PErsonal Calendar

Final Report

1/24/2020

CSC 340 Ethics and Software Engineering

Thomas Morris

Contents

[I. Introduction 2](#_Toc39801637)

[A. Problem Statement 2](#_Toc39801638)

[B. Proposal 2](#_Toc39801639)

[II. System Description 2](#_Toc39801640)

[III. System Requirement 2](#_Toc39801641)

[A. Functional Requirements 2](#_Toc39801642)

[B. Non-functional Requirements 12](#_Toc39801643)

[IV. Use Case Diagram 12](#_Toc39801644)

[V. Class Diagram 13](#_Toc39801645)

[VI. Sequence Diagrams 14](#_Toc39801646)

[VII. Activity Diagrams 19](#_Toc39801647)

[VIII. State Diagram 24](#_Toc39801648)

[IX. Database Design 25](#_Toc39801649)

[X. Conclusion 25](#_Toc39801650)

[XI. Data Dictionary 25](#_Toc39801651)

# Introduction

## Problem Statement

Company ABC, Inc is in need of a software system to help their employees to keep a personal calendar. This system will allow employees to keep track of their personal events.

## Proposal

I propose a personal calendar software system to help ABC, Inc to resolve the problem.

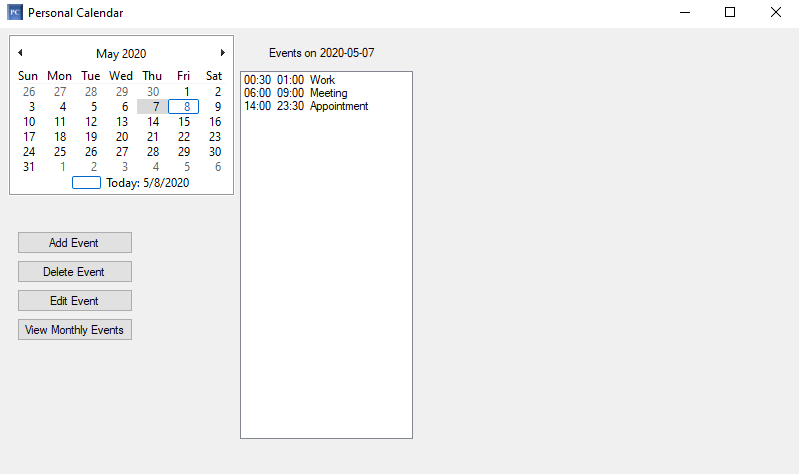
# System Description

The personal calendar system shall allow users to add an event to the calendar and check for conflicts with existing events. The system shall also allow a user to delete, edit, or view an event on the calendar. The system shall allow a user to view a monthly-based event list.

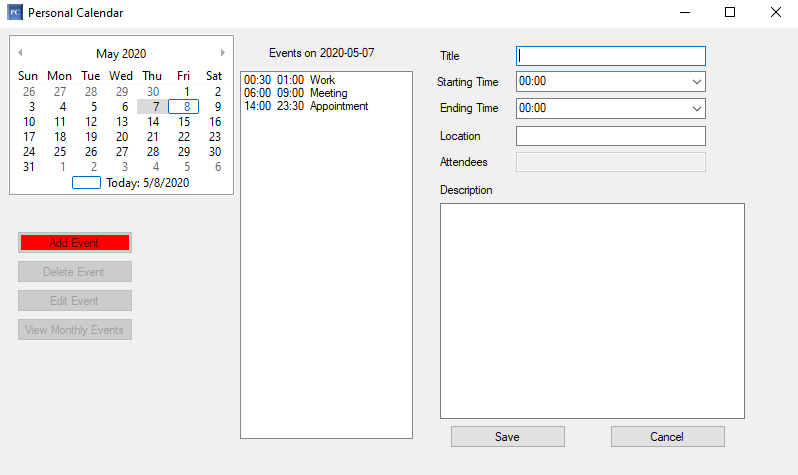
# System Requirement

## Functional Requirements

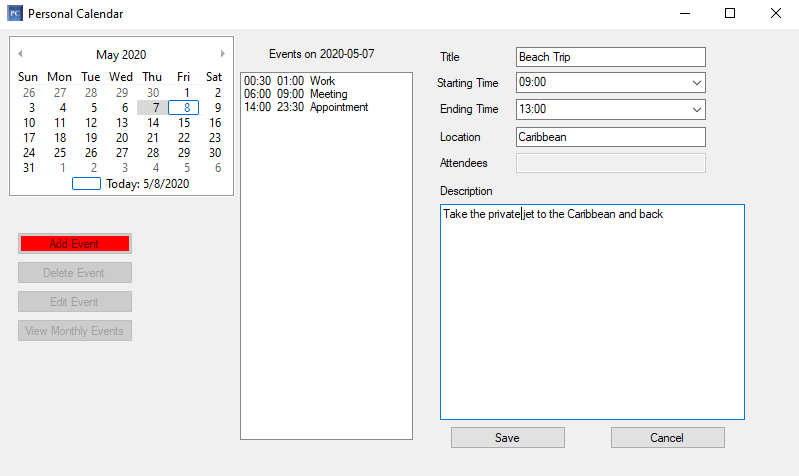
1. The system shall allow a user to add a new event to the calendar.
   1. The user shall select a date from the monthly calendar to add the new event.



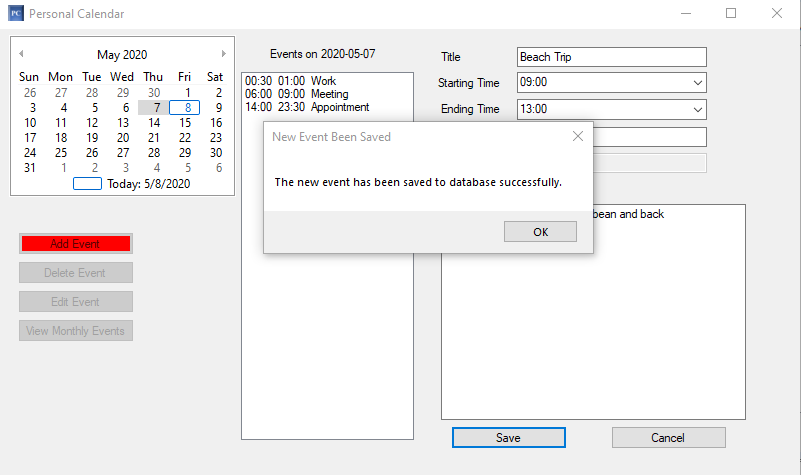
* 1. The system shall highlight the selected date on the calendar.
  2. The user shall press the “Add Event” button.



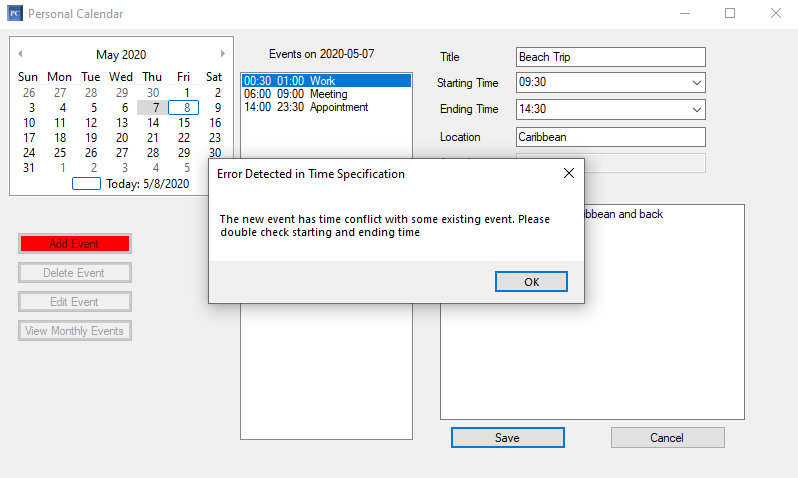
* 1. The system shall display a form to allow a user to enter the information of the new event, including event title, starting time, ending time, location, and description.
  2. The user shall enter the data for the event information and press the “Save” button.



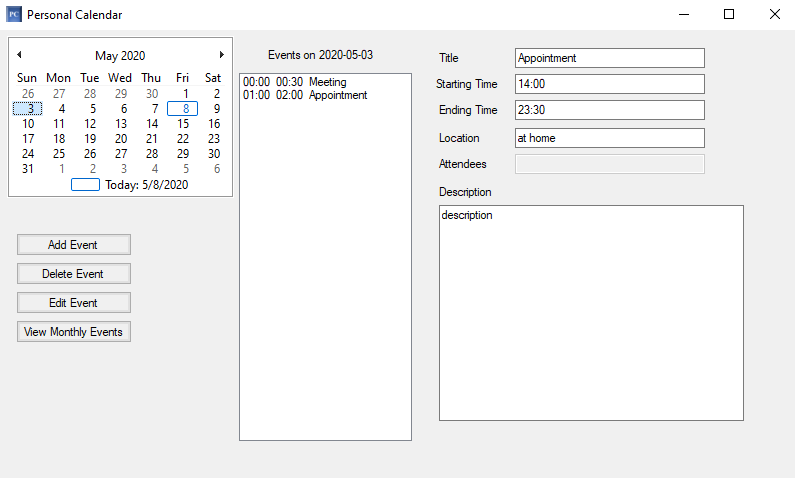
* 1. The system shall check whether the new event conflicts with existing events.
     1. If there is no conflict, then the system shall save the even to the database and display the event title on the calendar.



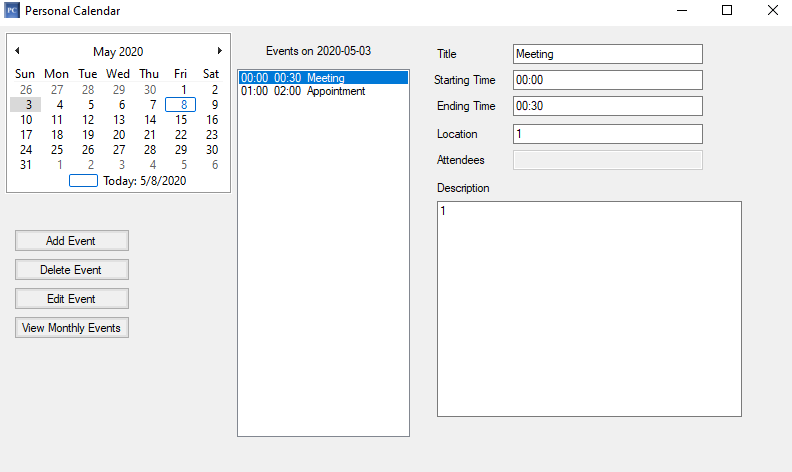
* + 1. If there is any conflict, then the system shall display an error message.



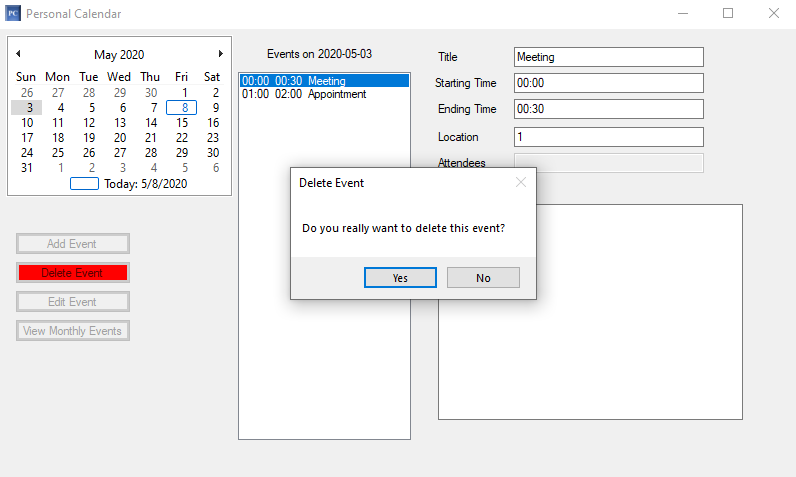
1. The system shall allow a user to delete an event from the calendar.
   1. The user shall select a date from the monthly calendar to view events for that date.



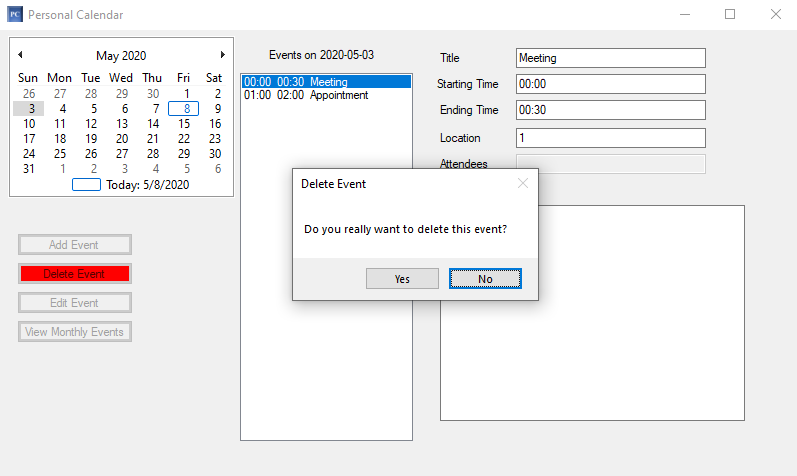
* 1. The system shall highlight the selected date on the calendar and display a list of events for that date.
  2. The user shall select an event to remove from the calendar.



* 1. The system shall display the contents of the event to the user.
  2. The user shall select the “Delete Event” button.
  3. The system shall then prompt the user to confirm if they would like to delete the event.
     1. If the user selects “Yes”, then the system shall remove that event from the database.

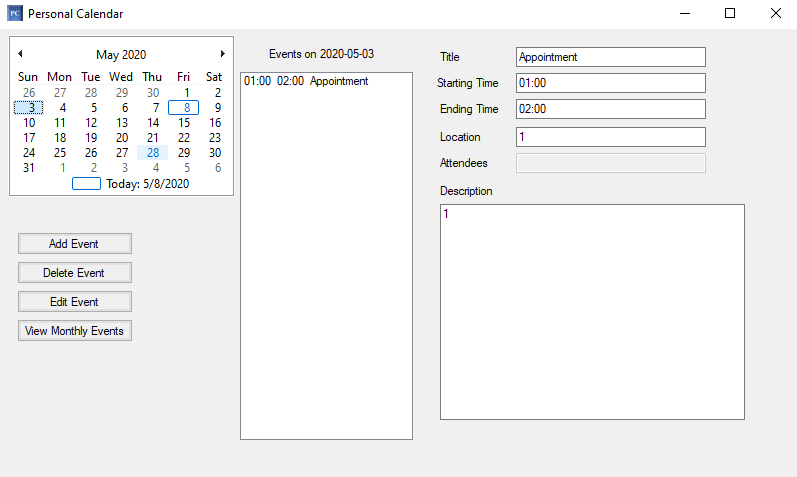


* + 1. If the user selects “No”, then no changes will be made to the event.

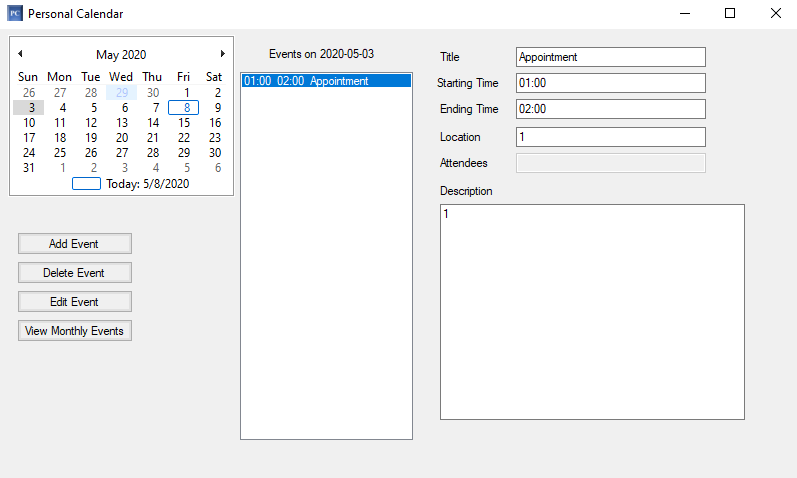


* 1. The system shall delete the event from the database.

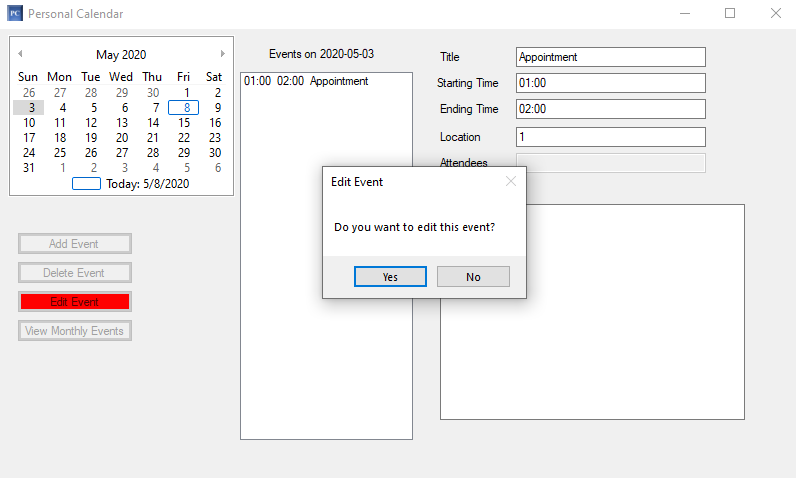
1. The system shall allow the user to edit an event on the calendar.
   1. The user shall select a date from the monthly calendar to view events for that date.



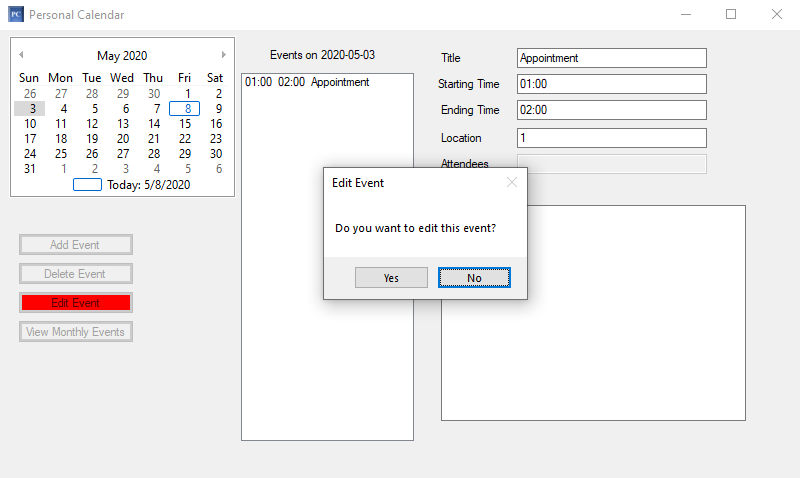
* 1. The system shall highlight the selected date on the calendar and display a list of events for that date.
  2. The user shall select an event to edit from the event list.



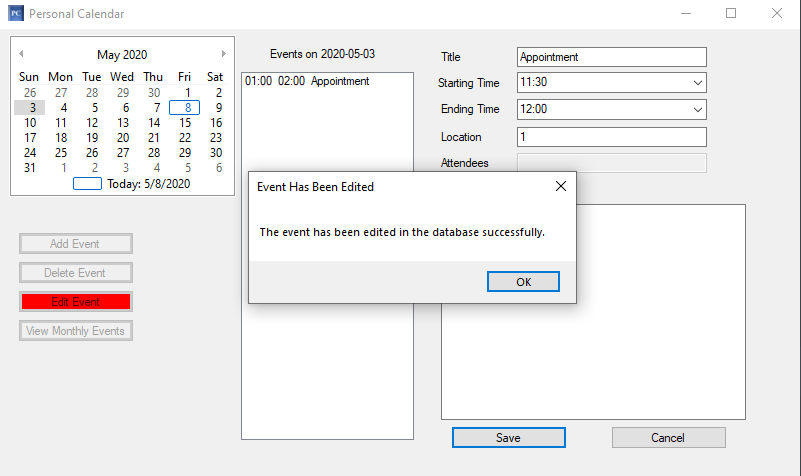
* 1. The system shall display the contents of the event to the user.
  2. The user shall select the “Edit Event” button.
  3. The system shall then prompt the user to confirm if they would like to edit the event.
     1. If the user selects “Yes”, then the system shall open the event form for editing.



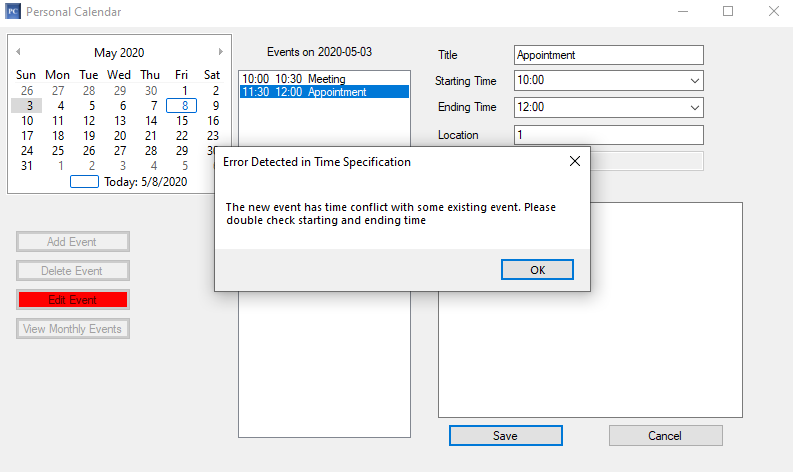
* + 1. If the user selects “No”, then the system shall revert to idle status.



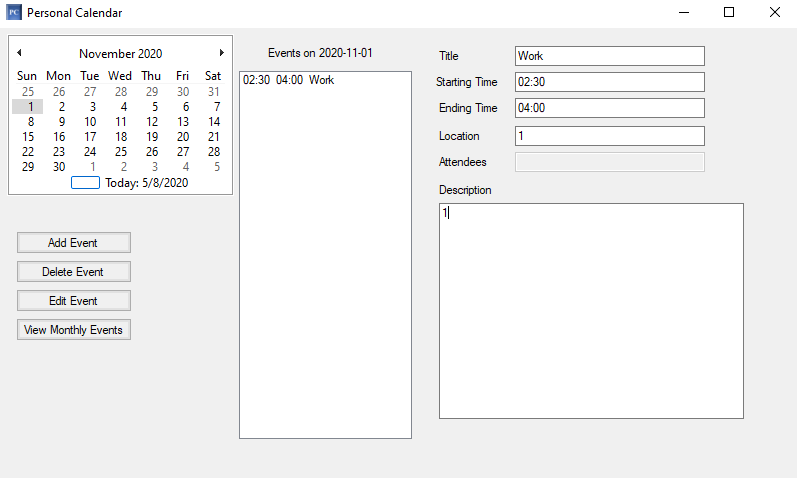
* 1. The user shall select the “Save” button.
  2. The system shall check whether the new event conflicts with existing events.
     1. If there is no conflict, then the system shall save the even to the database and display the updated event information.



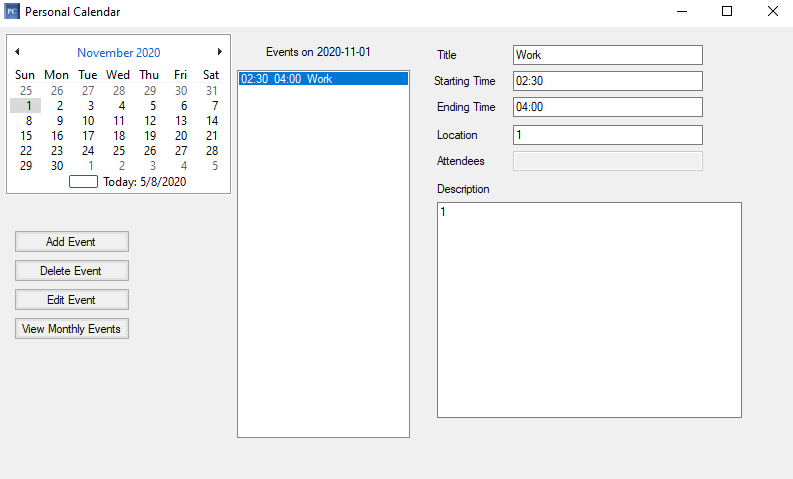
* + 1. If there is any conflict, then the system shall display an error message.



1. The system shall allow the user to view an event.
   1. The user shall select a date from the monthly calendar to view events for that date.

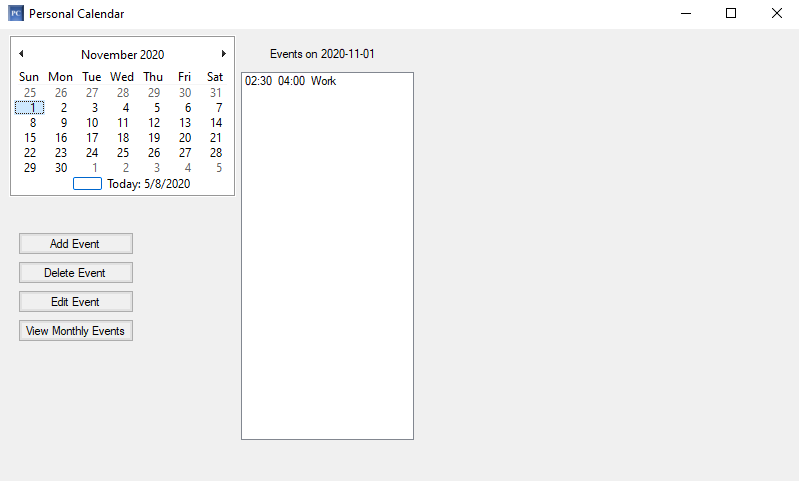


* 1. The system shall highlight the selected date on the calendar and display a list of events for that date.
  2. The user shall select an event to view from the list of events.

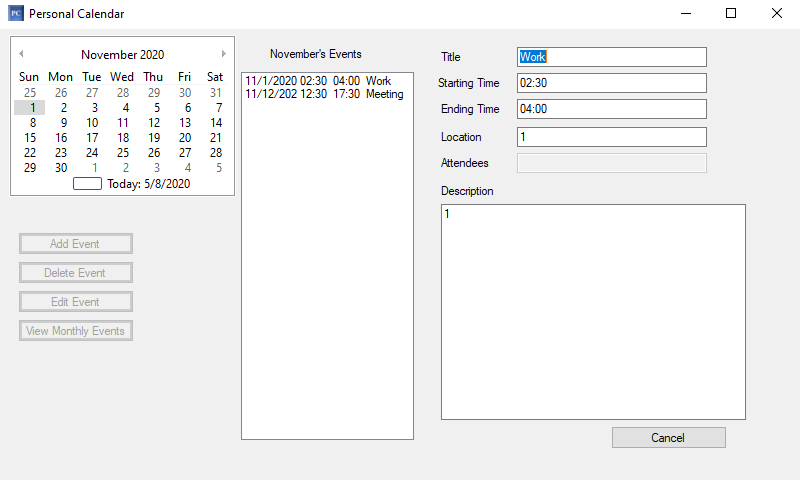


* 1. The system shall display the contents of the event to the user.

1. The system shall allow the user to view a monthly-based event list.
   1. The user shall select a day from the month they would like to view all of that month’s events from the calendar.
   2. The system shall update the calendar to show the dates for the selected day of that month.



* 1. The user shall select the “View Monthly Event” option.
  2. The system shall update the event list to reflect all events for the month.



* 1. The user shall select the “Cancel” button to return to the events for today’s date.

## Non-functional Requirements

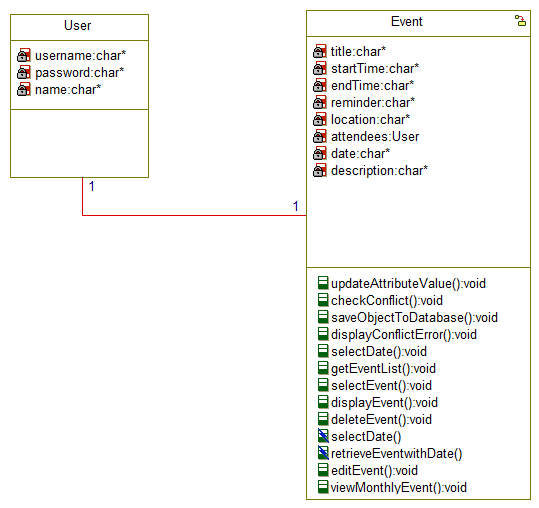
1. The system shall present information to the user in the English language.
2. The system shall use the Gregorian calendar.

# Use Case Diagram



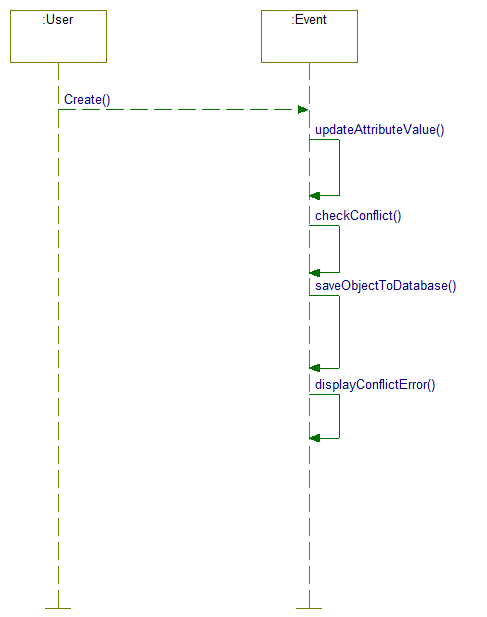
This diagram shows the expected behaviors a user is able to do in the personal calendar system. The user will be able to add, delete, edit, view specific events, and view events for an entire month.

# Class Diagram

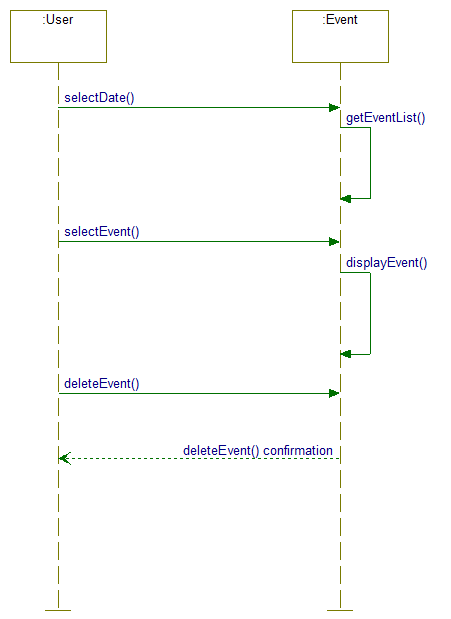


This diagram shows an overview of the system’s user and event classes. Inside of each class are the class’ operations, which includes checking for conflicts, saving events to the database, and show the users errors.

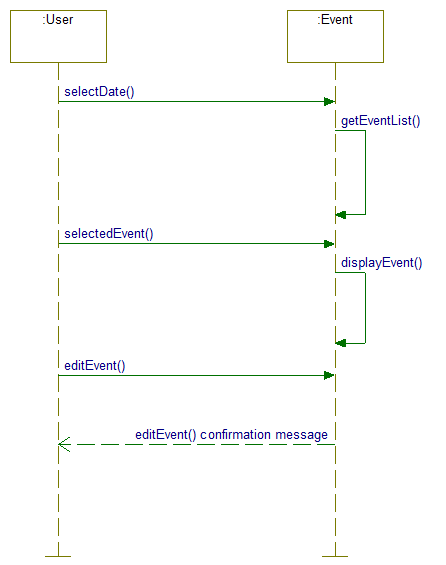
# Sequence Diagrams



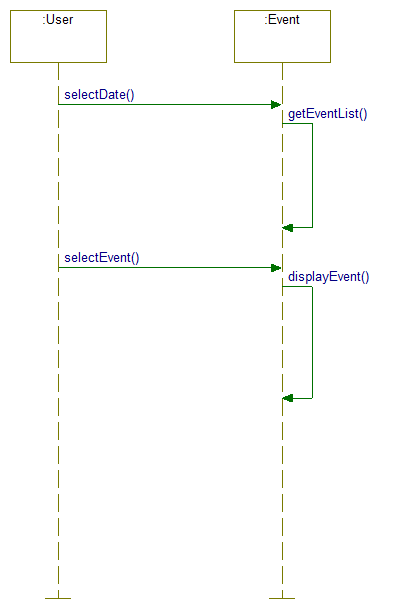
*Add Event* - This diagram shows the interactions that take place for a user to add an event to the personal calendar. The users add the event information to the form, the system then checks for conflict and saves the data into the database if there is no conflict. The system then tells the user the status of the conflict.



*Delete Event* - This diagram shows the interactions that take place for a user to delete an event from the users’ personal calendar. The user selects a specific date from the calendar and then the system shows events on that date. The users then selects a specific event from the selected date and the system displays the information for that event. The user then clicks delete event and the system tells the user the status of that action.



*Edit Event* - This diagram shows the interactions that take place for a user to edit an event in the meeting coordinator for the user. The user selects a specific date, then the system retrieves the event list for that date. The user then selects a specific event, then the system displays the information for that event. The user then selects the “Edit Event” option and changes the event information in the form. The system will the display the status of that edit.



*View Event* - This diagram shows the interactions that take place for a user to view the details of an event that is on their calendar. The user selects a specific day from the calendar and then the system displays events for that day. The user then selects a specific event and then the system displays that events information.

A screenshot of a cell phone

Description automatically generated

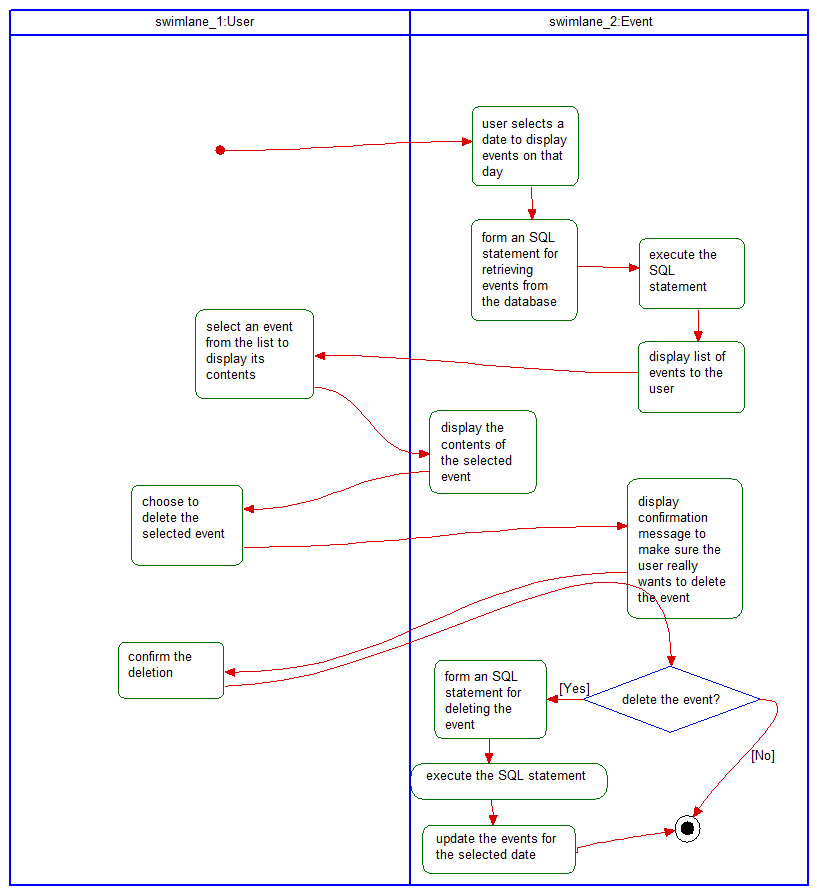
*View Monthly Event* - This diagram shows the interactions that take place for a user to view all events for the month that are on their calendar. The user selects any date in the month they want to view all events. The system gets the events for that specific day. The user then selects “View Monthly Events,” then the system retrieves the monthly event list and displays the monthly events.

# Activity Diagrams

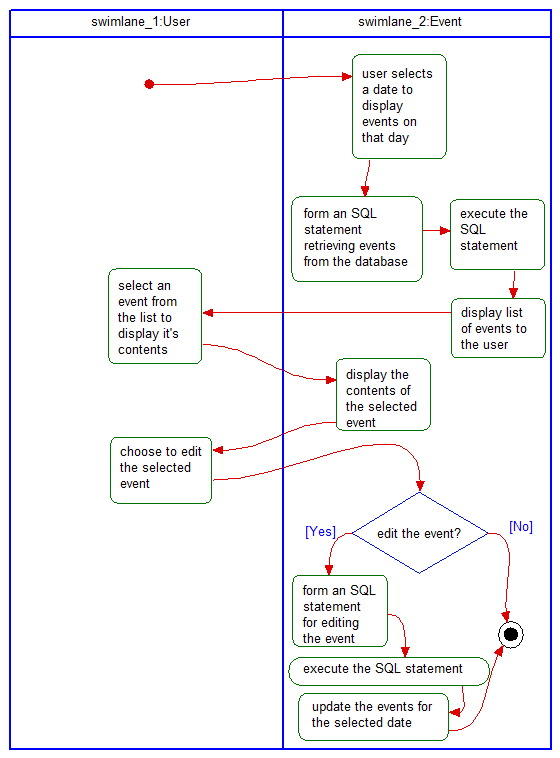
A screenshot of a cell phone

Description automatically generated

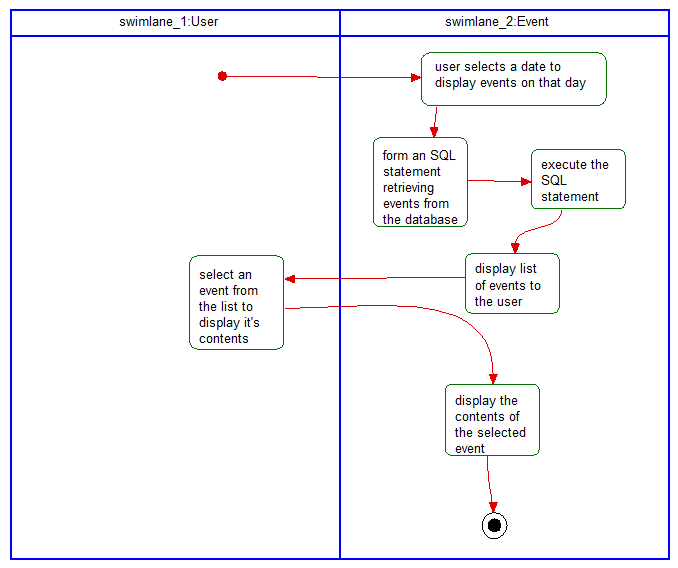
*Add Event* - This diagram shows the activity flow of the system when a user adds an event to their personal calendar.



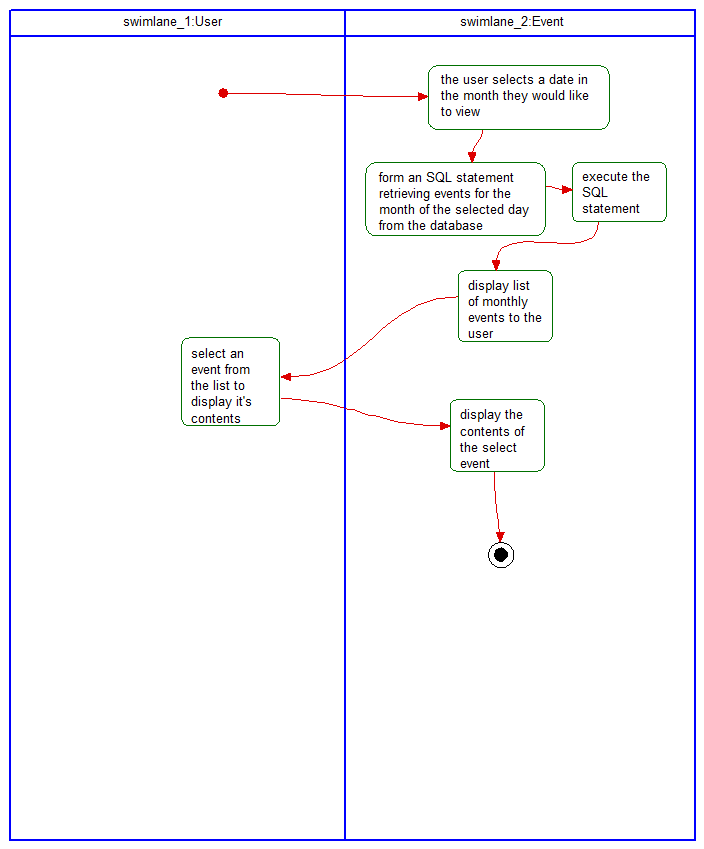
*Delete Event* - This diagram shows the activity flows that take place for a user to delete an event to from their personal calendar.



*Edit event* - This diagram shows the activity flows that take place for a user to edit an event on their personal calendar.

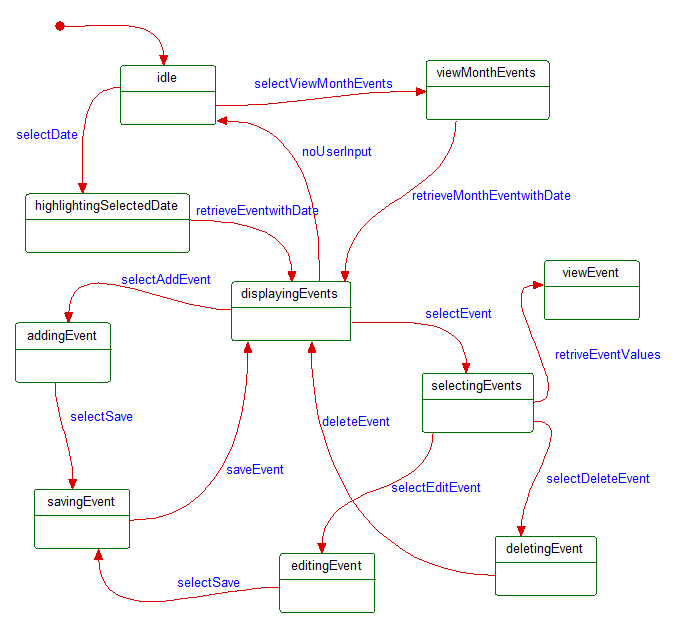


*View Event* - This diagram shows the activity flows that take place for a user to view an event that is on their calendar.



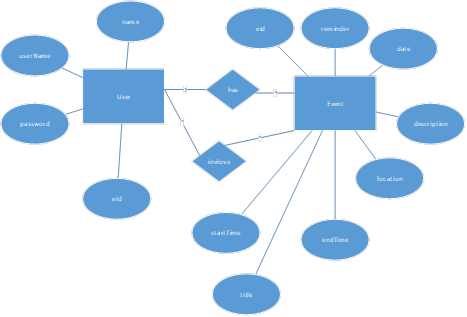
*View Monthly Events* - This diagram shows the activity flows that take place for a user to view all events for the month that are on their calendar.

# State Diagram



This diagram displays the state of the personal calendar system during a specific user action or system behavior. All use case lead back to displaying events (which includes an empty event display for no events) and with no user input displaying events leaves the system in an idle status.

# Database Design



User

|  |  |  |  |
| --- | --- | --- | --- |
| empid | name | username | password |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| eventid | title | date | starttime | endtime | reminder | description | location | empid |

Event

Involve

|  |  |
| --- | --- |
| empid | eventid |

# Conclusion

The calendar system created allows employees at ABC Inc to keep track of their personal events.

# Data Dictionary

Conflict – In a calendar system, this means that either the start time or end time coincides with another event or the start time takes place after the end time.

Database - an organized collection of data, generally stored and accessed electronically from a computer system.

Event – added by a user. These have a date, start time, end time, title, location, and description. These are tied to a specific user.

Gregorian calendar – a solar calendar with 12 months of 28–31 days each. A regular Gregorian year consists of 365 days, but in certain years known as leap years, a leap day is added to February.

Use case – a specific situation in which a product or service could potentially be used.

User – a person that has events and is used to check the availability of events.

State – specific instance in time