|  |
| --- |
| CSC 340 Computer Ethics and Software Engineering |
| Meeting Coordinator |
| Final Report |

|  |
| --- |
| Aidan Barbour, Jeff Davidson, and John Holtzworth  4-24-2020 |

Contents

[I. Introduction 2](#_Toc38640534)

[A. Problem Statement 2](#_Toc38640535)

[B. Proposal 2](#_Toc38640536)

[II. System Description 2](#_Toc38640537)

[III. System Requirements 2](#_Toc38640538)

[A. Functional Requirements 2](#_Toc38640539)

[IV. Use Case Diagram 16](#_Toc38640540)

[V. Class Diagram 17](#_Toc38640541)

[VI. Sequence Diagram 17](#_Toc38640542)

[VII. Activity Diagram 23](#_Toc38640543)

[VIII. State Diagram 29](#_Toc38640544)

[IX. Database Design 30](#_Toc38640545)

[X. Conclusion 30](#_Toc38640546)

[XI. Data Dictionary 31](#_Toc38640547)

# Introduction

## Problem Statement

* + Multiple companies need to allow managers or team leaders to coordinate meetings with their subordinates

## Proposal

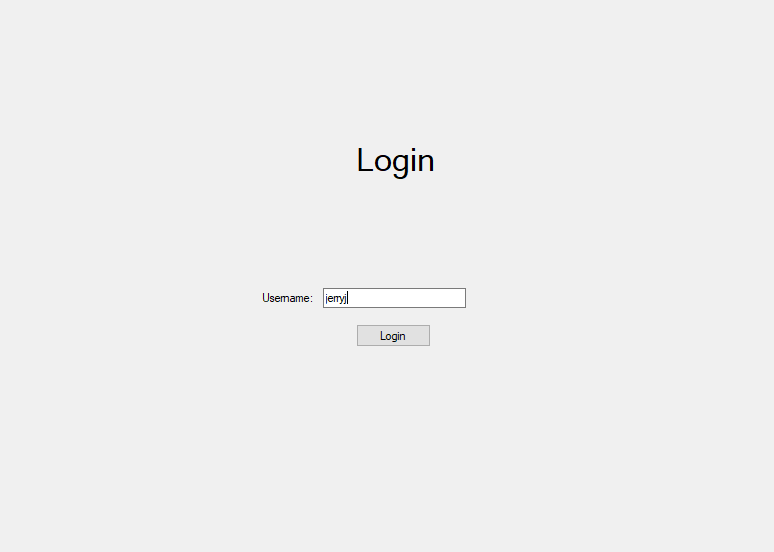
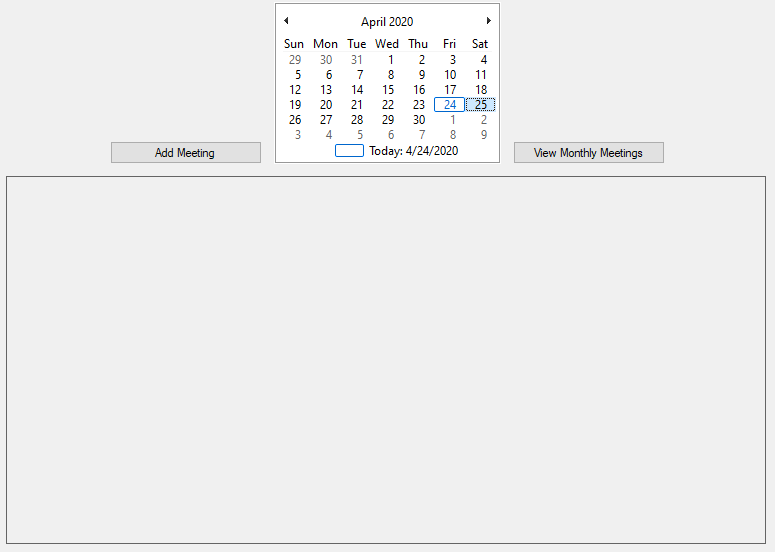
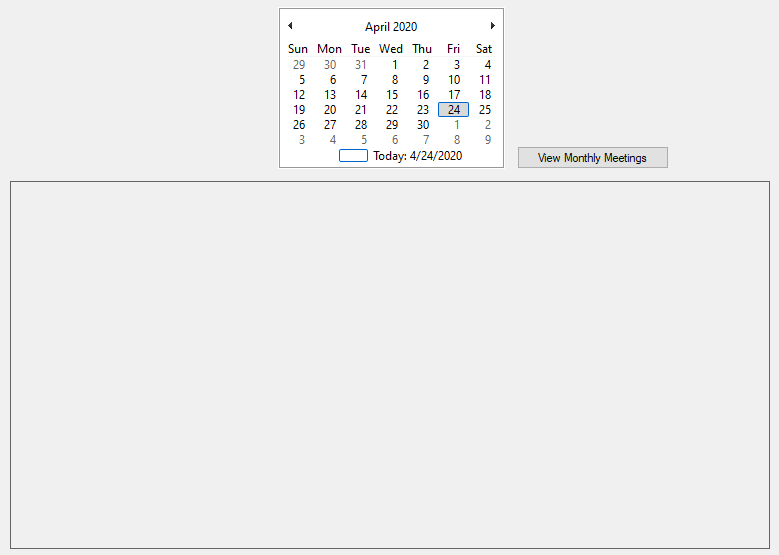
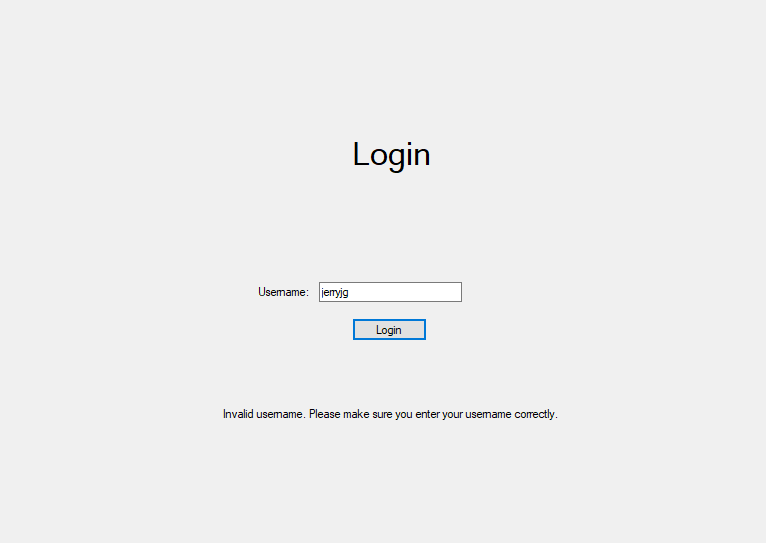
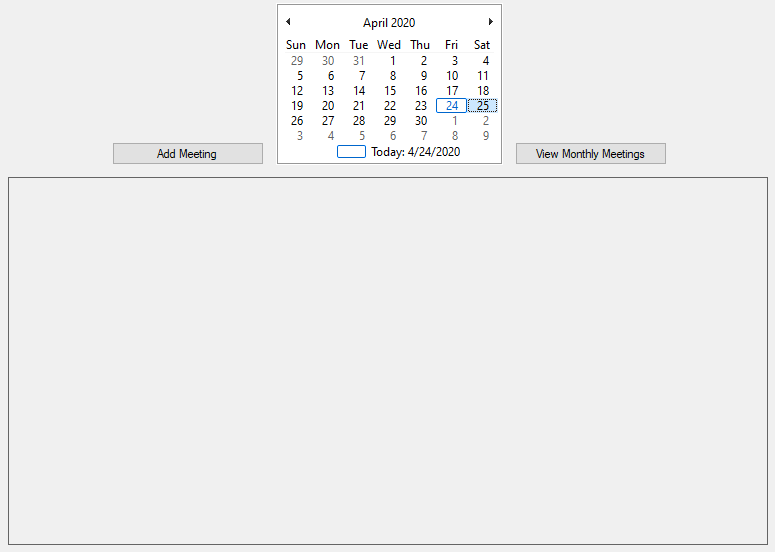
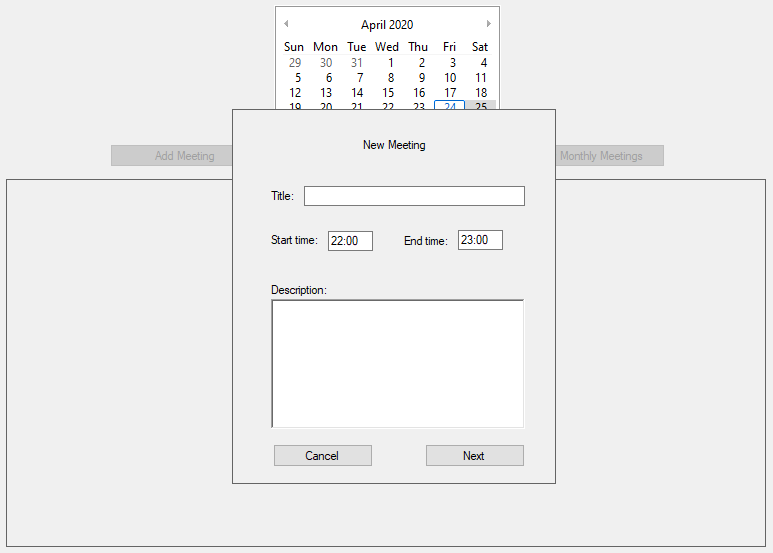
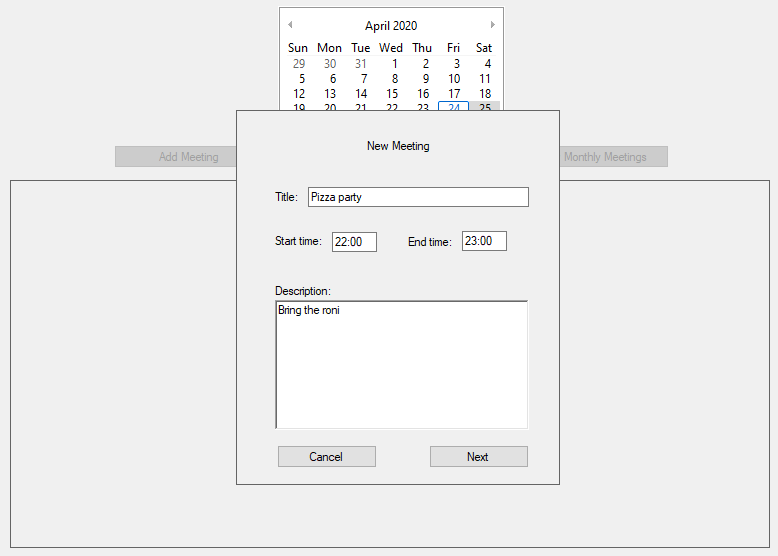
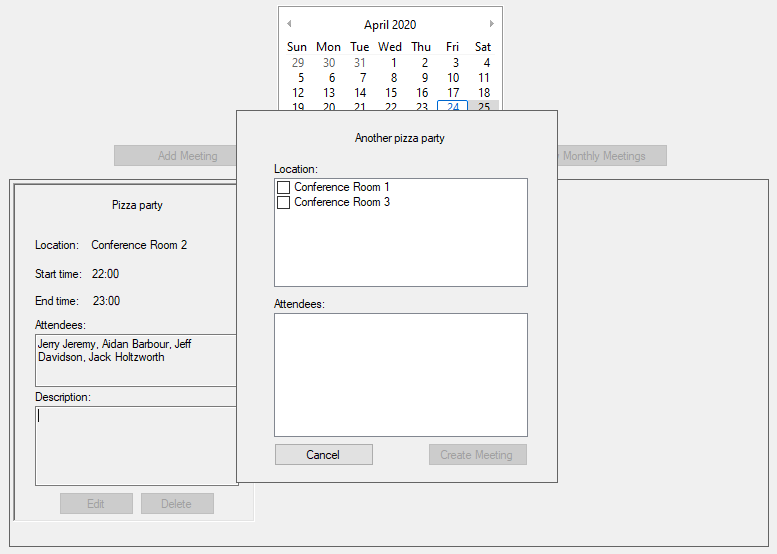
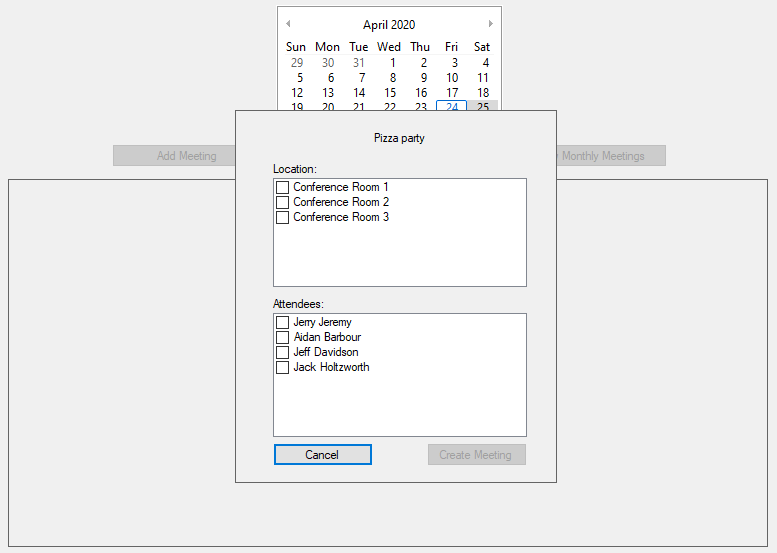
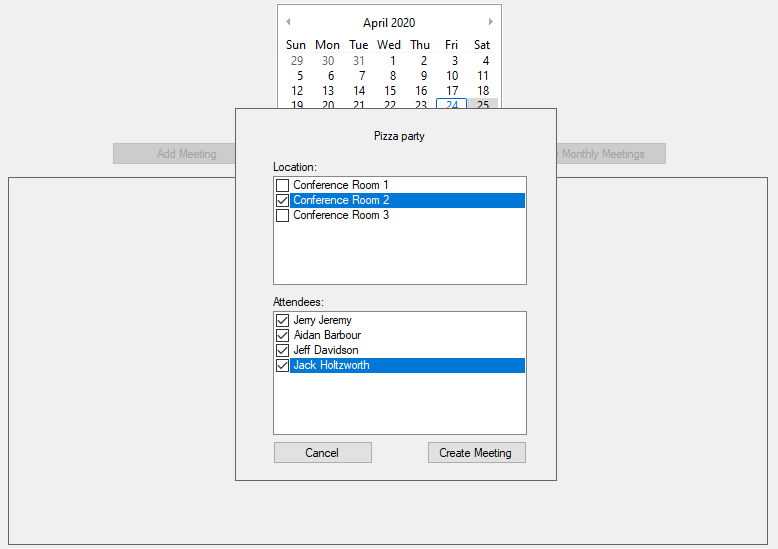
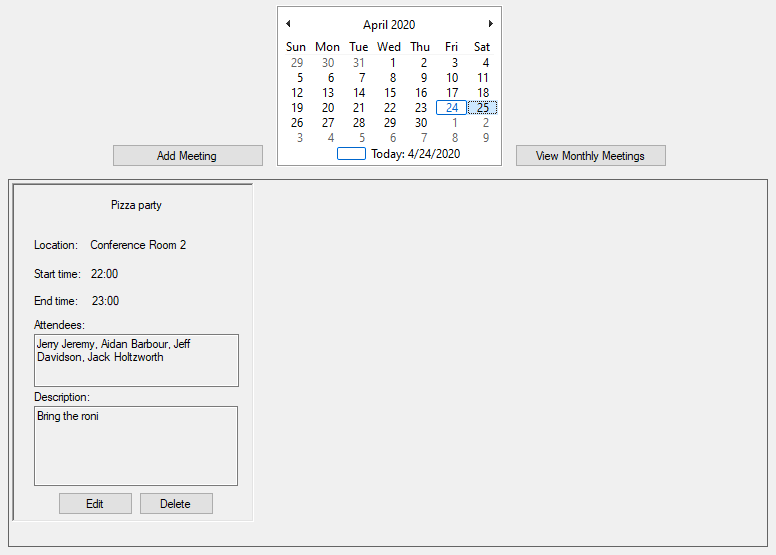
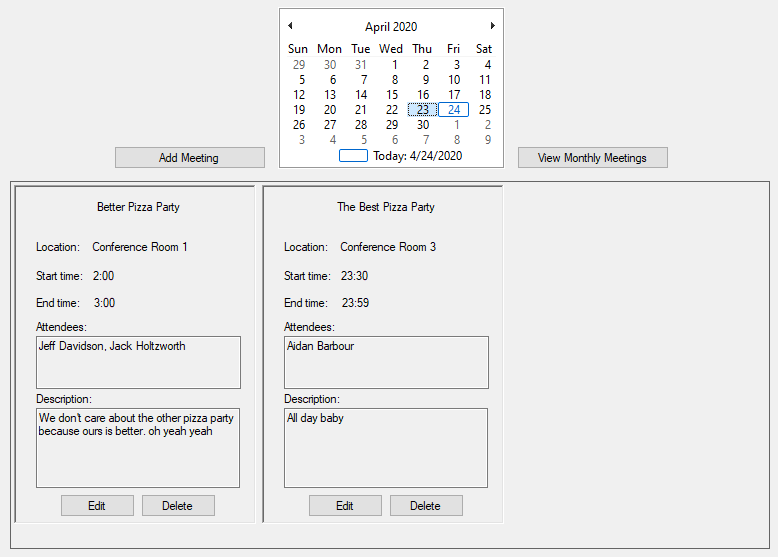
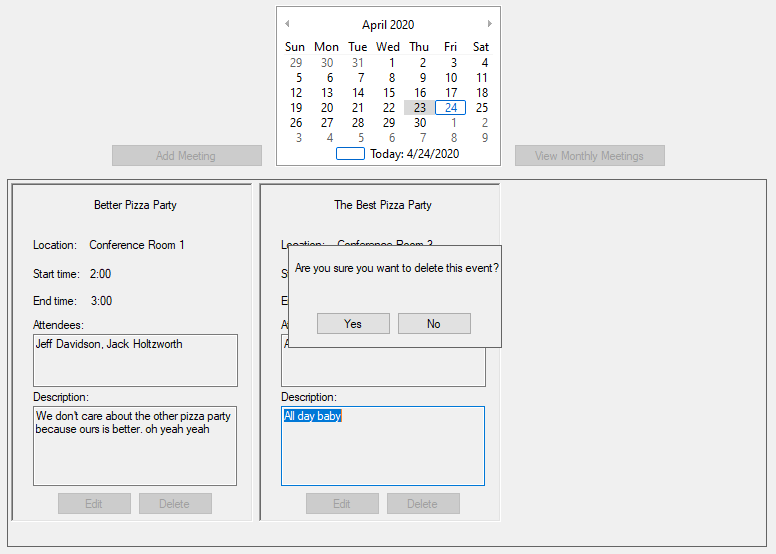
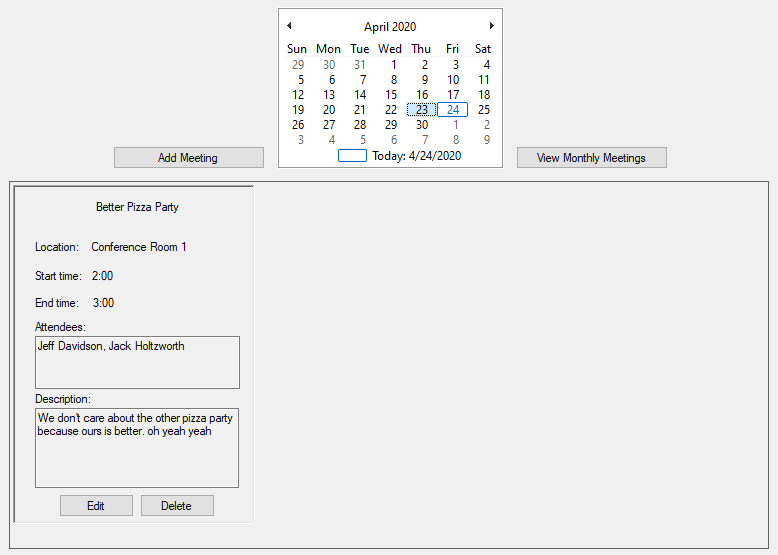
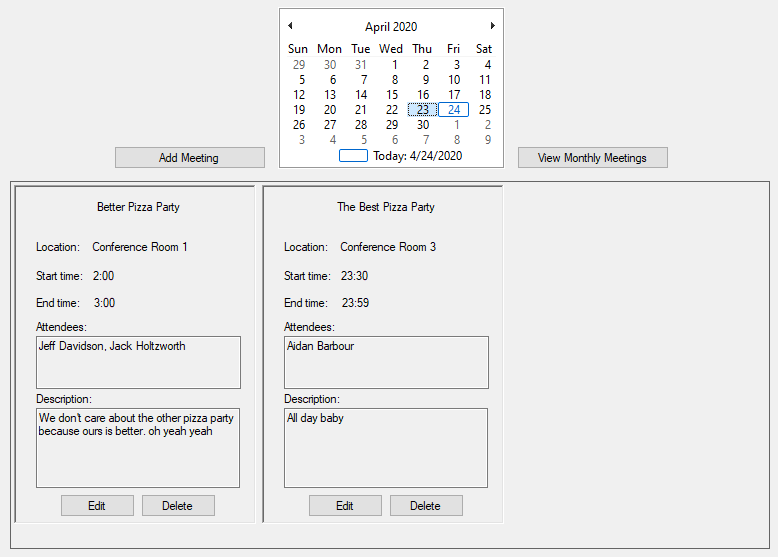
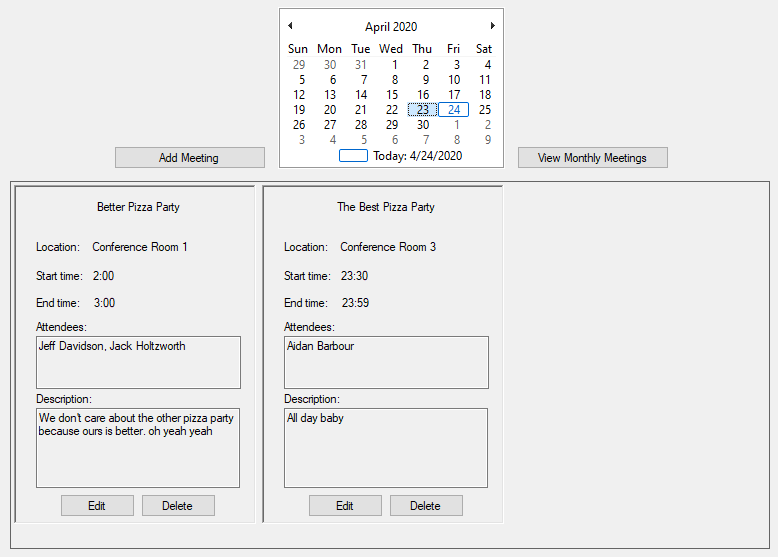
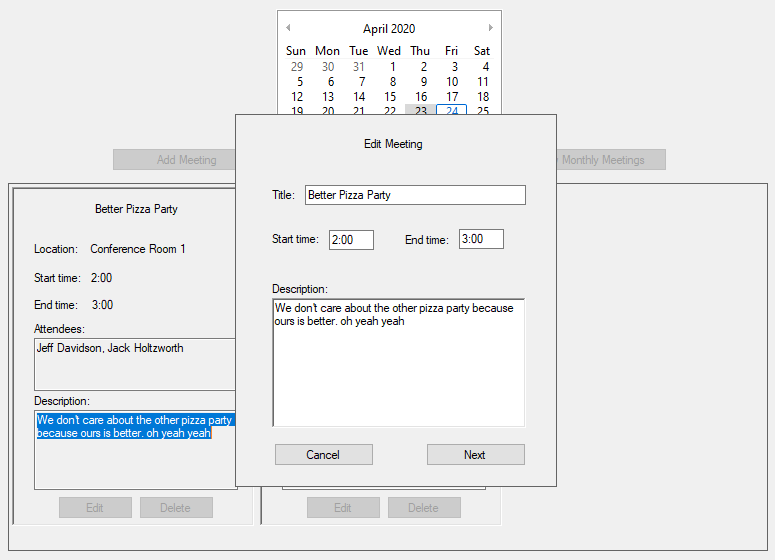
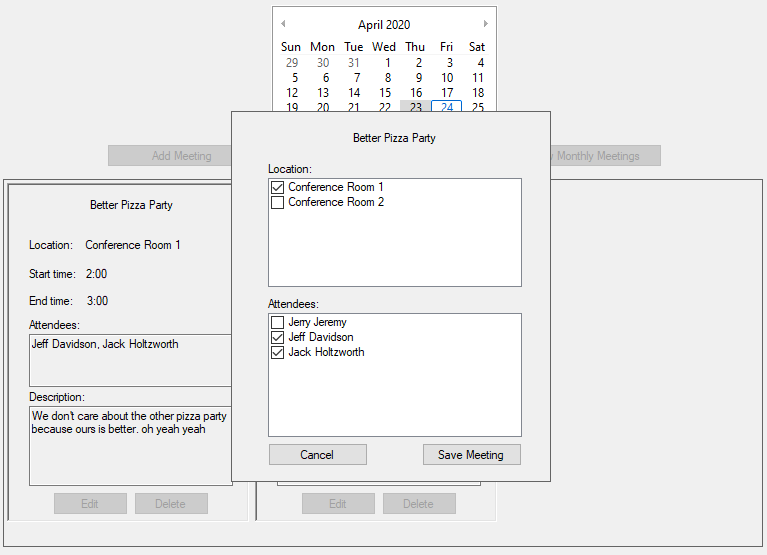
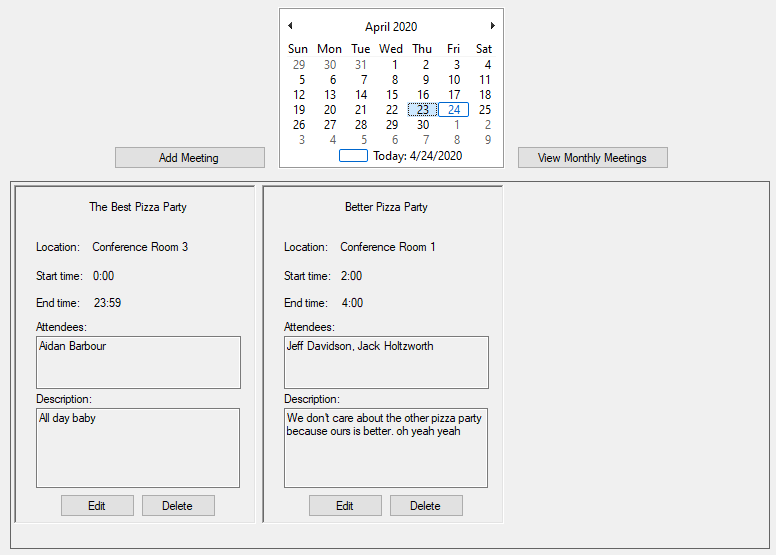
* + We propose a calendar system which will allows for managers to track events and add employees to the events

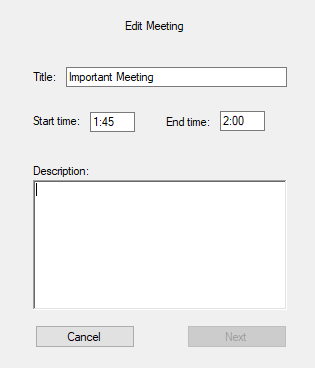
# System Description

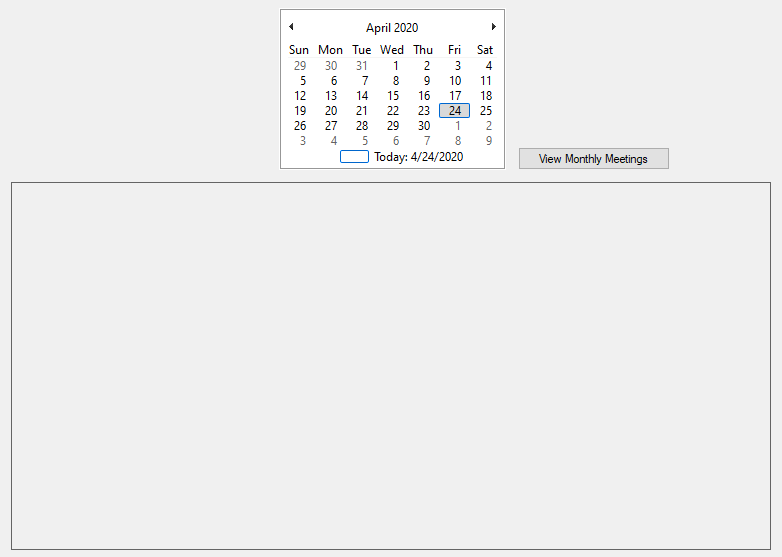
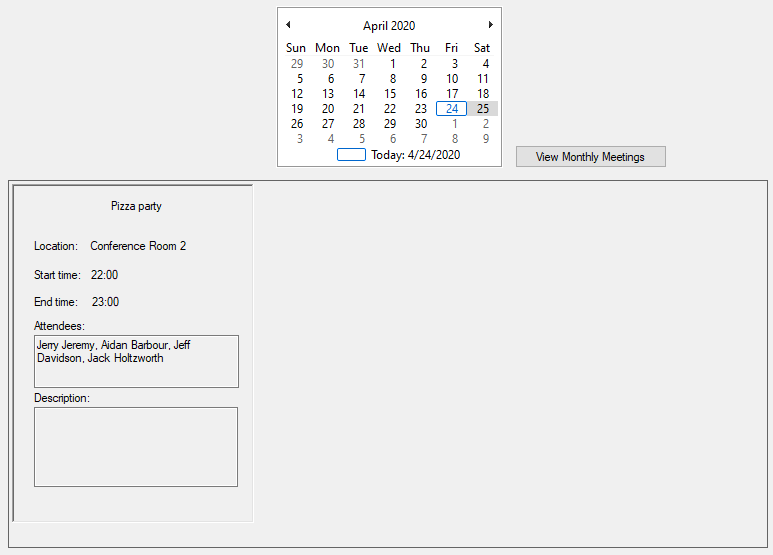
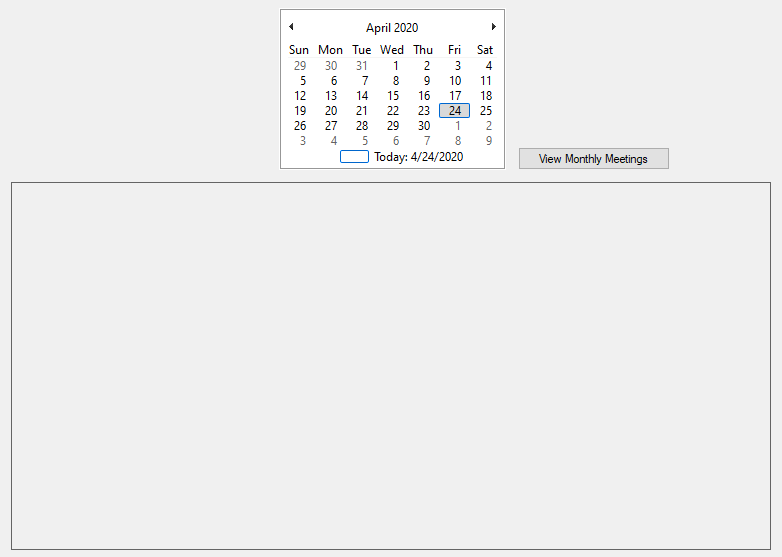
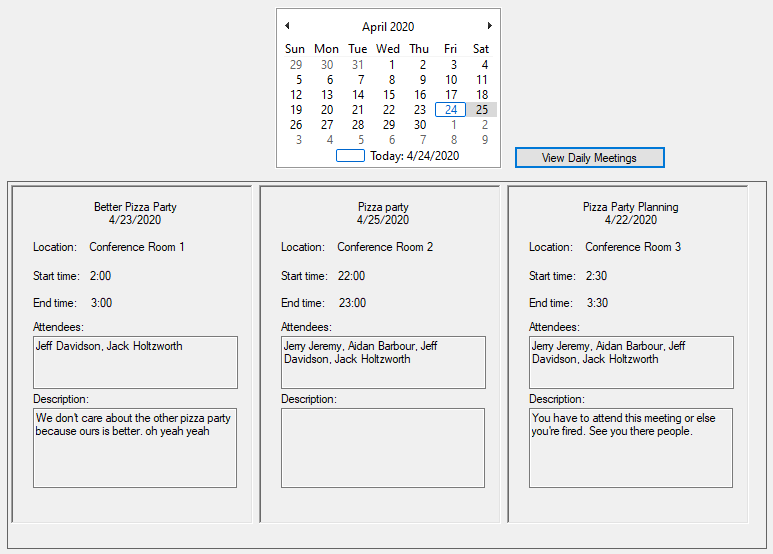
* + The system shall provide the following services to the user:
    - Allow users to login
    - Add a meeting to the calendar of themselves and their employees
      * Check employee availability
      * Check conference room availability
    - Delete a meeting from the calendars of all attendees.
    - Edit a meeting
    - View a daily list of meetings
    - View a monthly-based meeting list.

# System Requirements

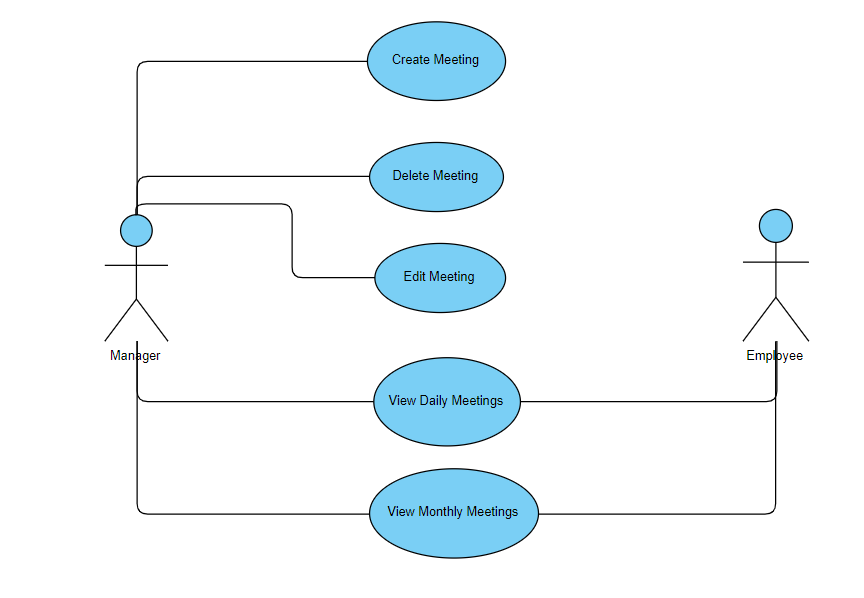
## Functional Requirements

1. The system shall allow users to login using their username.
   1. The user shall type in their username.
   2. 
   3. The user shall hit the login button.
   4. The system shall check if the users entered username is correct.
      1. If the username is correct the system shall login and check the users access level.
         1. If the user is a manager, they will be able to add meetings.
            1. 
         2. If the user is an employee, they will only be able to view meetings.
            1. 
      2. If the username is incorrect the system shall display an invalid username message.
         1. 
2. The system shall allow the manager to create meetings on selected days.
   1. The manager shall select a date from the monthly calendar to add the new meeting.
      1. 
   2. The manager shall press the “Add Meeting” button.
   3. The system shall display a form to allow a manager to enter the information of the new meeting, including title, description, starting time, ending time.
      1. 
   4. The manager shall enter the data for the meeting and press the “Next” button.
      1. 
   5. The system shall check to see which rooms and who is available during the selected time of the meeting and display a new form for location and attendees.
      1. If there is a time conflict, the system shall display only which rooms and who is available at the time selected.
         1. 
      2. If there is no time conflict, the system shall display everyone.
         1. 
   6. The manager shall select who is attending and the location, then select the “Create Meeting” button.
      1. 
   7. The system shall display the new meeting and its contents on the selected day.
      1. 
3. The system shall allow a manager to delete meetings.
   1. The manager shall select a date from the monthly calendar to delete the meeting.
   2. The system shall display meetings of the currently selected day.
      1. 
   3. The manager shall select the “Delete” button on the desired meeting to delete it.
   4. The system shall display a conformation box.
      1. 
      2. If the “yes” button is pressed the system shall delete the meeting from the database and return to display meetings.
         1. 
      3. If the “no” button is pressed the system shall do nothing and return to display meetings.
         1. 
4. The system shall allow a manager to edit meetings.
   1. The manager shall select a date from the monthly calendar to edit a meeting.
      1. 
   2. The manager shall press the “Edit” button on the meeting desired to edit it.
   3. The system shall display a form to allow a user to edit the information of the meeting, including title, description, starting time, ending time, location, and attendees.
      1. 
   4. The user shall edit the data for the meeting and press the “Next” button.
      1. 
   5. The manager shall edit the location and attendees and press the “Save Meeting” button.
   6. The system shall check the conflict of the edited meeting with existing meetings.
   7. The system shall save the meeting to the database and display the meeting on the calendar.
      1. If there is no conflict, the system shall save the meeting to the database, and display the meeting title on the calendar.
         1. 
      2. If there is any conflict, the system shall not allow the manager to edit the meeting.

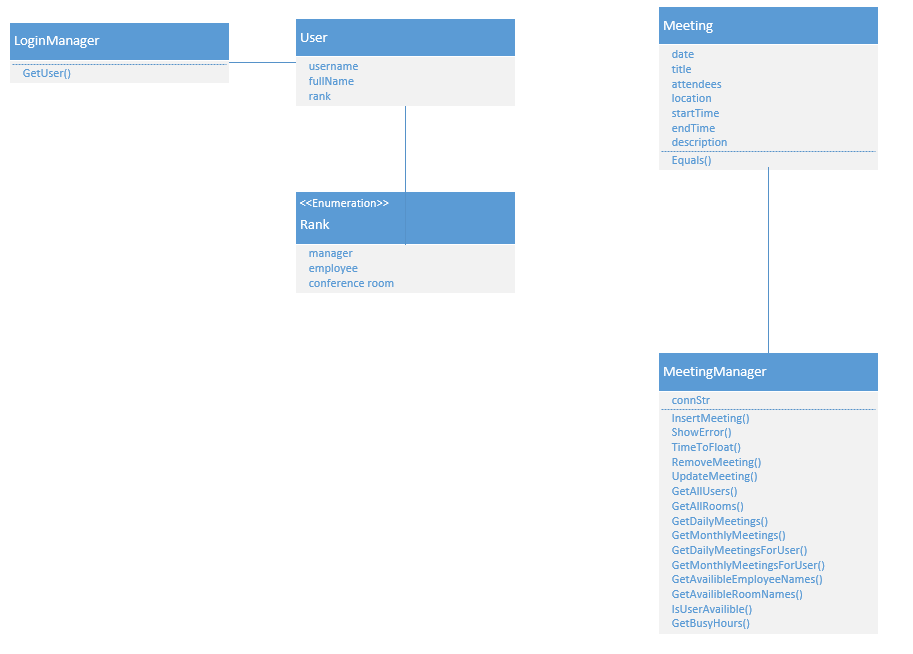


1. The system shall allow users to view meetings on selected days.
   1. The user shall select a date to view the meetings in that day.
      1. 
   2. The system shall display meetings that are scheduled on selected days.
      1. 
2. The system shall allow users to view all meetings on a selected month.
   1. The manager shall select a date from the monthly calendar to select the month associated with it.
      1. 
   2. The user shall press the “View Monthly Meetings” button to see all meetings in the selected month.
      1. 
   3. The user shall press the “View Daily Meetings” button to return to all meetings shown on a selected date.

# Use Case Diagram



# Class Diagram



# Sequence Diagram

1. Employee Login
   1. A screenshot of a cell phone

      Description automatically generated
2. Add Meeting
   1. A close up of a map

      Description automatically generated
3. Delete Meeting
   1. A close up of a map

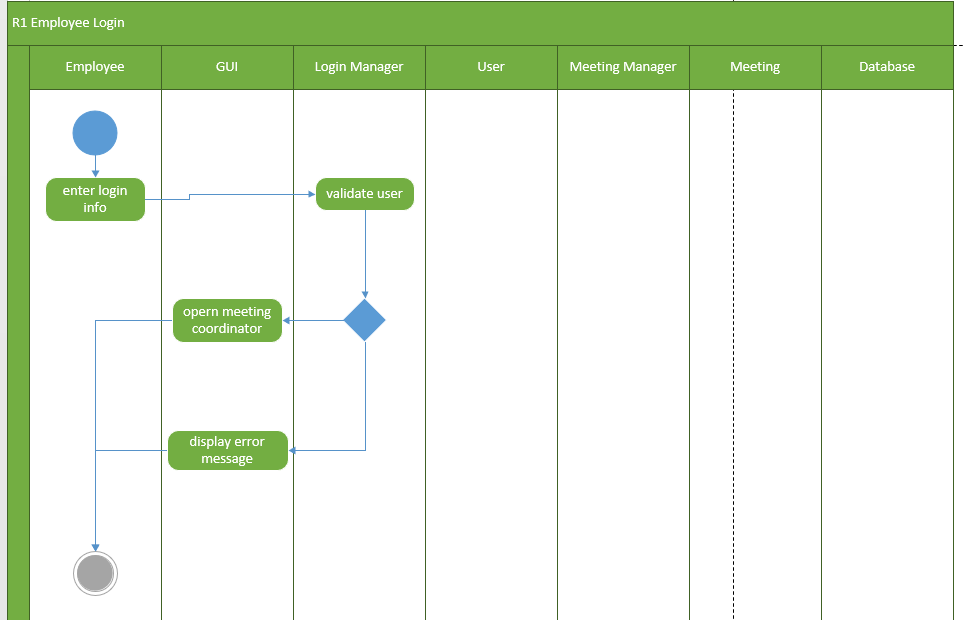
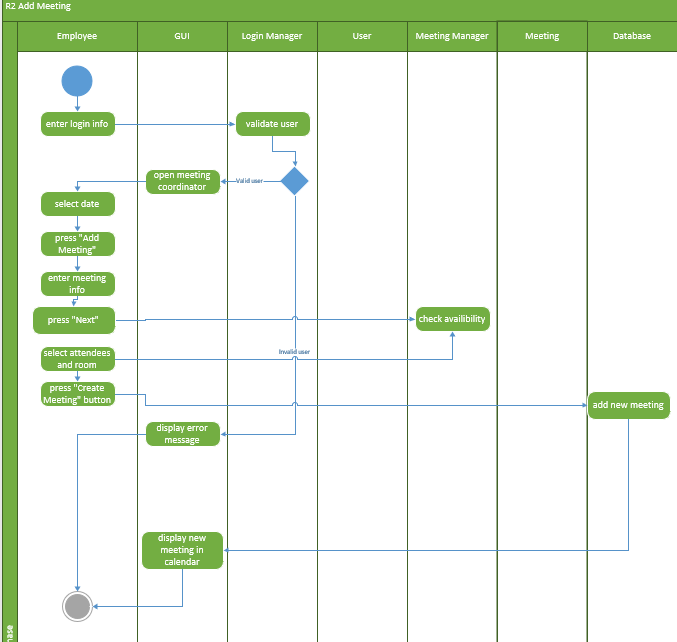
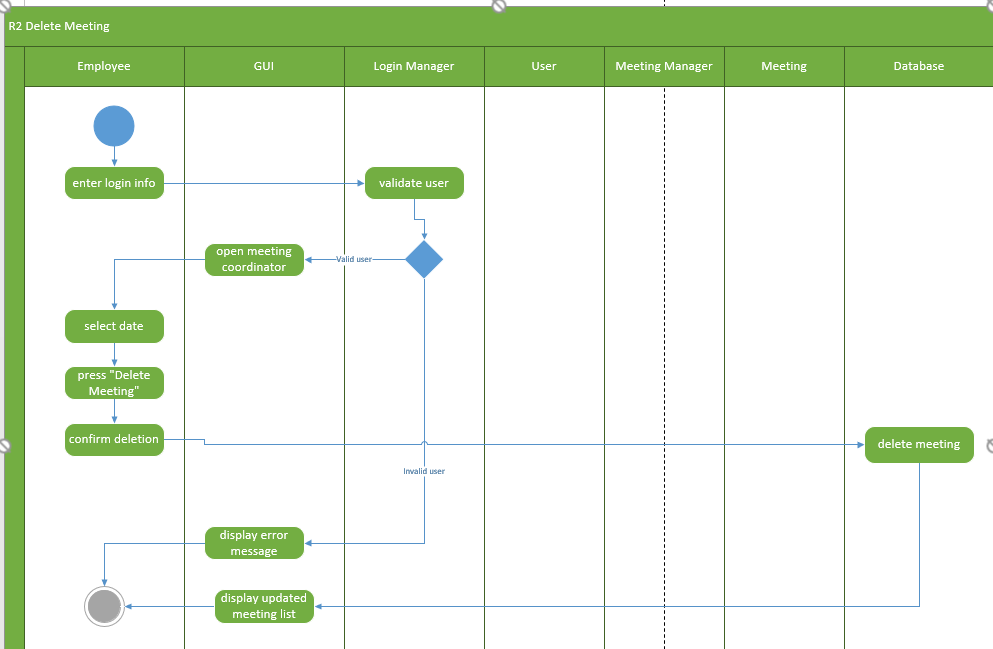
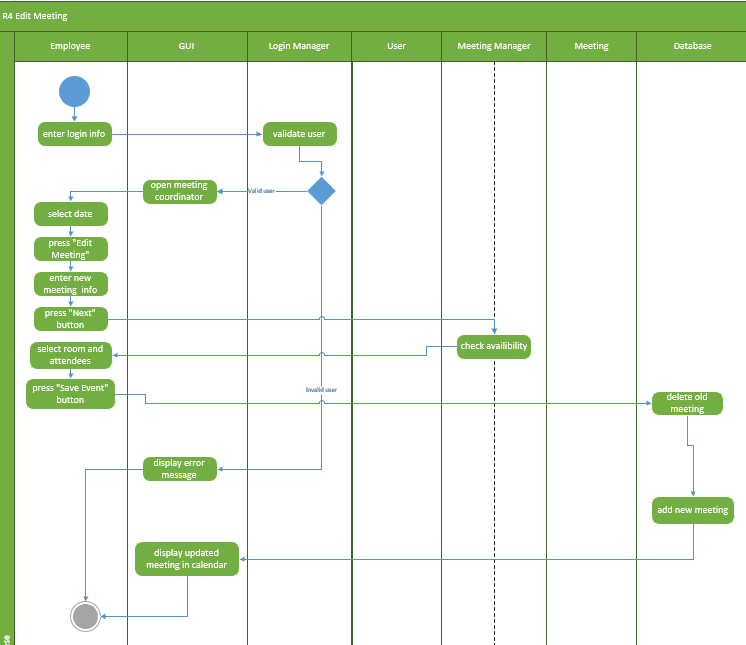
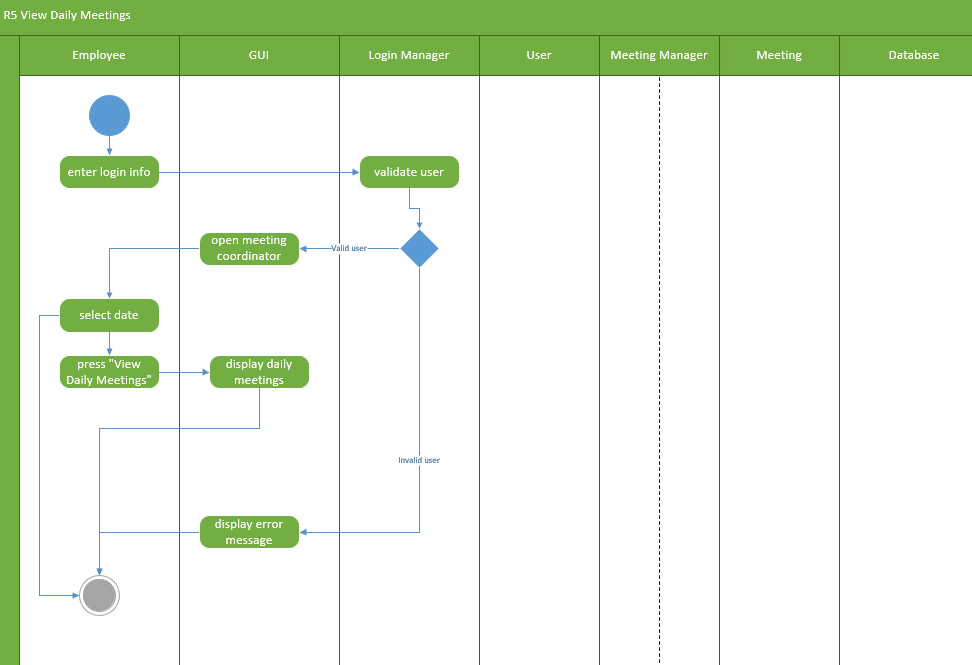
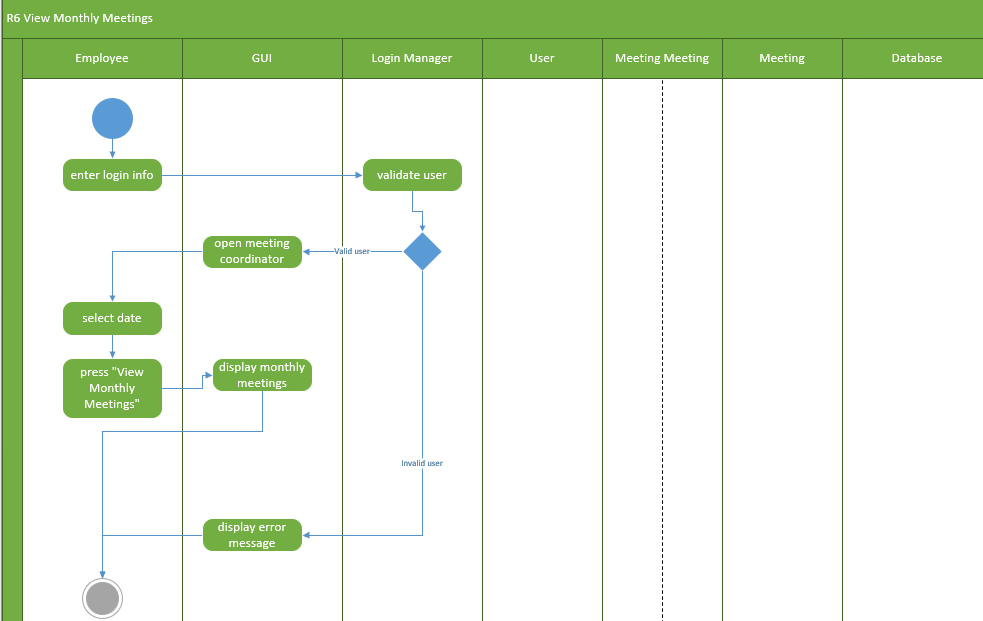
      Description automatically generated
4. Edit Meeting
   1. A screenshot of a cell phone

      Description automatically generated
5. View Daily Meetings
   1. A screenshot of a cell phone

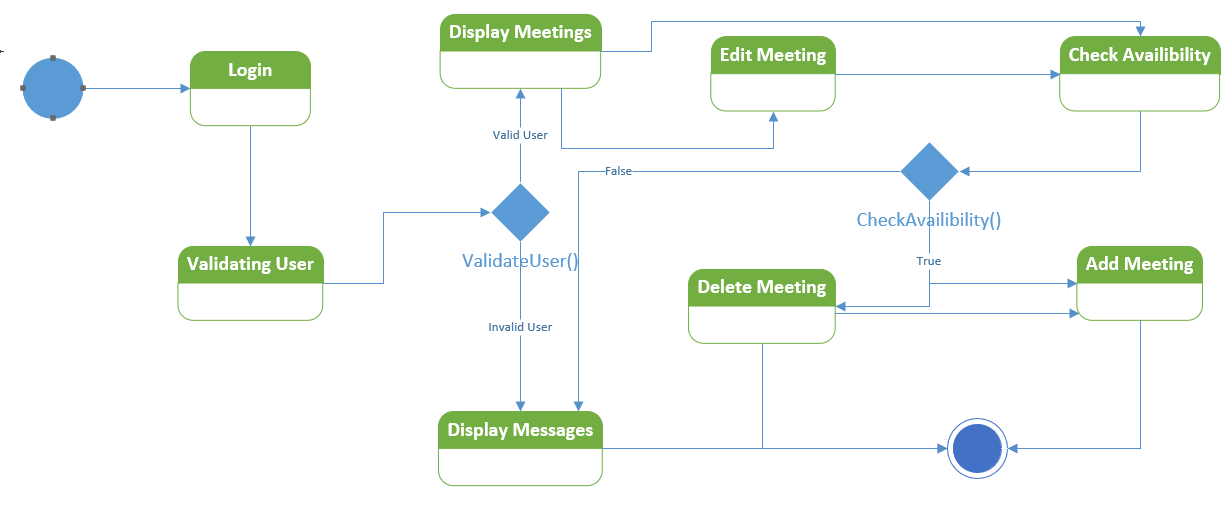
      Description automatically generated
6. View Monthly Meetings
   1. A close up of a map

      Description automatically generated

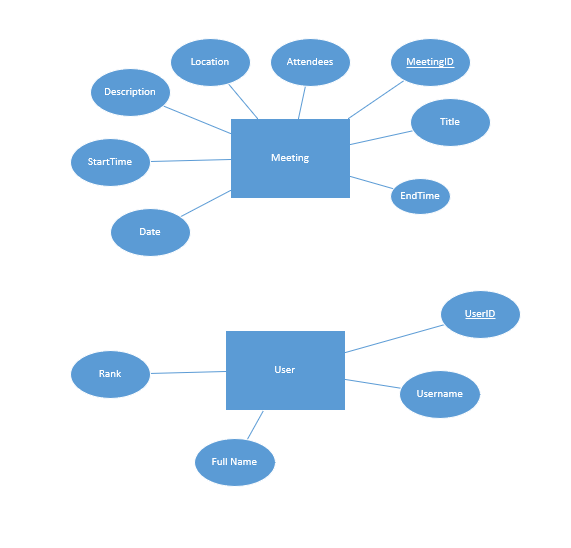
# Activity Diagram

1. Employee Login
   1. 
2. Add Meeting
   1. 
3. Delete Meeting
   1. 
4. Edit Meeting
   1. 
5. View Daily Meetings
   1. 
6. View Monthly Meetings
   1. 

# State Diagram



# Database Design



Meeting

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| meetingID | date | title | startTime | endTime | location | attendees | description |

User

|  |  |  |  |
| --- | --- | --- | --- |
| userID | fullName | username | rank |

# Conclusion

* 1. We have developed a meeting coordination system which implements the required specifications. This will solve the personal and company problems as listed.
  2. We have included a functional requirements section which shows a step-by-step process of how each functional requirement of the system works.
  3. We have included a use case diagram which shows the relationship between the users and the functions that different users can perform.
  4. We have included a class diagram which describes the basic structure and operations of the meeting and event classes.
  5. We have included sequence diagrams for each functional requirement which show the chronological order of how the system processes information and fulfills a certain duty.
  6. We have included activity diagrams which describes the different user’s process of completing a use case.
  7. We have included a state diagram for the entire coordination system which displays how the system can move from one state to another.
  8. We have included an ER Diagram which shows the basic structure of the database. Below the diagram is a small table showing how the information will be saved in the different tables.

# Data Dictionary

ajjmeetings Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Attribute Name | Data Type | Valid Values | Null Allowed | Description |
| meetingID | INTEGER | 1-Max\_Value () | No | AUTO\_INCREMENTS |
| date | DATE | YYYY-MM-DD | No | Date, Set by calendar |
| title | VARCHAR | Any | Yes | Title of event |
| location | VARCHAR | Confrence Room 1-3 | No | Conference room used for the event |
| startTime | VARCHAR | 00:00 – 23:59 | No | Start time of given event |
| endTime | VARCHAR | 00:00 – 23:59 | No | End time of given event |
| attendees | VARCHAR | Any | No | Attendees list of the given event |
| description | VARCHAR | Any | Yes | Description of given event. |

1. ajjusers Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| userID | INTEGER | 1-Max\_Value () | No | AUTO\_INCREMENTS |
| fullName | VARCHAR | Any | No | Full name of employee or manager. |
| Username | VARCHAR | Any | No | Username of employee or manager |
| Rank | INTEGER | 0-2 | No | Assesses level of this user. 0 is manager, 1 is employee, 2 is conference room. |