By: Addison Bradley, Seth Dean, and Zach Robertson

12/14/2017

CSC 340

Fall 2017

meeting coordinator

Team Project

Table of Contents

[1. Introduction 2](#_Toc500438307)

[2. System Description 2](#_Toc500438308)

[3. System Requirements 2](#_Toc500438309)

[I. Functional Requirements 2](#_Toc500438310)

[II. Non-Functional Requirements 9](#_Toc500438311)

[4. Use Case Diagram 10](#_Toc500438312)

[5. Class Diagram 11](#_Toc500438313)

[6. Sequence Diagram 12](#_Toc500438314)

[I. Create Event 12](#_Toc500438315)

[II. Delete Event 13](#_Toc500438316)

[III. Edit Event 14](#_Toc500438317)

[IV. List Month 15](#_Toc500438318)

[V. Room Schedule 16](#_Toc500438319)

[VI. User Schedule 17](#_Toc500438320)

[VII. View Event 18](#_Toc500438321)

[7. State Diagram 19](#_Toc500438322)

[8. Activity Diagram 20](#_Toc500438323)

[9. Database Diagram 21](#_Toc500438324)

[I. ER Schema 21](#_Toc500438325)

[II. Table Schema 22](#_Toc500438326)

[10. Conclusion 22](#_Toc500438327)

[11. Data Dictionary 22](#_Toc500438328)

# Introduction

In this project, we were tasked with creating a program to take multiple peoples calendar information as well as conference room availability and create a report over the system and the services that it provides. To accomplish this, we will design a system that allows the Coordinator to take individual Coordinators schedules, consolidate them into one system, display them as well as the room availability and create the meeting that they find works the best with everybody.

# System Description

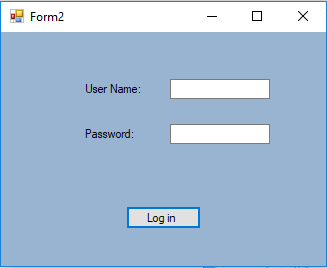
Create a system that allows a Coordinator (meeting coordinator) to take the personal schedules of multiple workers and determine the best time and day for a meeting between them to happen. The attendants would send their own personal calendars to the Coordinator to consolidate each personal schedule. This system will allow the Coordinator to cross-reference each personal schedule to determine the best date and time to set up a meeting. The system will also account for conference room availability and will cross-reference with the meeting times that are possible to find the easiest possible schedule.

# System Requirements

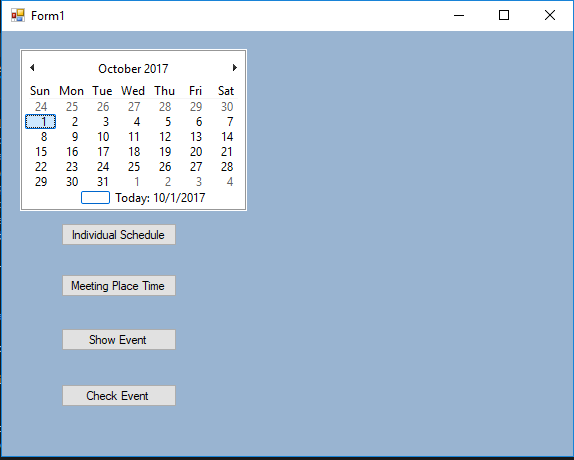
## Functional Requirements

This section will illustrate functional requirement of the system with scenario analysis

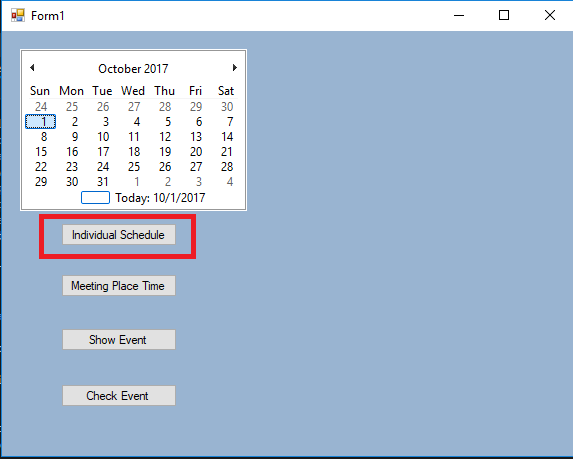
NFR.1 Prompt coordinator for log in information in which they can enter and access their own calendar



R1. System will allow the Coordinator to consolidate all workers schedules into one with given information

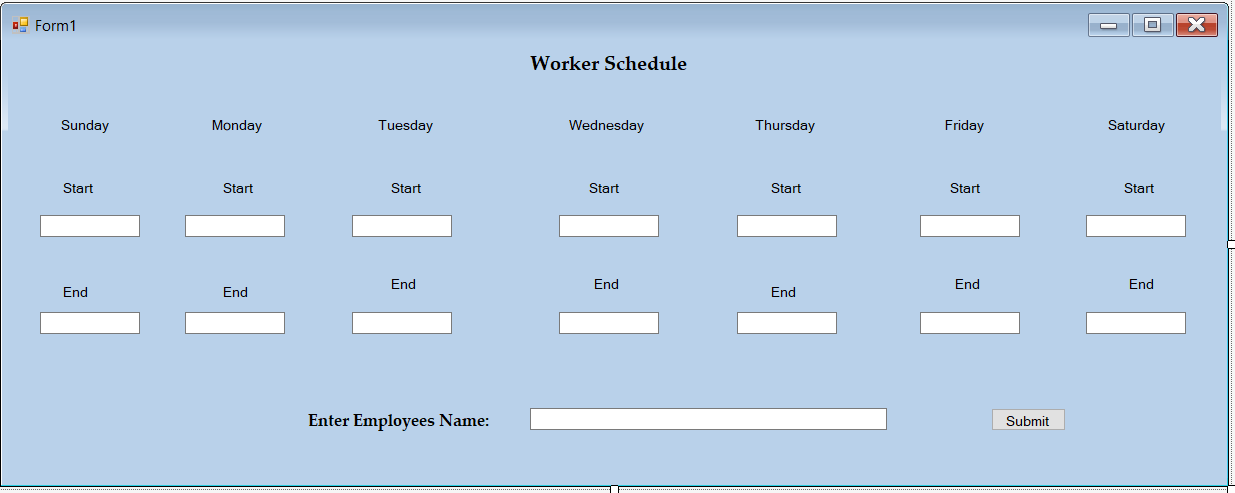


R1.1 The system will allow the Coordinator to create individual schedules that can be used to check their availability for meeting times.

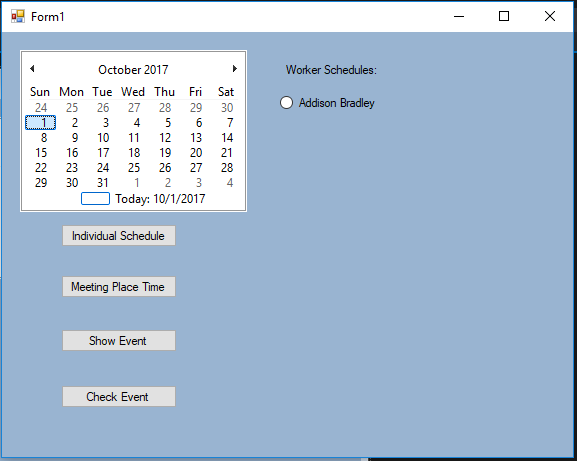


R1.1.1 The Coordinator will put in name of the worker in the “Enter Employee Name” box to keep track of their schedule.

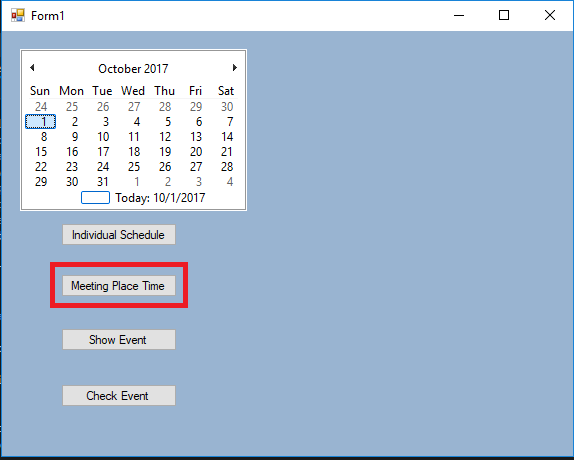
R1.1.2 Coordinator puts in times that the employee works during the week



R1.2 The system will allow the Coordinator to have the option to put in another person’s schedule and display worker’s schedules

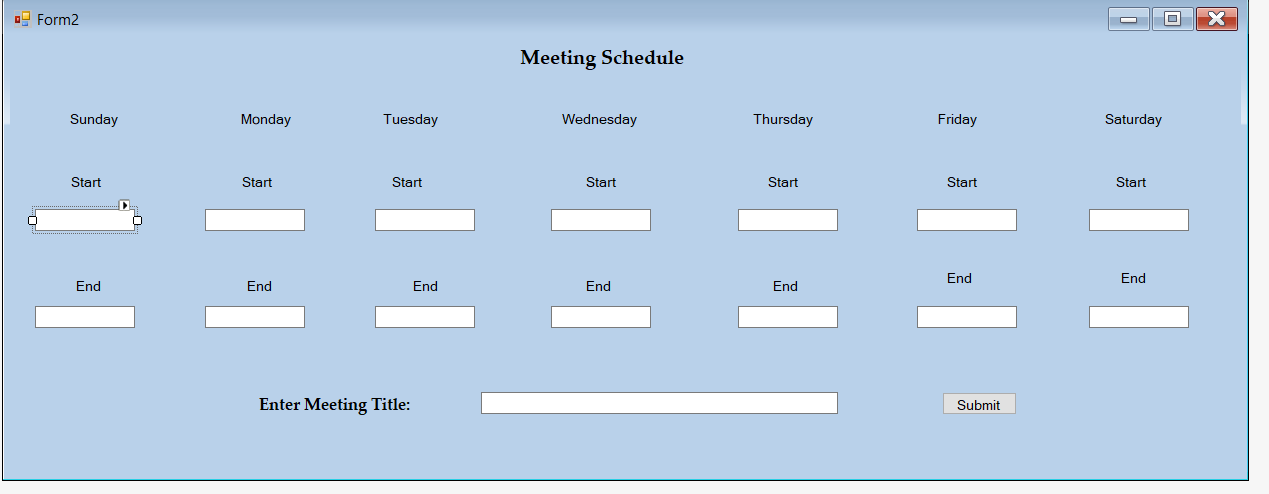


R2. The system will allow the Coordinator to create schedules for the available meeting rooms (refer to how coordinator can create individual schedules)

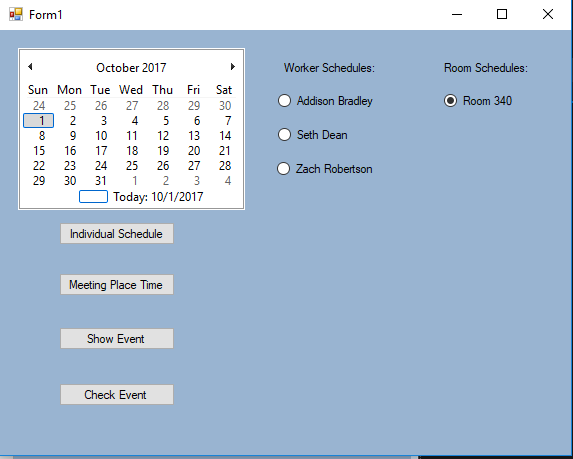


R2.1 The system will allow the Coordinator to put in location of the conference room in “Enter Meeting Title” (building and number)

R2.2 The system will allow the Coordinator to put in the time of availability for that select conference room.



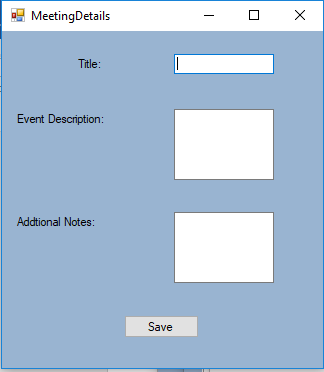
R2.3 The System will allow the Coordinator the option to put in another location and other possible conference room availability.



R3. The system will allow the Coordinator to view the multiple schedules of both the workers and the conference room availability.

R3.1 The system will allow the Coordinator to create a meeting

R3.1.1 The system will create buttons which allow the Coordinator to put in participants, where the meeting will be at, time, date, how long it would be, reminder time, description, and additional notes

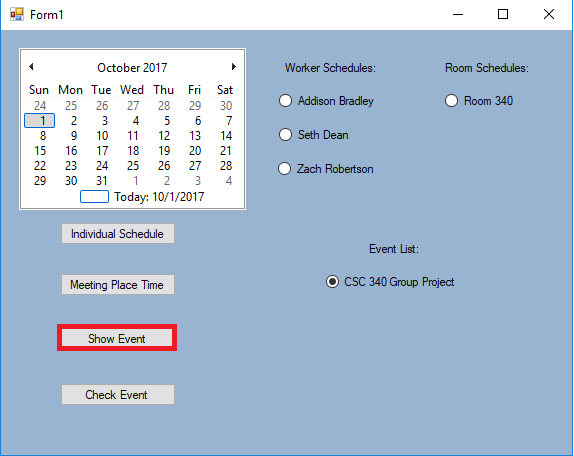


R3.2 The system shall allow the Coordinator to have button that saves the meeting date and time

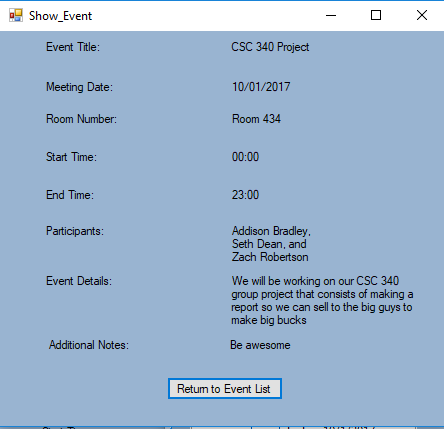
R3.2.1 When Coordinator presses button to save meeting, a list of possible times and locations will appear

R4. The system shall allow the Coordinator to show the event details

R4.1 The system will show a view event button that allows them to access the event details

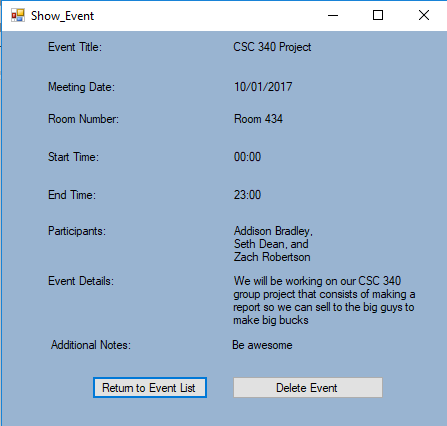


R4.2 The system shall show the meeting title, date meeting is held, room it’s held in, start time, end time, reminder time, participants, details of the meeting and additional notes on the meeting.

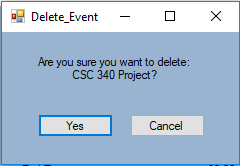


R5. The system shall allow the coordinator to Delete an event after it has been created.

R5.1 When the show event button is clicked, the system will allow the Coordinator to delete the event.

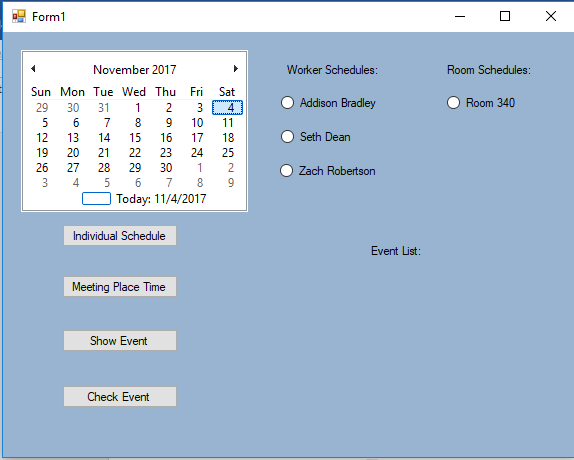


R5.2 The system will bring up a dialog box, that allows the coordinator to choose whether they want to delete the event



R5.2.1 The System will allow the coordinator to choose to cancel, which will allow the system to keep that event in its records and return the show event screen (refer to R4.2/R5.1)

R5.2.2 The system will allow the coordinator to choose the option yes, which will delete the event from the event list



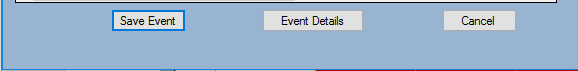
R6. The system will allow the coordinator to edit the event that they have created

R6.1 The system will allow the coordinator to access the Edit Event button through the View Event button (like delete Event).

R6.1.1 Coordinators will be able to access the workers schedule in a list to view available times. (Refer to R3.1.)

R6.1.2 Coordinators can edit the participants, where the meeting will be at, time, date, how long it would be, reminder time, description. (refer to R3.1.1)

R6.2 The system will allow the coordinator to save the event through the save event button.



R6.2.1 A dialog box will appear that asks the user if they want to update the information that they have entered in the boxes.

R6.2.2 Clicking Yes will update the information in the details list

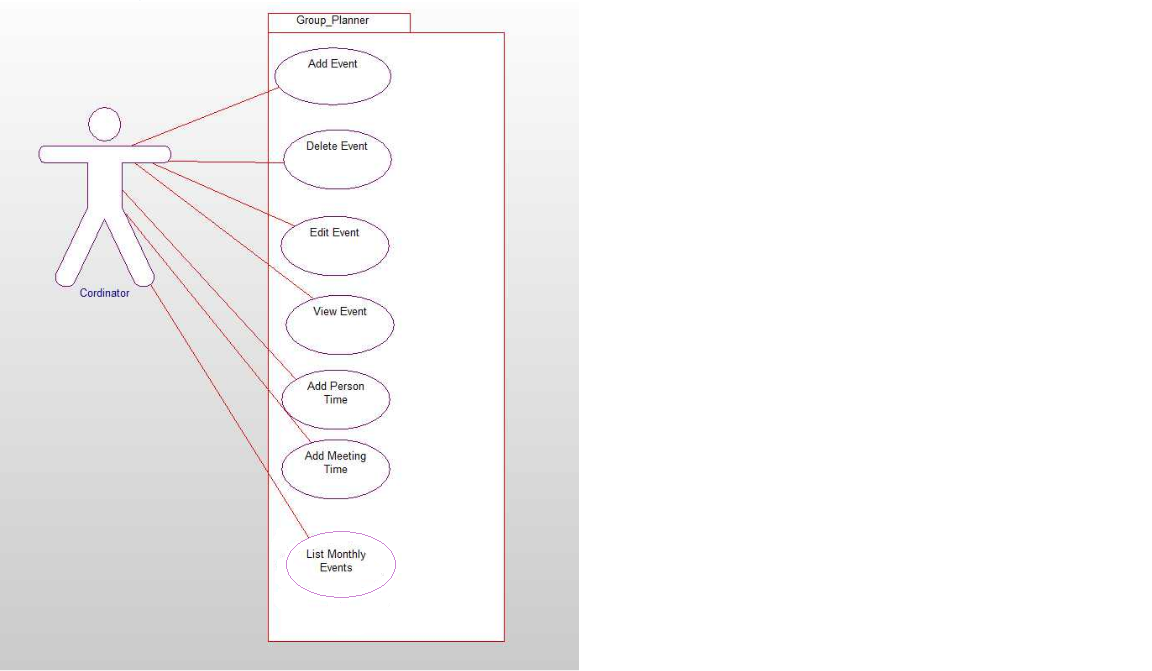
R6.2.3 Clicking Cancel will not update the information

R6.3 The system will return the user to the details event to show the changes that might have been made.

## Non-Functional Requirements

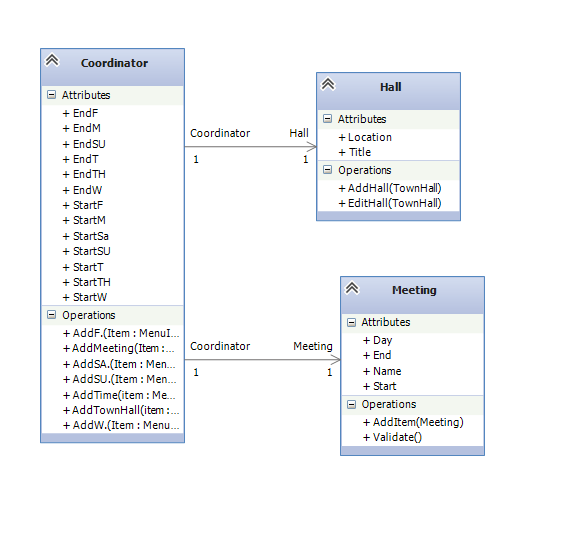
1. The system shall encrypt data that is given to the Coordinator and to the workers to keep schedules confidential (i.e. by logging in)
2. The system will be designed with the mindset of ease of usability to keep it simple for the Coordinator
3. The system shall be able to create as many conference possibilities per day
4. The system will create errors if the information provided is not correct or is unable to work with all schedules

# Use Case Diagram



In the use case above, these are all the possible options that the coordinator has when they access their meeting schedules. The first four options that a user has; Add, Edit, Delete, and View Event all focus on how the meeting schedule it handled. The Three that are below it; Add Person Time, Add Meeting Time, and List Monthly Events, allows the coordinator to add individual schedules and meeting time availability in which the coordinator then can Add an Event or Delete certain events. With these two options, they add the times that the user is available (work schedule) and the availability of the conference rooms that they might be able to use. When they complete this, they will show up in the event list to show which schedules have been placed into the calendar, which then adding an event will create a chart of all the times that the workers are free and will allow the coordinator to pick the best time to hold a meeting. Viewing an Event will allow the user to check the description of the event, time it starts, location, title, people attending, and any additional notes. From here, they will be able to delete the event that they have chosen or similarly edit the event to tweak any issues.

# Class Diagram



This class diagram shows the functions that each class must execute if the user chooses to do so. This also illustrates the connections that each individual event has, what they contain and how they relate to each other. Schedule is the main idea, with each individual class containing that main idea, meaning that they are all either part of schedule or contain a schedule within them. Starting with how the worker connects to the schedule class, it shares a many to many relationships because a worker can have many schedules and a schedule can have many workers. To know which worker’s schedule, we must use a placeholder for their name and function or button that allows them to create their schedule. The hall class is very similar to the worker class in that it has a many to many relationship, title for the room, and a create schedule. The meeting class, however only has a specific many to one relationship, in that there can only be time for this meeting, but it can contain many schedules.

# Sequence Diagram

## Create Event



This portion of the sequence diagram shows how the coordinator can create the meeting space and which it must compare both the team schedule (individual users) as well as the available halls (meeting places/times). This will then alert the user to whether the schedule that they are trying to create will work effectively or not.

## Delete Event



This allows the user to delete the meeting times that they have setup.

## Edit Event



In this sequence diagram, the coordinator can edit the event that has been created and is in the schedule time. When the coordinator is finished editing the data, they are then presented with the option to save the changes that they have made. This will then return them back to the originally screen (function) and do it again or exit the program.

## List Month



In this sequence diagram, the coordinator can choose the month that they are looking at to allocate the days in which they would like to look for the meetings that need to be made that month.

## Room Schedule



In this sequence diagram, the coordinator can create a schedule for a certain room to check its availability. It will also save the information of the room (meeting times, meeting events, etc.) and will be saved under the schedule class. They will then be able to record the title and data for the room, and return all of this information back to the coordinator.

## User Schedule



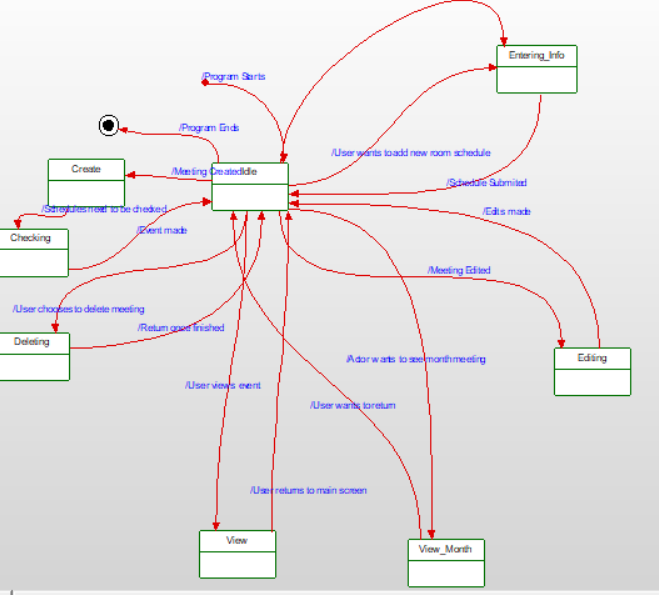
This sequence diagram is like the room schedule in that you can create a schedule for them, and save the things that they have going on for that day. It will also record who the employee is that the schedule has been created for.

## View Event



In this sequence diagram, when the coordinator asks the program to display event details, it will ask the schedule class and meeting to return the information back to the user about a particular event.

# State Diagram



In this state diagram, we can see that when the program starts, it becomes idle with no coordinator interaction. From this point, the coordinator can add a new schedule to the calendar by entering user information and returning to the idle state after submitting the newly entered information. Editing to the event can be made, which would return to the idle state after submission. View month and normal view would be accessible from this state as well, and would consist of viewing the entire month events list or viewing a specific event. There would be the option to create an event as well, which would check if the event had all of the necessary pieces of information to make sure the event is possible.

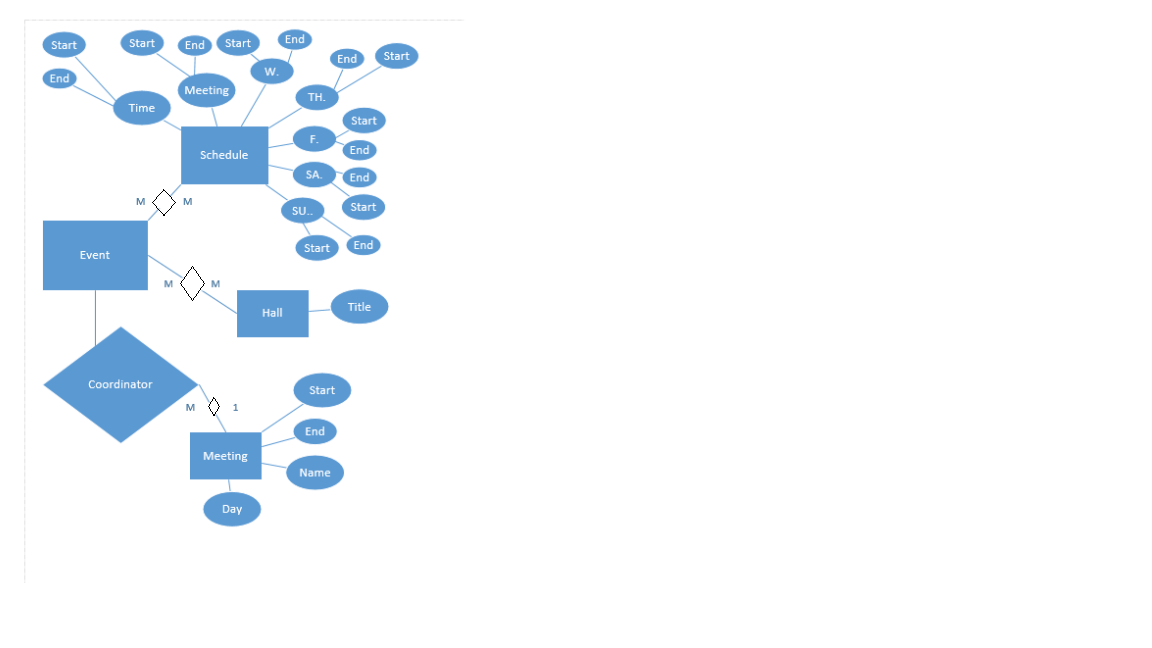
# Activity Diagram



This activity diagram details the options that the coordinator can take with in the program. It begins with prompting the coordinator for a login screen, which will allow them access other people’s schedules and otherwise private information. It will then bring them to the calendar view, which is the default view when logging into the program. They are then prompted with the ability to check if there are the necessary pieces of information to create an event. If there are (i.e. individual schedules and room schedules), then it will create the event with the details that the coordinator needs to make the meeting time work. If there are neither individual schedules or room schedules, then it will it prompt the user to create either of these options. It will then return to the calendar view and display the newly created schedules. The coordinator has the option again to create an event, and then can view the details of said event. There is then a split on with the what the coordinator can do. One option is that they can exit the calendar completely which would result in the final state being achieved. The second choice is that the coordinator can delete the event. The coordinator is given the choice of “yes” or “no”, in which if “yes” is selected, the event will be deleted from the database. The system will then update the calendar view and return to it. If the coordinator chooses the “no” option, then the deletion will be canceled and it will return to the calendar view without updating any event.

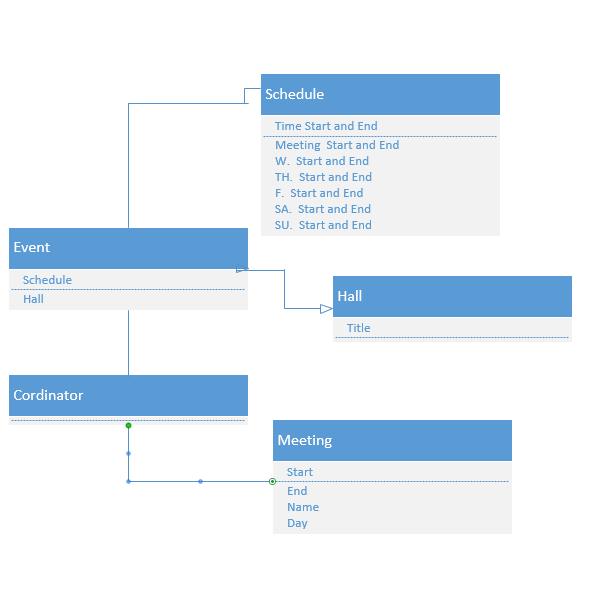
# Database Diagram

## ER Schema



This ER schema shows us how each entity relates to each other and the attributes that each one contains. The attributes in Schedule are relatively similar in that they all have the same start, end conditions in which the coordinator will need in determining the best possible times that everybody can meet to be part of the meeting. These attributes will be stored and then displayed to the coordinator. The event is the meeting that people will be part of when they determine the schedule. In this way, an event can contain multiple schedules, and a schedule can be part of multiple events, thus the Many-to-Many relationship. Like that is the relationship between Event and Hall, which is also a Many-to-Many relationship. Hall only contains one attribute, which is Title, and is basically the location and the room number of the event.

## Table Schema



# Conclusion

The goal was to create system that will allow a Coordinator to gather other workers schedules and the availability of conference rooms to check the best time for a meeting. This would allow the Coordinator to have an easier time in determining the best possible place and time to have a meeting.

# Data Dictionary

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Size | Constraint | Description |
| Schedule\_ID | Int | 50 | Primary Key | Number given to schedule |
| Hall\_ID | Int | 50 | Primary Key | Number given to hall |
| Meeting\_ID | Int | 50 | Primary Key | Number given to meeting at event |
| Time\_Start | Time | HH:MM:SS | None | Time that personal/room schedule starts |
| Time\_End | Time | HH:MM:SS | None | Time that personal/room schedule ends |
| Meeting\_Start | Time | HH:MM:SS | None | Time that meeting starts |
| Meeting\_End | Time | HH:MM:SS | None | Time that meeting ends |
| Meeting\_Name | Char | 20 | None | Name of the meeting given by meeting coordinator |
| Meeting\_Day | Char | 20 | None | Day that the meeting will occur on |
| Hall\_title | Char | 20 | None | Name of Hall |