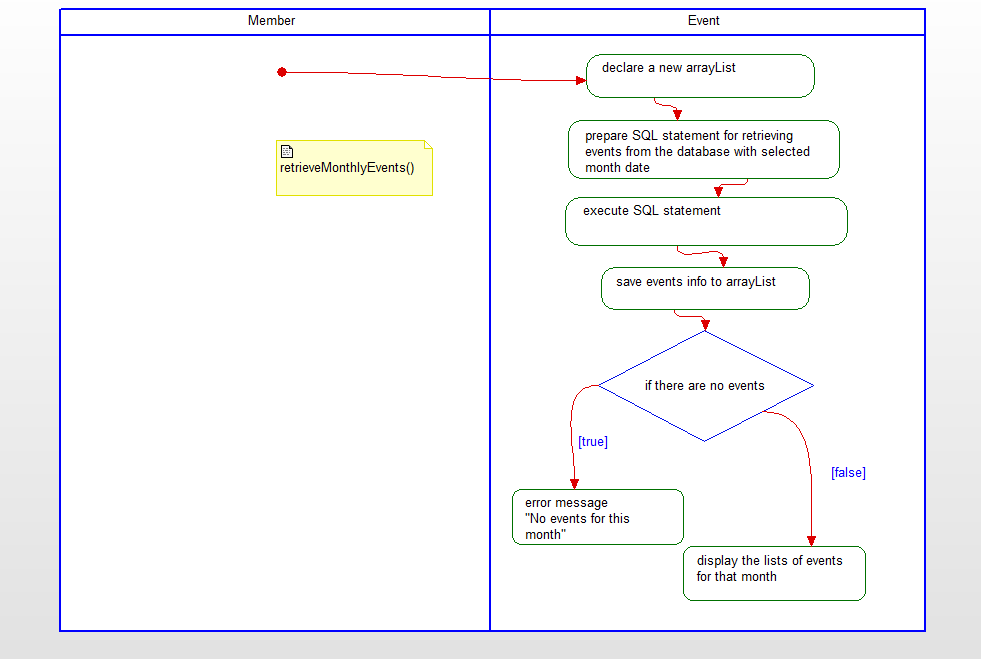
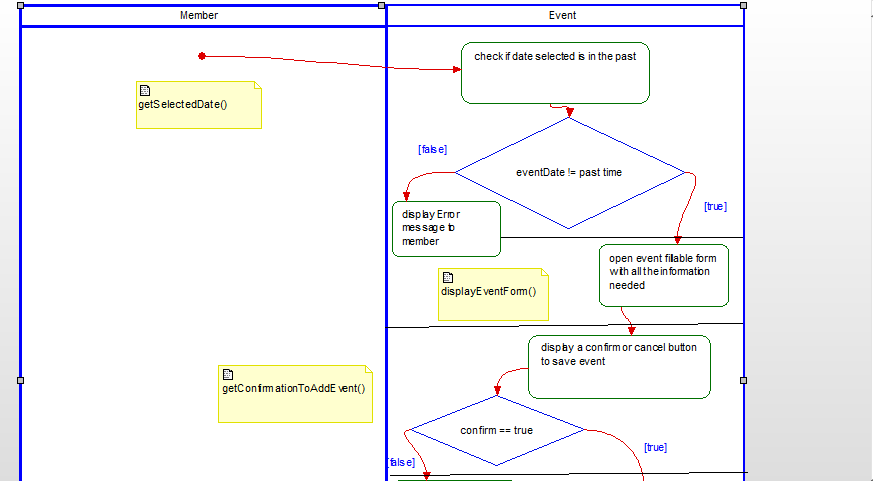
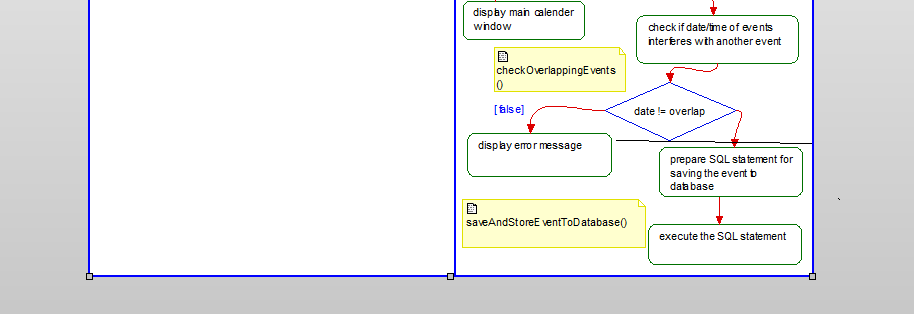


Figure 14 - Activity Diagram for viewing events based on the month

The view monthly events use case will primarily use one function in order to retrieve the events from the database based on the month selected by the user. If there are not events for that month, the system will provide a message saying nothing could be found for that month.

The use case for adding an event will mostly consists of five functions in order to perform everything needed. The system will use the database to retrieve and save the event permanently.

Figure 15- Activity Diagram for adding an event



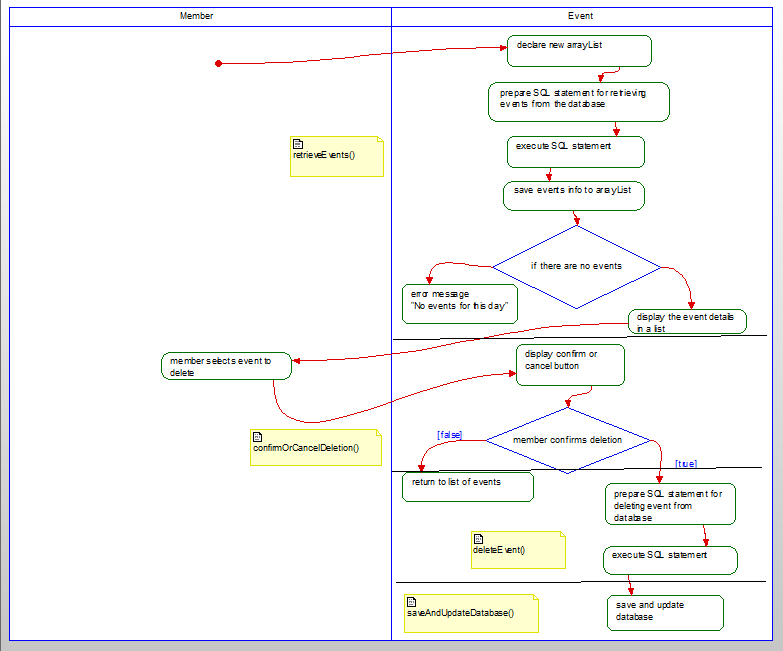
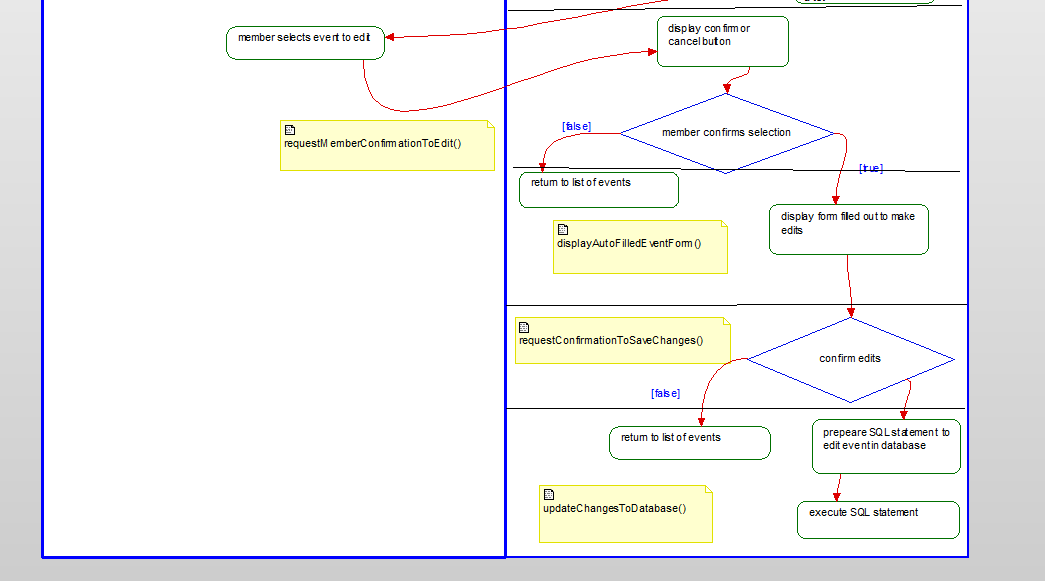
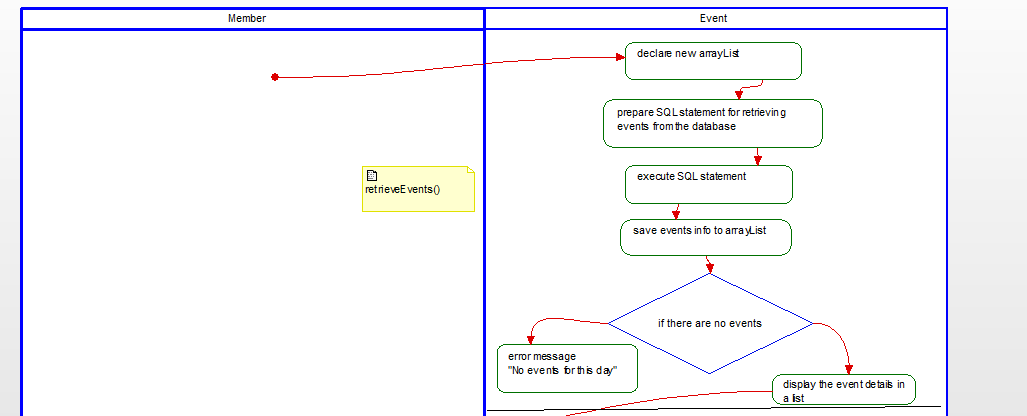
The use case for deleting an event will consists of four main functions. Within the use case, the system will check if there are no events in the system and confirmation from the manager or member.

Figure 16 - Activity Diagram for deleting an event

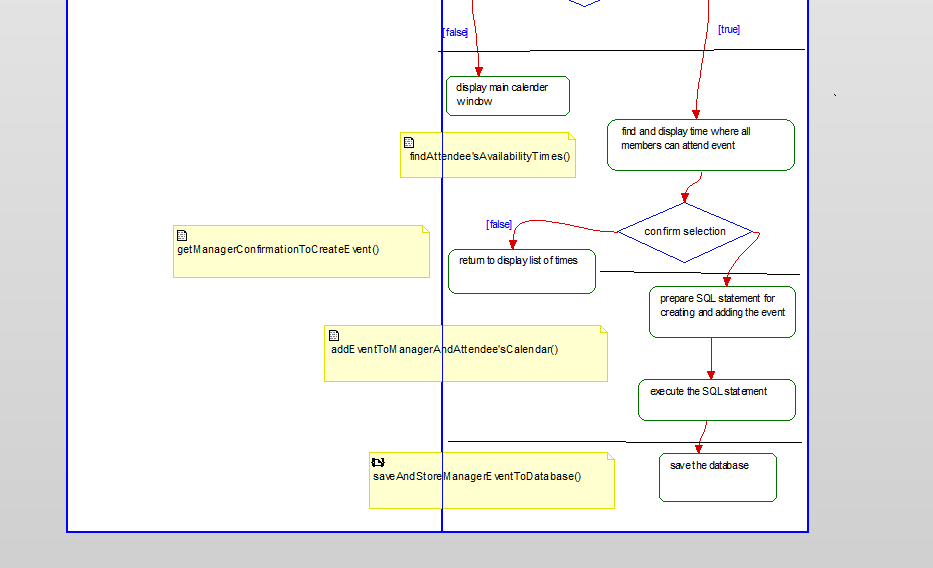
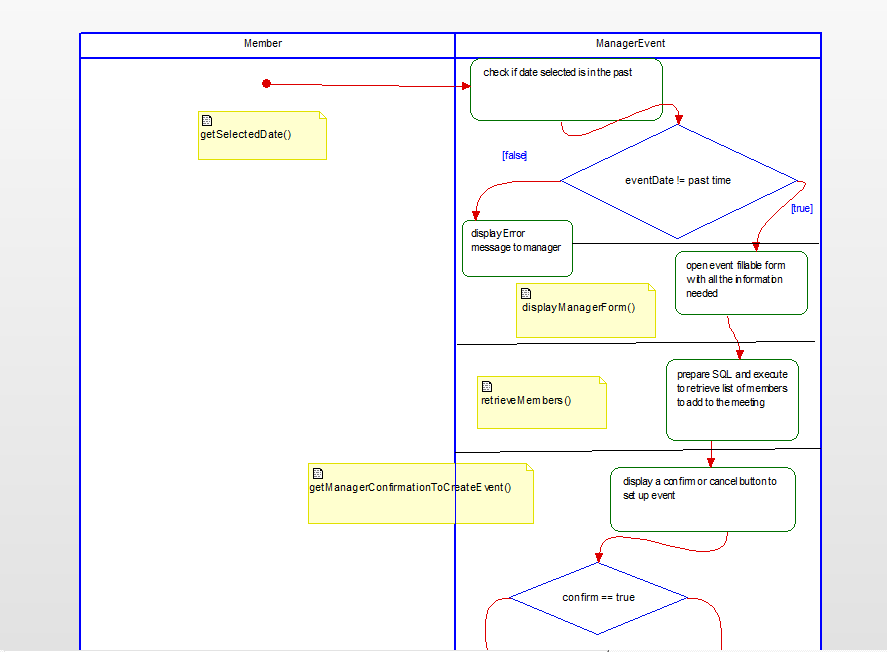
The use case of editing an event will consist of five primary functions. The system will retrieve the events and allow the user to change any details of the event selected. The system will save those changes to the database.

Figure 17 - Activity Diagram for editing an event



The coordinate use case will allow the manager to select a date to hold a meeting or event. They will have to select the attendees for the event in order for the system to find the best time for everyone to attend. The manager will then be able to select the desired time and the system will save it to all attendee’s calendars.

Figure 18 - Activity Diagram for coordinating a meeting from a manager



# State Diagrams

Figure 19 State diagram for signing in and adding an event

The system will be in various states and transitions in order to login into the system and determine the appropriate menu for the manager or member. In regard to adding an event, the system will be validating and checking for various things in order to prevent conflicting scheduling with other events.

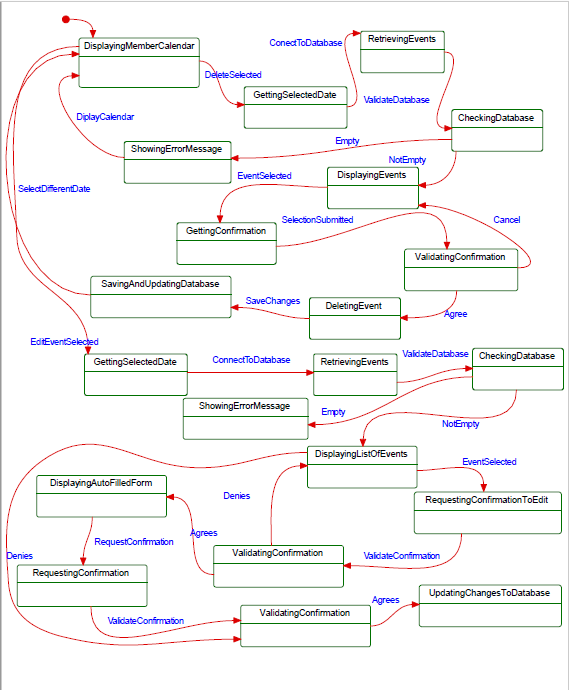


Figure 20 - State Diagram for deleting and editing an event

The main states of the system for deleting an event involve retrieving the event, checking the database for events, along with the actual deletion of the event. For editing an event, the system will be displaying an auto filled form to the manager or member in order to make the changes desired to the event. Lastly, it will update those changes to the database.

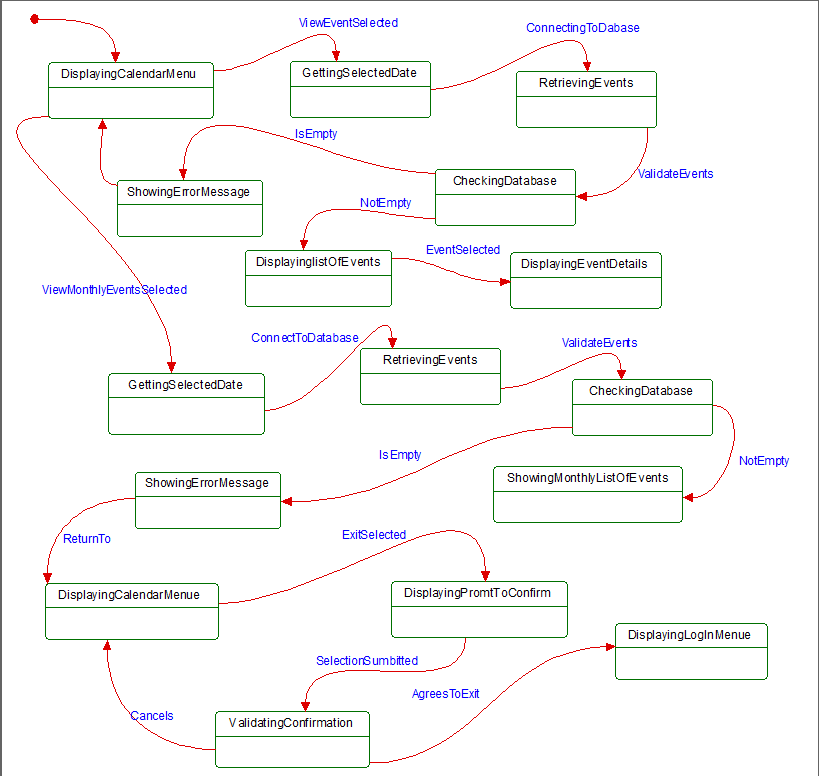
The state diagram of the system for viewing events will involve a lot of retrieving from the database in order to list all the events to the manager or member. There will be the ability for the user to click on the name of an event in the list in order to show all the details associated with that event. When signing out of the system the sign in menu will be shown afterwards.

Figure 21- State Diagram for viewing an event, viewing monthly events, and for signing out of the system

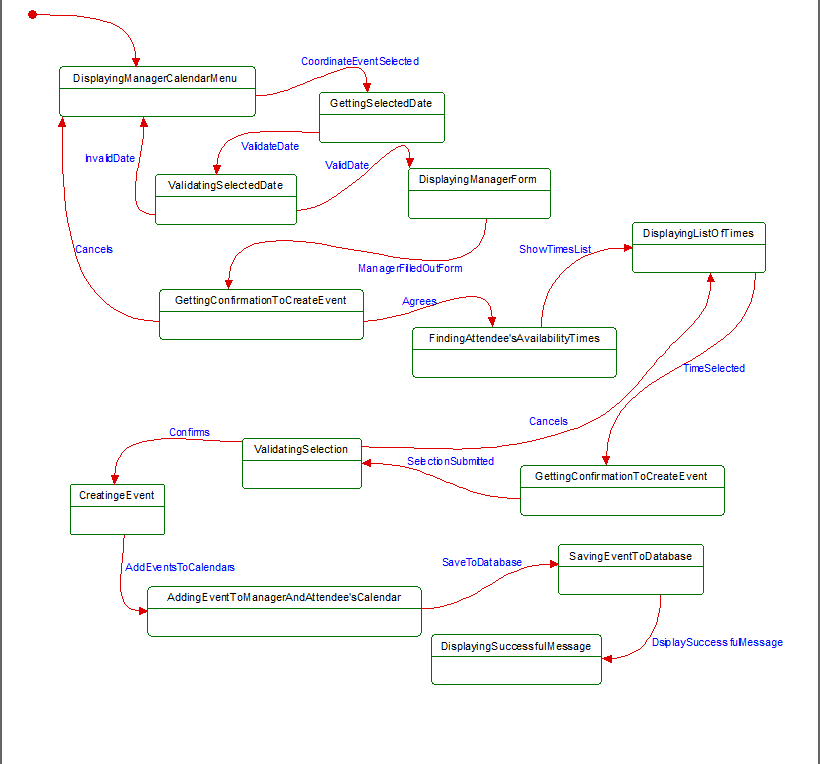


Figure 22- State diagram for coordinating an event

The state diagram for coordinating an event will include many states and transitions, one of the most notable ones being “findAttendeesAvailabilityTimes”. This will allow the system to find times in which everyone selected to be at the meeting can be available for. Once a time is selected, the system will add that event to the manager’s calendar system, but also to all the attendees selected.

# Database Design

## ER Diagram



Figure - ER Diagram for database

This is the ER diagram that is used to design the database. The member has the basic attributes including the username, which will always be unique for each person. The userName and password attribute will be used to sign into the system. The event entity will have all the details regarding the events scheduled. The managerEvent entity will serve as a sub-entity of event, with its own attribute, “attendees”, that will contain all the member for the event.

## Table Schema

CREATE TABLE IJmember(

memberID INT PRIMARY KEY AUTO\_INCREMENT,

userName VARCHAR(9),

password VARCHAR(8),

firstName VARCHAR(10),

lastName VARCHAR(10),

managerStatus BOOLEAN,

UNIQUE (userName)); //allows to refer to userName as a foreign key in IJevent table

CREATE TABLE IJevent(

eventID INT(8) PRIMARY KEY AUTO\_INCREMENT,

eventName VARCHAR(30),

location VARCHAR(30),

startTime DATETIME,

endTime DATETIME,

duration INT(4),

description VARCHAR(70),

DAY VARCHAR(2),

MONTH VARCHAR(2),

YEAR VARCHAR(4),

userName VARCHAR(9),

CONSTRAINT userName\_FK FOREIGN KEY (userName) REFERENCES ijmember(userName));

CREATE TABLE IJmanagerEvent(

eventID INT(8) PRIMARY KEY AUTO\_INCREMENT,

eventName VARCHAR(30),

location VARCHAR(30),

startTime DATETIME,

endTime DATETIME,

duration INT(4),

description VARCHAR(70),

DAY VARCHAR(2),

MONTH VARCHAR(2),

YEAR VARCHAR(4),

userName VARCHAR(9),

attendees VARCHAR(70),

CONSTRAINT userName\_FK\_M FOREIGN KEY (userName) REFERENCES ijmember(userName));

This is what is used to structure the organization of the data in the database for the system to use and manage. In order for an event to be created a manager or member must already be in the database.

IJmember

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| memberID | userName | password | firstName | lastName | managerStatus |

IJevent

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| eventID | eventName | location | userName | startTime | endTime | duration | description | day | month | year |

IJmanagerevent

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| eventID | eventName | location | userName | startTime | endTime | duration | description | day | month | year |

This is another way to visualize the data stored for each table and how certain attributes are being used in other tables to identify which event belongs to which member.

# Conclusion

For this project, in our team we made sure to carefully plan out the design of the system. We started with the database. Using the desired system requirements, we came up with the proper schema that had the appropriate structure and organization of the data. We took into account the functionalities wanted, especially the coordination capability for the manager. It was important that we carefully planned the coordinate capability out given that it would consist of many different methods and would be one of the most important parts for the client. We used our diagrams to implement our code and through that we were able to facilitate the process of making the system work.

We first created sequence diagrams in order to track step by step what each functionality would do, what it would check for and what it would display to the screen. We then made state diagrams to keep track of the different states, transitions, and activities that the system would perform. Lastly, we all did activity diagrams in order to have an outline of what our functions within each use case would do.

We used a simple and minimalist graphical user interface for the client to understand. The GUI is set up in a way and designed to give the client different options to choose from depending on where you are at in the menu. Certain options in the menu will be disabled and grayed out in order to let the client know what options they have given the context. The frontend of the system and communication to the database is all done in the language c# and the database is created using MySQL.

# Data Dictionary

Activity Diagram – describe parallel and conditional activities, use cases, and system functions at a detailed level.

Attribute – a specification that defines a property of an object, element, or file.

Class – blueprint or a set of instructions to build a specific type of object

Entity (in database) – is a thing, person, place, unit, object or any item about which the data should be captured and stored in the form of properties, workflows and tables.

ER Diagram – An entity relationship diagram shows the relationships of entity sets stored in a database

Function – part of a program that has its own name. This name can be used in the program as a command. When a function is called, the commands of which it consists are executed.

Operation – an action that is carried out to accomplish a given task

Sequence Diagram – type of interaction diagram because it describes how and in what order a group of objects works together

Use Case – a description of how users will perform tasks on a system or software

State diagram – describes the behavior or status that the system may be in. It consists of states, transitions, events, and activities the system may perform