Bilkent University

Department of Computer Engineering

Object-Oriented Software Engineering Project

CS 319 Project: Tempo

Analysis Report

Project Group: 1.G

Member Names: Mert Saraç, A. A. M. Jubaeid Hasan Chowdhury, Burak Erkılıç, Kaan Kıranbay

Course Instructor: Eray Tüzün

Progress Report
February 17, 2018

Table of Contents

Hata! Yer işareti tanımlanmamış.

Analysis Report

CS 319 Project : Tempo

1. Introduction

1.1 Purpose

This report is a description of a Software Requirements Specification (SRS) for our smart schedule application that is called “Tempo”. This report includes many other features that we want to add to this application. These features will be mostly focused on one particular feature which is the thing that we call a “Smart Scheduling”. Furthermore, this report includes an overview of the application, describes the basics of the application by using UML: it describes functional requirements, non-functional requirements, use-case models including scenarios and use-case diagrams.

1.2 Content of The Project

Tempo is a smart scheduling app that combines to-do-list, scheduler and event calendar to help people organize their time with other people more efficient and more effectively. Tempo’s core idea is to enable people to arrange their meetings easily by seeing other individuals’ free times. Tempo also provides a help for self-improvement by reminding tasks that people should do, keeping track of people’s activities and statistics about them.

1.3 Terms and Definitions

Since the project has many features, there are some terms that are should be explained:

Smart Scheduling: This feature makes people be able to use their time more efficiently with their friends or co-workers by showing the common time among people to the users.

Event: This feature is the core feature of Tempo. An event is simply an activity that can be whether with your friends (that is called SmartEvent) or without them (that is called Personal Event). In addition, it has some other categories like timeless event, timing event, temporary event and permanent event so, Tempo enables users to combine these categories for describing their event in the best way.

SmartEvent: This feature is an event that is created by one of the users' requests to other users who are attending this meeting. For this, when one user is scheduling, he/she can see who is available during this period so he/she can easily change the time period for other individuals. When decided on the time, the user request for SmartEvent and then, other participants are notified for this meeting, its explanation, participants, time period. Other participants can whether accept this request or reject. When one participant accepts it, the system waits for other participants response. If all of them accept it, this SmartEvent is added to all participants schedule. If any participant does not accept it, system inform the one who creates the event for who did not accept and asks "Do you want to still arrange this event?" without the one who did not accept.

Personal Event: This feature is an event that user's friend list cannot see what user is doing on this time period but they can see he/she is not available during this time period.

Timeless Event: This event type is that is can happen anytime in a day. In simple words, there is no specific time period to do this event like drinking water.

Timing Event: This event type has specific time allocated time for doing this event. For schedule this, user use the weekly-time table.

Temporary Event: This event type is one-time event unless user specifies it will repeat.

Permanent Event: This event type is repeated so when user adds this event to his / her schedule, it will remain until user decides to this event will not be repeated anymore. For this feature, user can adjust how often this event will repeat like every day, every other day, weekly etc.

2. Overview

In general, Tempo is an application that uses social network idea for scheduling. Like other apps, this application will be also easy to understand and to use. Visuals will help users by making them check and prefect their time management. Users will be able to use this application by registering to the system. After registration process, people will be able to login and start benefiting the application's many features. Those features will be adding events to their calendar easily, adding their friends or coworkers for choosing the common free time and making an event that is suitable for everyone that wants or is forced to join it, and by doing these regular or irregular events users will be able to track their statistics and get an idea of what they should do to make their time management better by looking at those statistics and the graphs that those statistics are produced by the application.

2.1 Registration and Login

Users needs register or login to access their accounts. To register, every user need a unique username and password. After that, he/she needs to give information about his / her name and surname, email address and type password again to eradicate mistakes about it.

2.2 Calendar

Calendar is main frame in Tempo since the user can see what user will do in this calendar with their start time and end time and when others are available. User can also change one activity time period by just dragging it to another time period. If this activity is a SmartEvent, the system sends a message to other participants for this request and this activity's place does not change until all participants accept the request. However, if it is just a personal event, the user is free to change the time of this event.

2.3 Events

Events are core concept in Tempo. There are many types of events that are SmartEvent, personal event, timing event, timeless event, permanent event and temporary event. By combining these types of events, the user can describe an event in the best way and also user can add some stickers to them for visualization.

2.4 Smart Scheduler

Smart scheduler's main idea is adding some activities for a group. User can see free times of other possible participants so, user can select the best possible time period for their meetings. When time period is finalized, user (in this case it is called event admin) select other participants and in this way, they are notified about this meeting. When one participant accepts this meeting, that participant is added to meeting and system waits for other participants' response. When all participants accept it, it is added every participants' calendar. However, if one or more participant reject this request, system asks "Do you want to still arrange SmartEvent without … ?" and event admin decides.

2.5 To - Do List

To - Do list is a list of timeless events and if user did this, he can click the checkbox.

2.6 Friends List

Tempo enables users to see when their friends are available and when they are not. However, there is a huge possibility that one can have so many friends so, user can filter the ones who are not necessary to see their available and unavailable times and in this way, he is able to see the crucial ones really quick. This feature prevents a complicated appearance of calendar.

3. Requirement Specification

3.1 Functional Requirements

- Render and display schedule
- Schedule common events
- Authorized by login (username and password)
- Sign up for a new account
- Add items to To-Do list
- Sign complete elements as completed on
- Relocate events
- Delete events
- Request a meeting
- Accept or ignore a meeting
- Search friends with their username
- Add friends
- Accept or ignore friend requests
- Edit user's profile
- See schedule of user's friends

3.2 Non-functional Requirements

- A system crash should not result in data loss
- When a user send a meeting request, the user should see schedules of people who are going to attend the meeting.
- Show events of user's friends based on event types. Users can not see their friends personal events but they will see blocked time event.
- Users will be notified when they are received a meeting request.
- Completed tasks on To-Do list will be deleted

- If there is multiple people on a meeting plan, when they all accepted the meeting request, meeting will be shown as a timing event. Until everybody accepts the meeting plan it will be moderated by user who created a meeting plan.
- Meeting plans can be deleted by their creators before the plan turns to a timing event.
- Meeting timed events can be deleted but it will notify other users.
- Meeting timed events can be destroyed for all attendees by the creator of the event.

3.3 Pseudo Requirements

- The implementation language must be Java
- The implementation user interface library must be JavaFX
- The Database implementation must be MongoDB

4. System Models

4.1 Use-Case Model

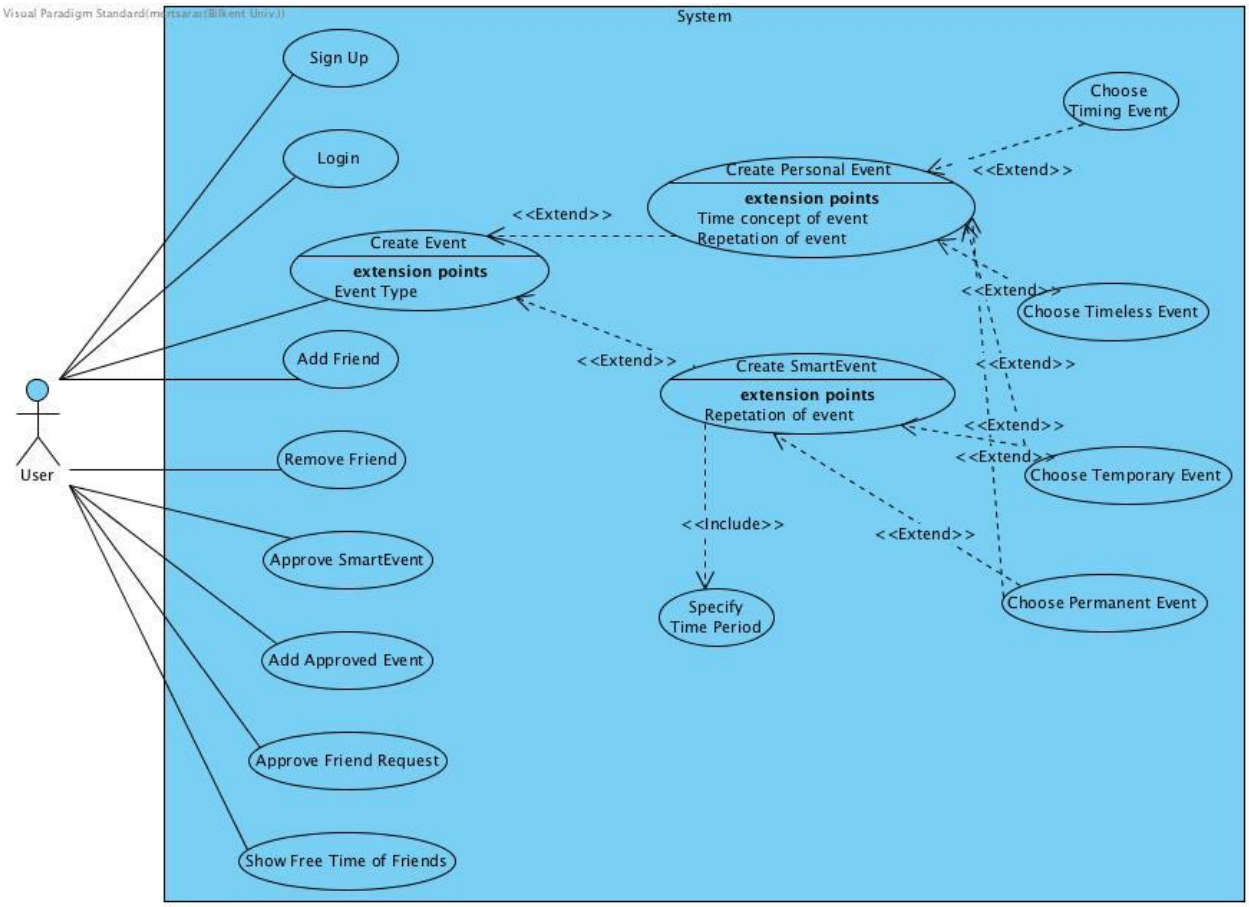


Figure 1 - Use Case Diagram

Use case name: ***Sign Up***

Participating actors: User

Flow of events: 1. The user enters his name, surname, email and password
2. The system stores this information in the database and logs user in.

Entry condition: First time use

Exit condition: The user has provided the database with a unique username and an appropriate password.

Quality requirements: The password must be at least 6 characters long. And the password is case-sensitive. The username must be unique.

Use case name: ***Login***

Participating actors: User

Flow of events: 1. The user enter his username and password.
2. The system checks this information against database and logs user in.

Entry condition: Must sign up once before.

Exit condition: The user's username and password has matched with of the UserInfo of the database.
Alternative Flow of events: If the username or password doesn't match, a warning is given and the user is informed to try again.

Use case name: ***CreatePersonalEvent***

Participating actors: User

Flow of events: 1. The user clicks “create event” and specifies the time and date.
2. The system creates the event and displays on the calendar.

Entry condition: The user is logged in.

Exit condition: The user has created a new event which is displayed on his calendar.

Alternative Flow of events: If the user clicks the checkbox repeat then this event is repeated accordingly. If the user clicks timeless event, then the event is added to “timeless event” section instead of calendar.

Use case name: ***AddFriend***

Participating actors: User

Flow of events: 1. The user searches the other user and clicks his username.
2. The system shows the profile of the other user.
3. The user clicks the add friend button.
4. The system responds by sending friend request to the other user.

Entry condition: The user is logged in.

Exit condition: The user requested and if approved the users are added to each other’s friend list.

Alternative Flow of events: If the request is not accepted then no friend is added.

Use case name: ***CreateSmartEvent***

Participating actors: User

Flow of events: 1. The user selects a group of friends.
2. The system responds by showing a superimposed version of the friends' calendars on his calendar and free time available.
3. The user chooses a time from available spot.
4. The system sends all of the selects friends approval request.

Entry condition: The user is logged in.

Exit condition: The user has sent his selected friends a request for a timed event.

Use case name: ***ApproveSmartEvent***

Participating actors: User

Flow of events: 1. The system display a requested event from a friend on your calendar.
2. You click approve.
3. The system sends this information to that friend's SmartEvent object.

Entry condition: CreateSmartEvent request from a friend.

Exit condition: The user responded by expressing his approval or disapproval.

Use case name: ***ApproveFriendRequest***

Participating actors: User

Flow of events: 1. The system display a friend request from a user.
2. You click approve.
3. The system sends this information to that friend and saves them as friends.

Entry condition: AddFriend request from a friend.

Exit condition: The user responded by expressing his approval or disapproval.
Alternative Flow of events: If the other user doesn't accept the request, then no friend is added.

Use case name: ***AddApprovedEvent***

Participating actors: User

Flow of events: 1. The system adds the event on the user's and all of the requested friend's calendars if everyone approved of it.

Entry condition: All of the requested friend approved for the event.

Exit condition: The requested event is display on the calendar of the user and the selected friends.

Alternative Flow of events: If any of the friends didn't approve, then the event is canceled.

4.2 Object and Class Model

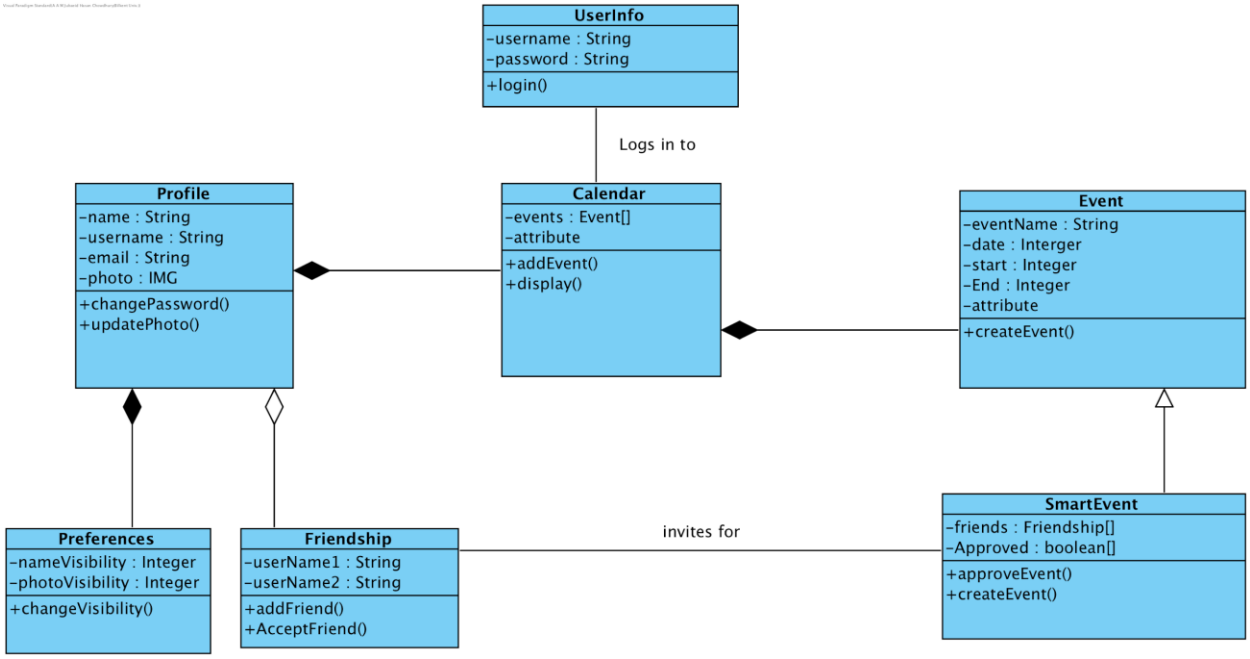


Figure 2 - Object and Class Model Diagram

This class diagram (figure 2) visualizes that when the user first enters the game, he logins through the “UserInfo” class, then is shown his “calendar” which displays events. This “Calendar” is aggregately associated with his own “Profile” which also contains his name, username, photo and a friend list. Profile is associated with “preference” which contains his preferences and “friendship” class for the friend list. Calendar is aggregately associated with “Event” class which creates events. This event class has a child class “SmartEvent” which creates smart events. This smartEvent class is associated with Friendship because only friends can be invited for smart events.

4.3 Dynamic Models

This part of the Analysis Report contains some crucial and detailed information about the Tempo by using dynamic models such as sequence diagrams and activity diagrams.

4.3.1 Sequence Diagrams

This part focuses on how starting the application occurs, how adding events to calendar occurs, how adding a common event to calendar occurs, and how adding a friend occurs by using sequence diagrams.

4.3.1.1 Start the Application

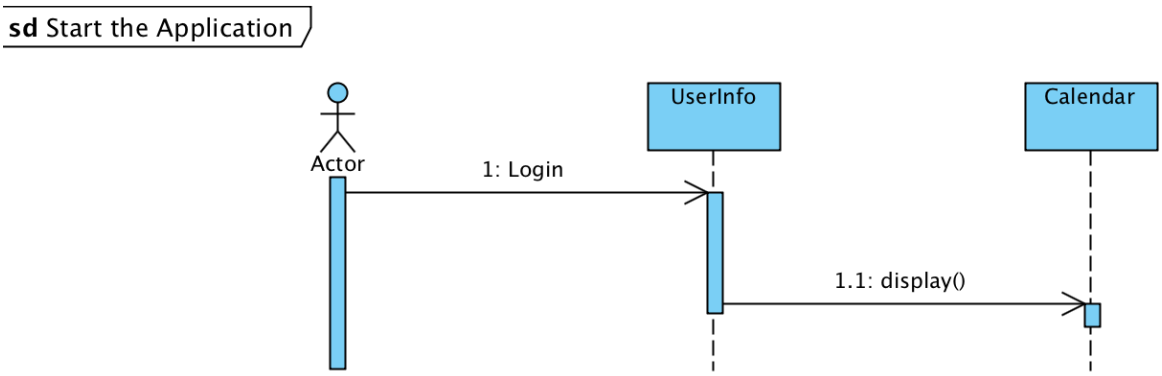


Figure 3 - Start the Application Sequence Diagram

Above sequence diagram (figure 3) illustrates the scenario explained below:

Scenario: User double clicks to the application's icon on the desktop. After clicking the icon from desktop, the system opens a window and shows the login page inside of the window. Into the login page, user writes his username and password to login to the system. The system uses user's information to open his profile and access his calendar, friends, and habits. After logging in, the system opens the user's calendar and shows his events, shows his habits and shows his friends.

4.3.1.2 Adding an Event to the Calendar

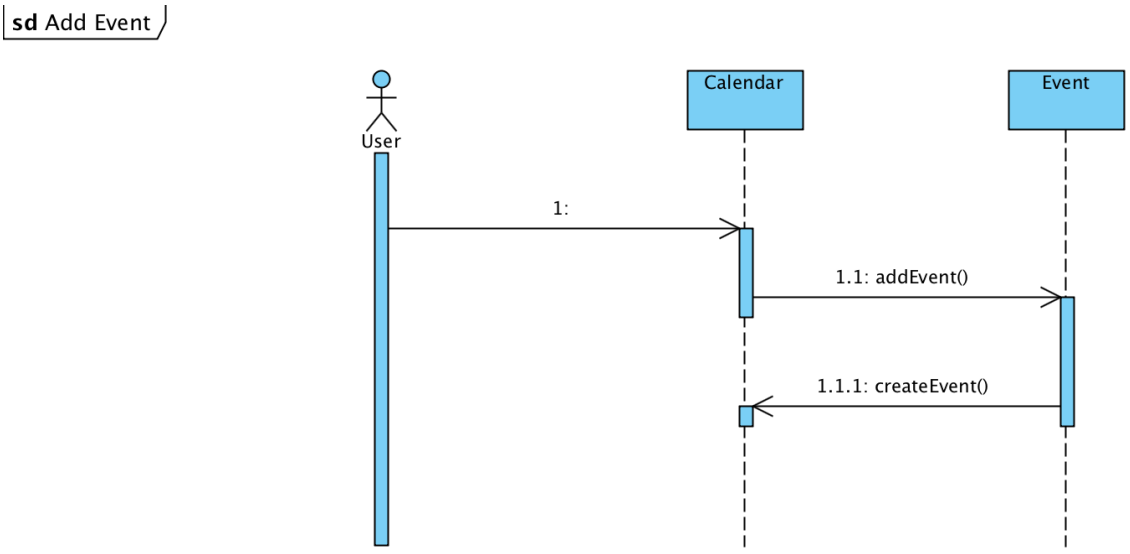


Figure 4 - Adding an Event to the Calendar Sequence Diagram

Above sequence diagram (figure 4) illustrates the scenario explained below:

Scenario: User clicks on the "Add Event" icon in the calendar window once. A new window will open to choose what the event will be and what the event's time will be. User chooses event's specifications and then clicks on a button to apply and add this event to the calendar. After button is pressed calendar window opens and shows the new event along with other events that are added before.

4.3.1.3 Adding a SmartEvent to the Calendar

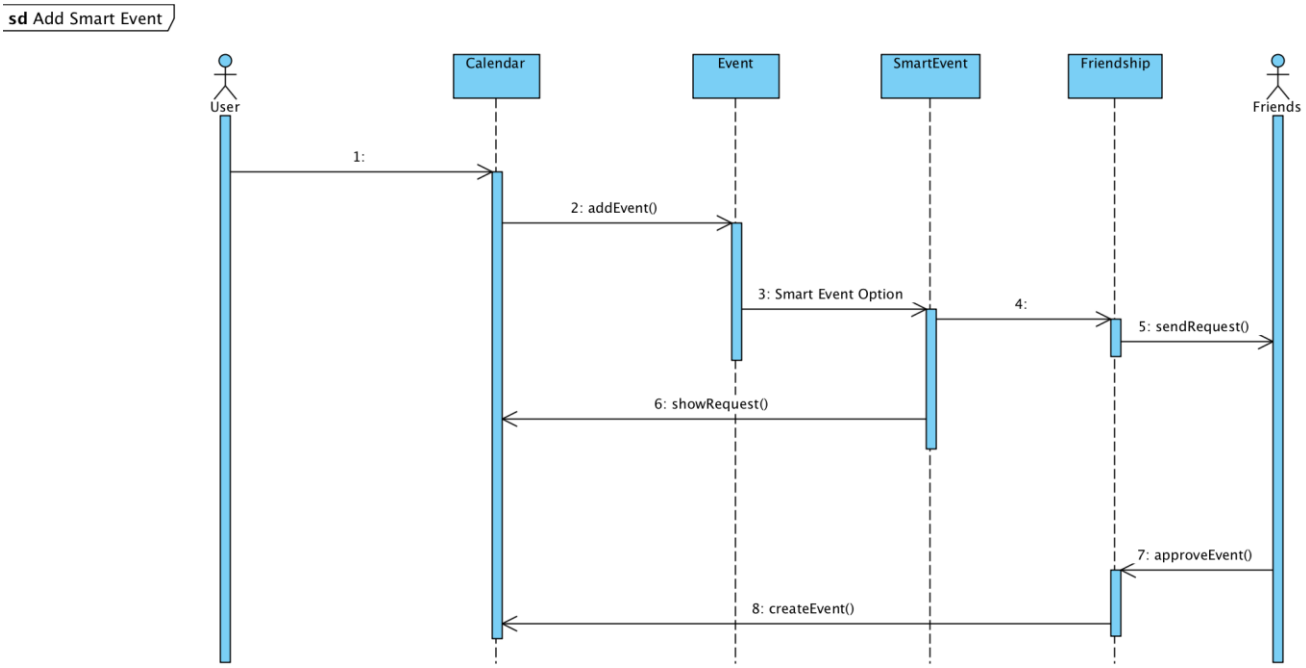


Figure 5 - Adding a SmartEvent to the Calendar Sequence Diagram

Above sequence diagram (figure 5) illustrates the scenario explained below:

Scenario: User clicks on the "Add Event" icon in the calendar window once. After a new window opens, there will be a button for smart event. If that is clicked another new window will open to choose what the event will be and what the event's time will be, and the people that the users wants to participate in this event. User chooses event's specifications and then clicks on a button to apply and add this event to the calendar. After button is pressed, the system waits to add this event to add to people's calendars' until everyone says "Okay" to this event. After everyone approves this event, everyone's calendars' shows this event as a common event.

4.3.1.4 Adding a Friend

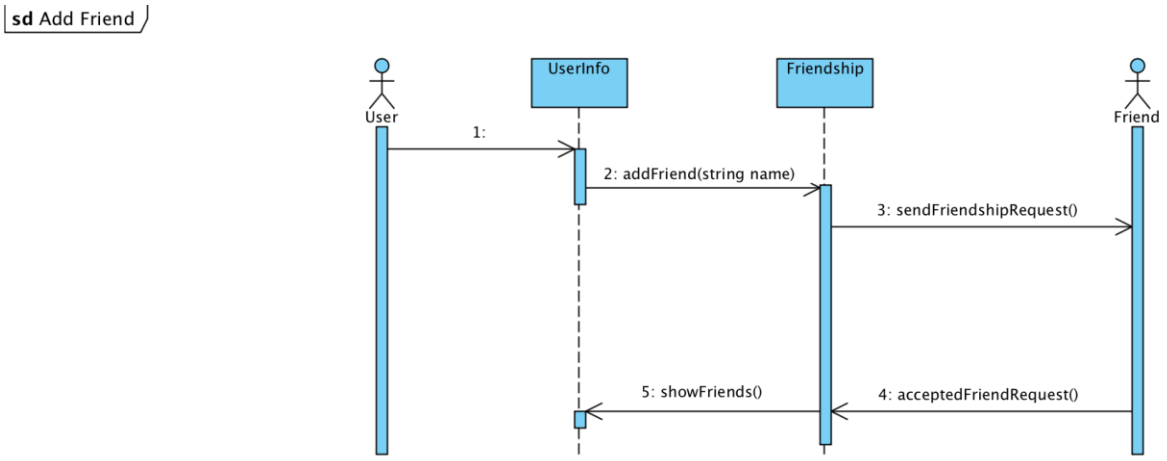


Figure 6 - Adding a Friend Sequence Diagram

Above sequence diagram (figure 6) illustrates the scenario explained below:

Scenario: User clicks on the “Add Friends” icon in the calendar window once. A new window will open for user to write his friend's name. After writing his friend’s name, user will search for this person and if this person exists in the system, user will send him a friend request. System won’t add them as friends until the requested person approves this request. After the person approves, both of them will be able to see them on their calendar window.

4.3.2 Activity Diagram

This part focuses on activities of the system during the usage of application are also shown in the activity diagram.

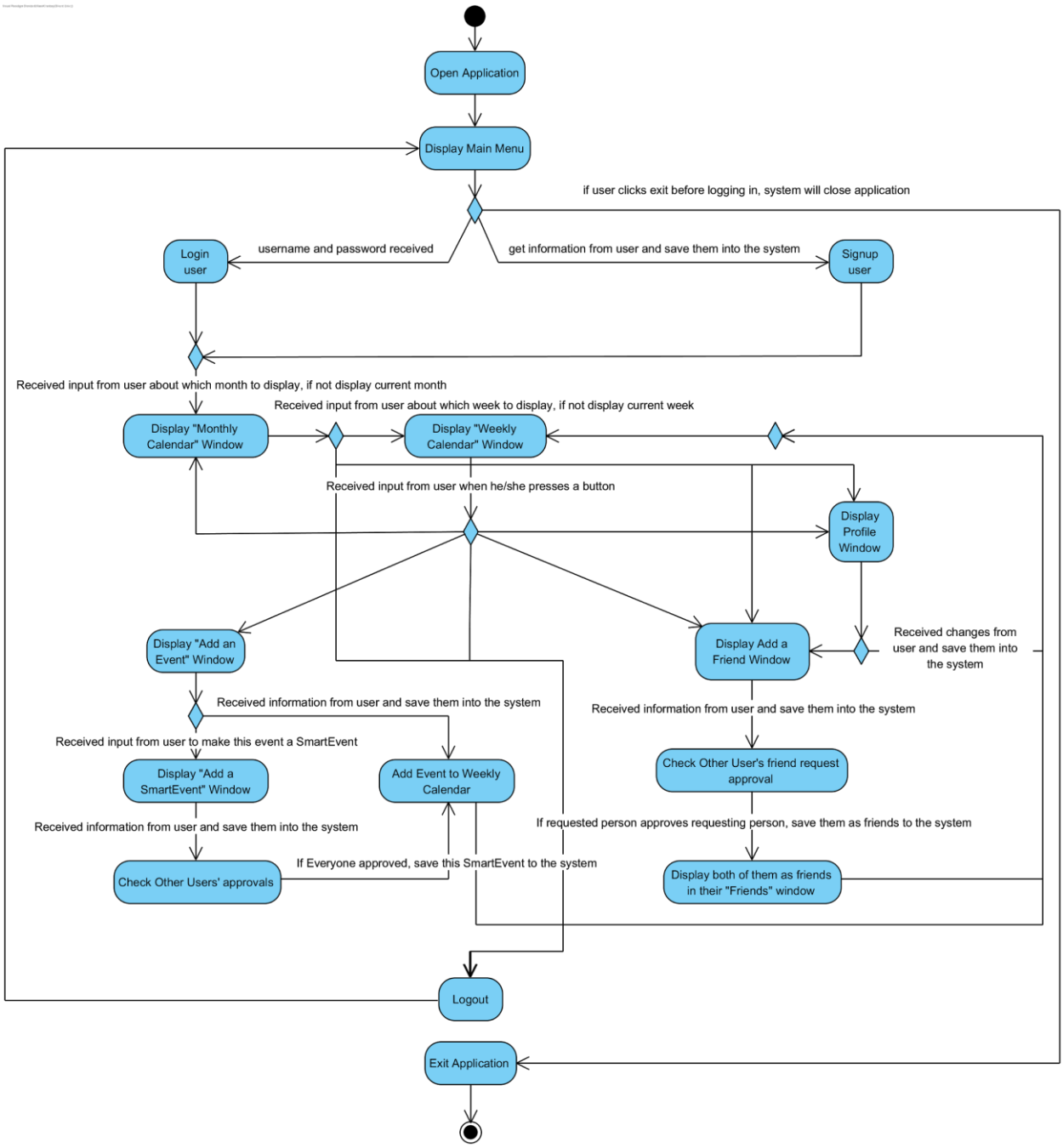


Figure 7 - Activity Diagram

This activity diagram (figure 7) indicates how the system works and flows. It illustrates the description of the application explained below:

A user will open the application by clicking on the application's icon on the desktop. System will initialize and display the main menu. Then the user will have 3 options: to login to the system or to sign up to the system or exit the application. When user logs in to the system, user will be able to use the application and its features. Logging the user in to the system will be by getting username and password from user and then system will check if the person exists or doesn't. If the person exists, then system will show them their choice of calendar. If not, then system will ask them to register. Registering the user in to the system will be by getting their name and surname, username, password, and e-mail. After registering and saving the information of the user to the system, system will open the application automatically for them. Then the usage of application will begin.

After logging in, the system will display monthly calendar after user chooses a month to open. If they won't choose a month to open, system will automatically open the current month as default. The user will have 2 options: Logging out or choosing a week. After pressing the button to open a month, system will display the monthly calendar. Then, the user can be able to use a button to open weekly calendar, they should also select a week to open. If they won't choose a week to open, the system will also automatically open the current week as default. However, after pressing the button to logout, the system will display the main menu window again.

After displaying the weekly calendar window, user will be able to do 7 different things: Logging out, displaying their profiles in a new window, displaying "Add a Friend" window, displaying "Add a Event" window, display their to-do list, display their friends and going back to displaying the monthly calendar. Again, after pressing the button to logout, the system will display the main menu window again. After pressing "Display Profile" button, the system will display the profile page of the user. In this window, the user will be able to display "Add a Friend" window and change his/her information like username or password. Those changes will be saved to the system. Furthermore, the user will be able to go back displaying the weekly calendar.

After displaying the weekly calendar window, the system will provide users to add events to their schedule. The user will have 2 different options: Creating a personal event or creating a smart event. After creating a personal event, the system will take the information from the user and save it to the system to show the event on the user's calendars. After creating a smart event, the system will take the information from the user and save it to the system but until everyone

person that has been invited to this event approves, this event will not be shown on the weekly calendars of users'. If even a person declines this event, the event will be deleted from the system.

After logging out from the system, users will be able to exit the application. The system will close the application if the user exits the app.

5 User Interface

5.1 Navigational Path

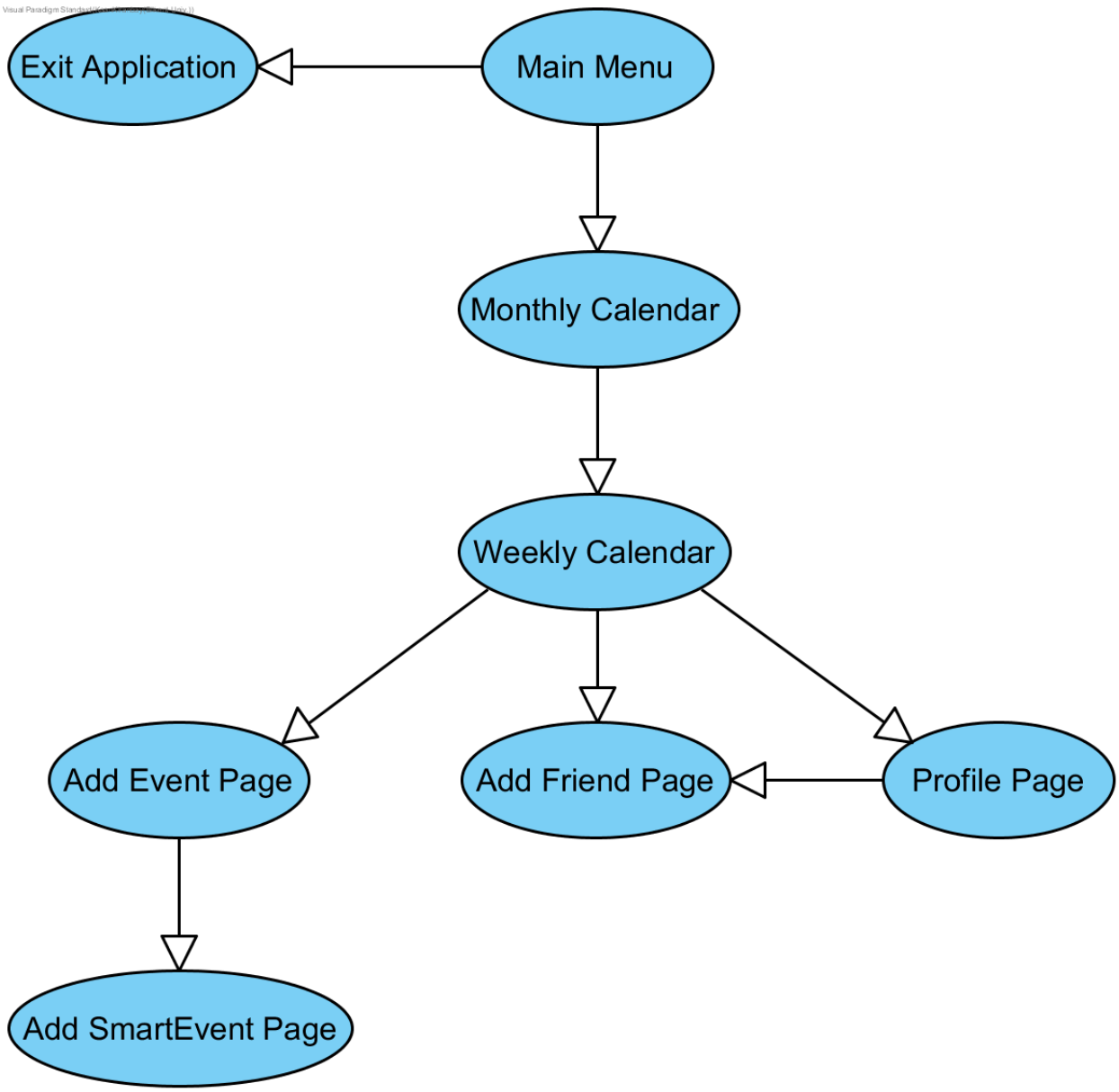
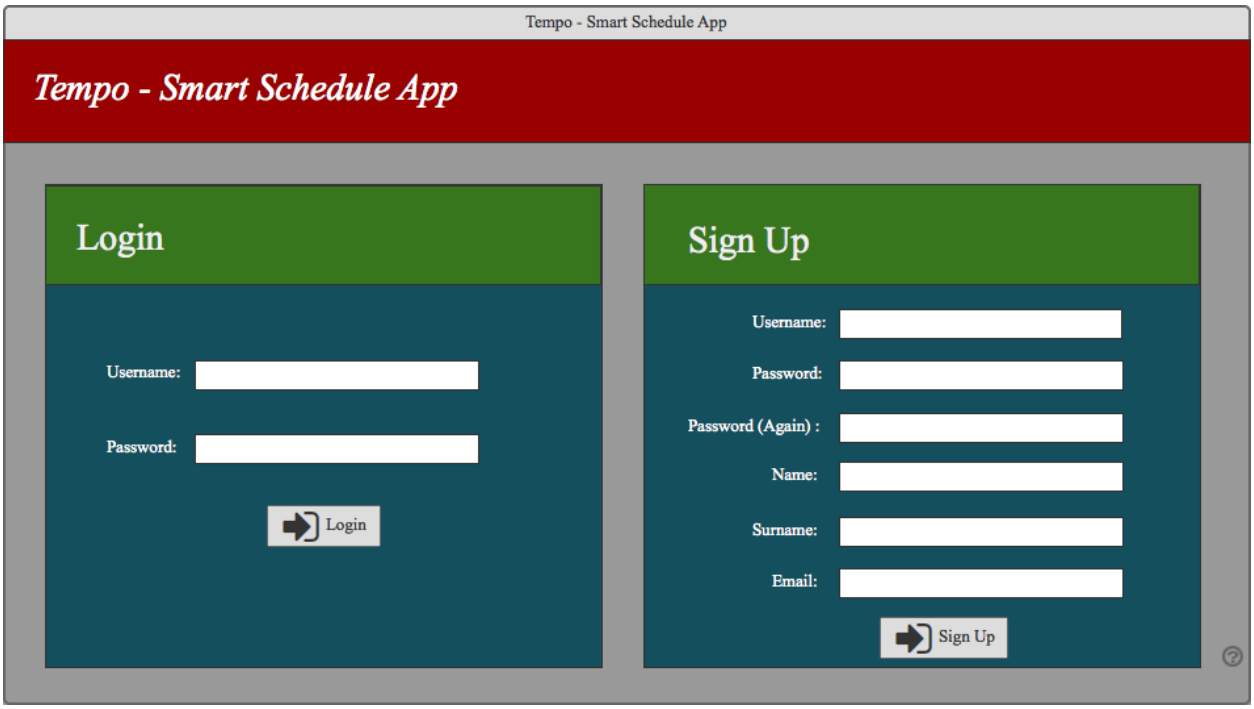


Figure 8 - Navigational Path

5.2 Screen Mock-ups

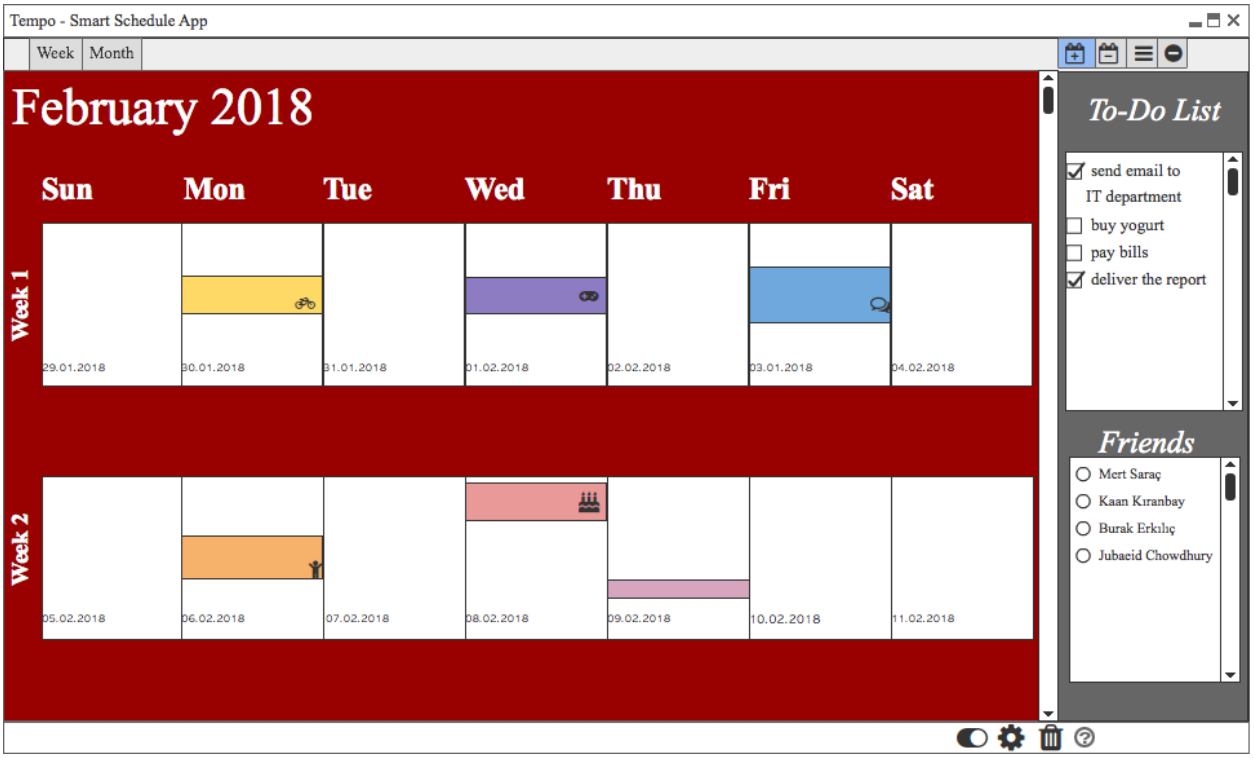
5.2.1 Opening Page



When user opens the app, user encounter opening page that contains Login and Sign Up pages. If user has no account, registration is needed. User need to choose a unique username and password. Then, for elimination of mistakes, user need to type password again. System checks whether they are equal. After that user needs to give information about name, surname and email. Final step is clicking to sign up button and user is signed up!

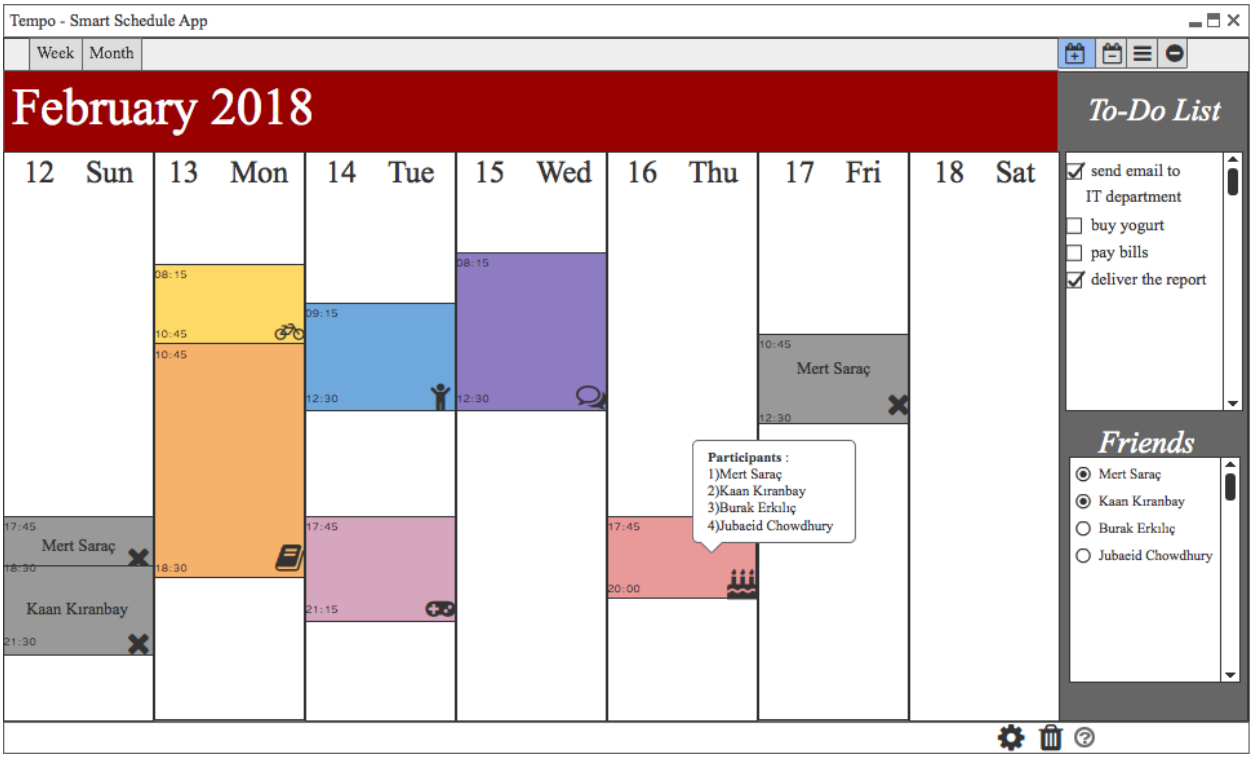
On the other hand, if user already has an account, typing username and password is needed to access profile.

5.2.2 Month Apperance of Calendar



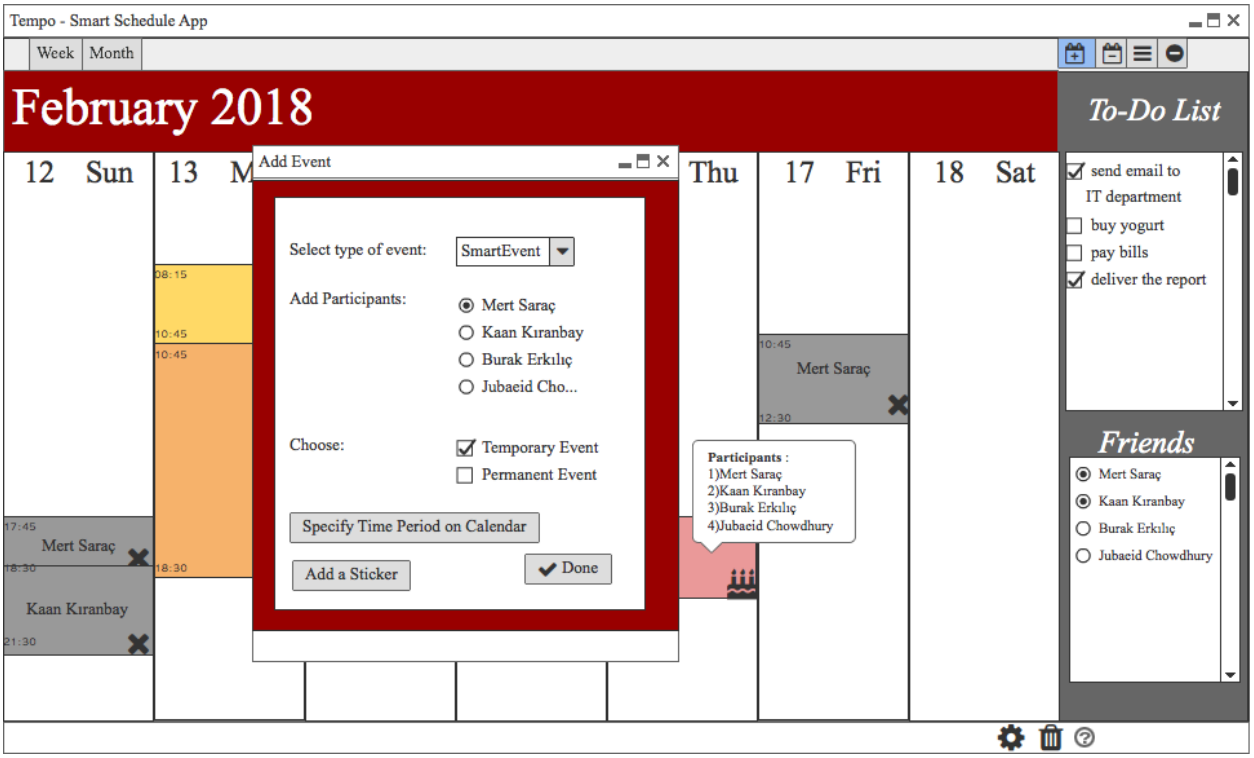
Opening page of Tempo is calendar page that is also main menu. From this menu, user can see all activities in general but not detailed for all 4 week of the month. If user wants to more detailed view for schedule, there is a week button to see detailed view of that current week. There are also 2 panels that are To-Do List and Friends next to the calendar page. To-Do List contain timeless events with checkbox so user can always see tasks to perform and when one or more of them are done, user can check it. Other panel is Friends that shows all of users friends with the radio buttons which enable user to select friend to see his unavailable and available times

5.2.3 Week Appearance of Calendar



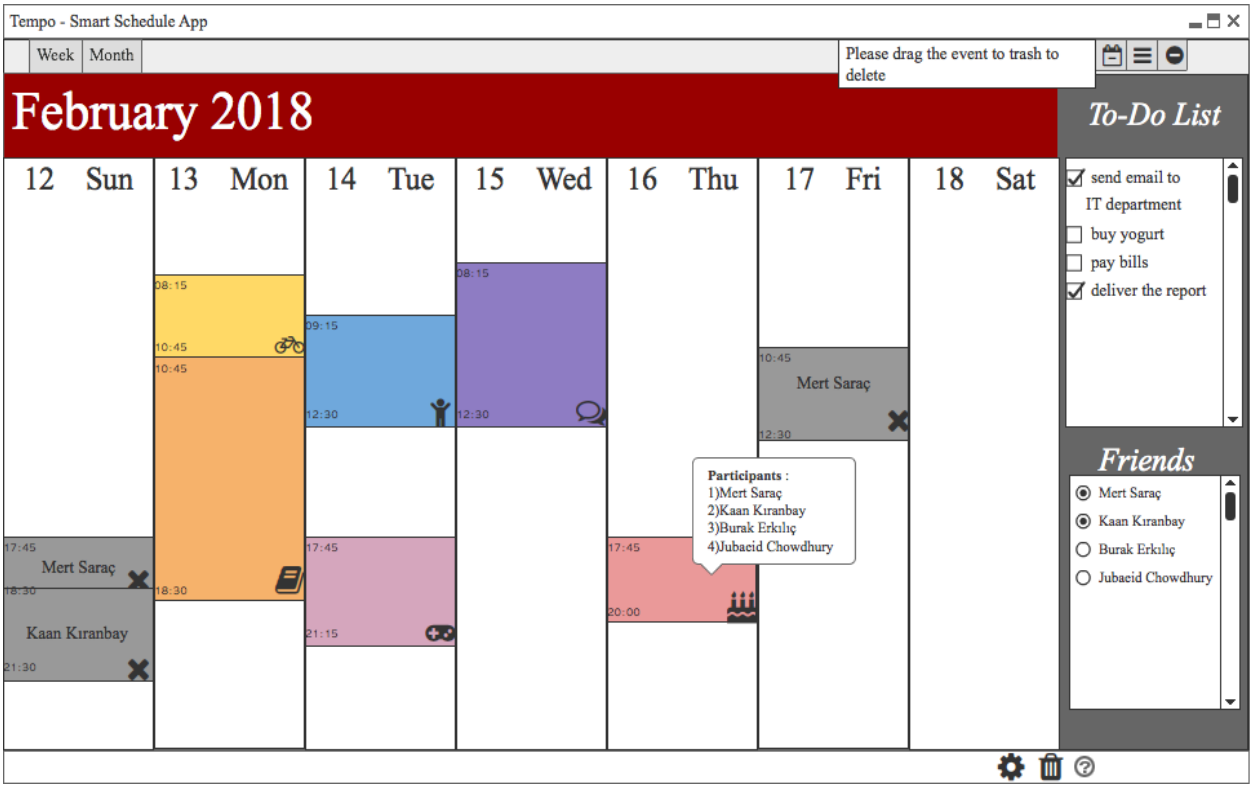
After clicking the “Week” button, user can access this page that contains detailed information about activities like participants of an activity and time period of activity. To see participants, user uses just right click of mouse. To add an activity, user uses add activity button.

5.2.4 Add Event Window



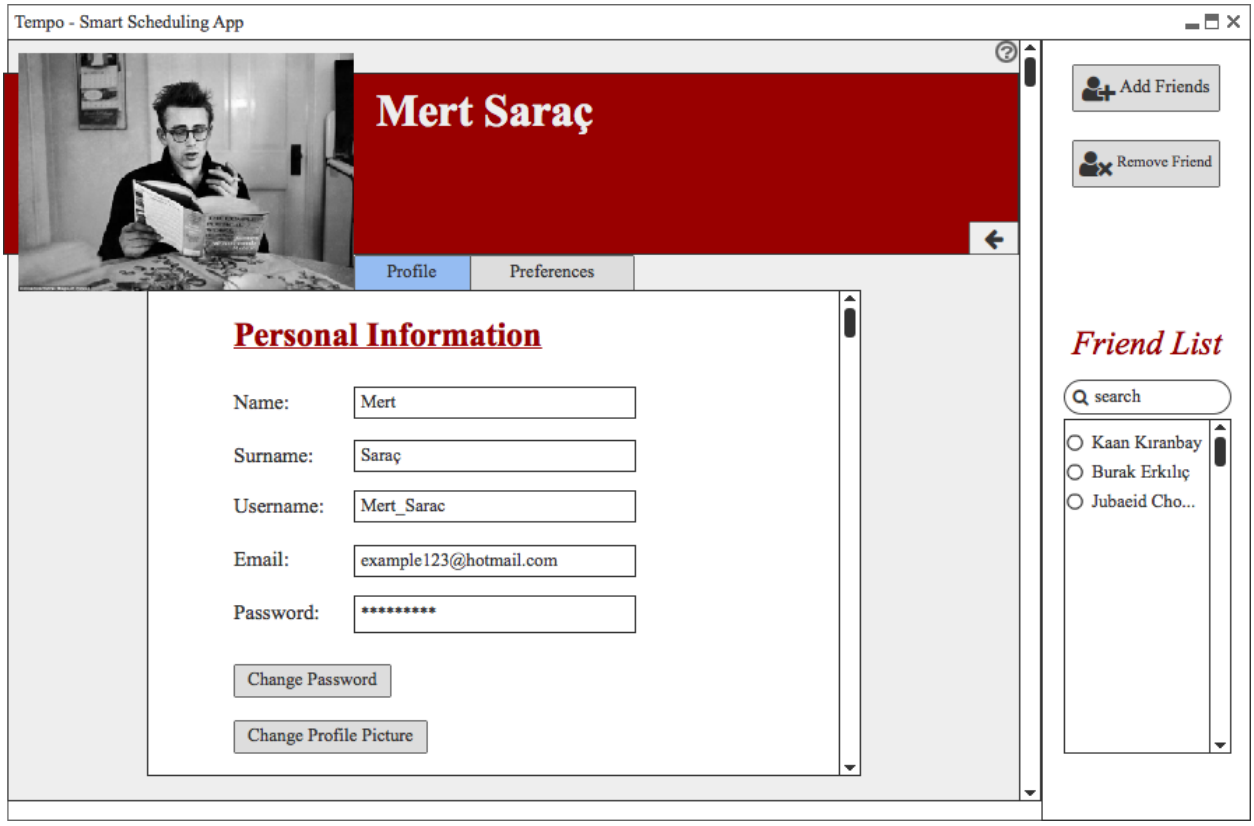
When “Add Activity Button” is pressed, user encounters with a new window which contains many options for describing the event. User can select whether it is SmartEvent or Personal Event, if it is SmartEvent, user can select participants otherwise, this part is disabled, and choose whether it is permanent or temporary event. If user wants many stickers are available for event view on calendar. Finally, user needs to specify time period on calendar.

5.2.5 Delete Event



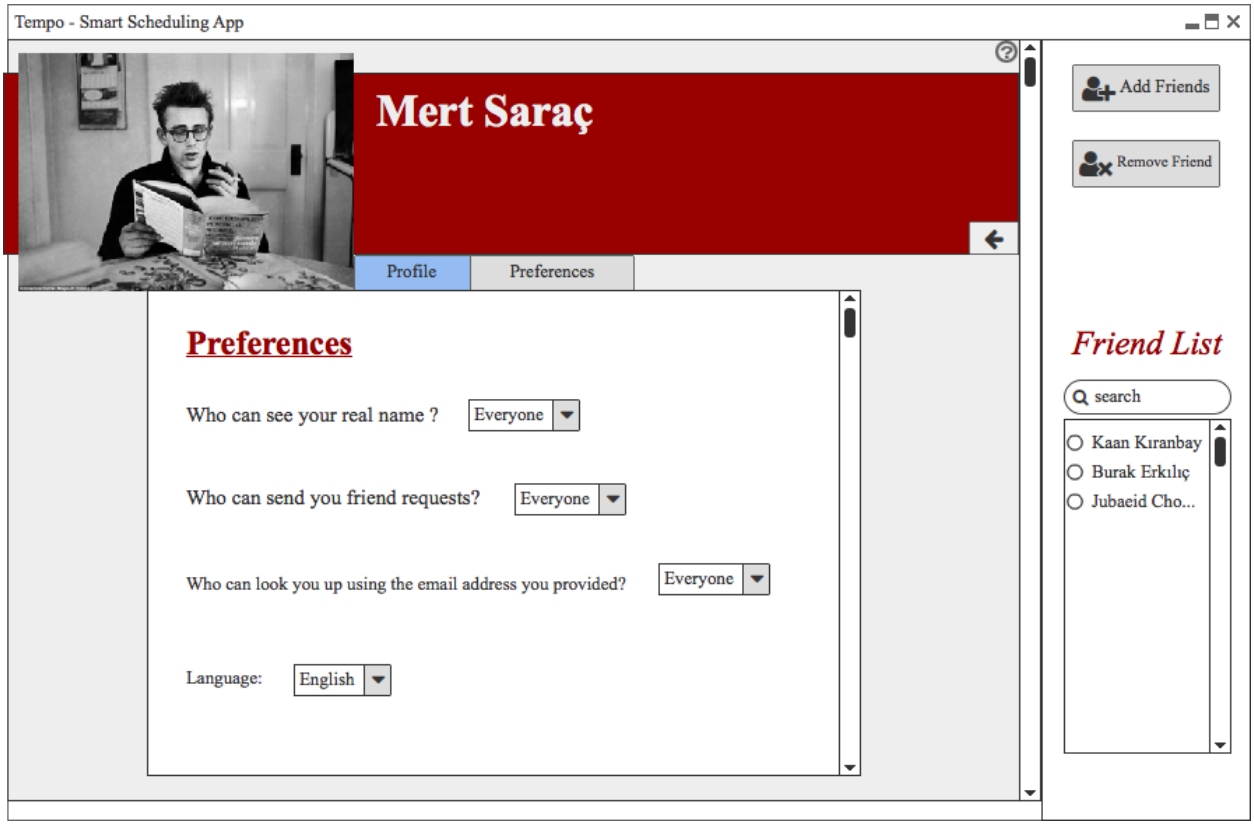
If user wants to delete a event, there is a “Delete Event” button. After clicking it, it is prompted that “Please drag the event to trash to delete.”. By dragging a event to trash, it will be deleted.

5.2.6 Profile



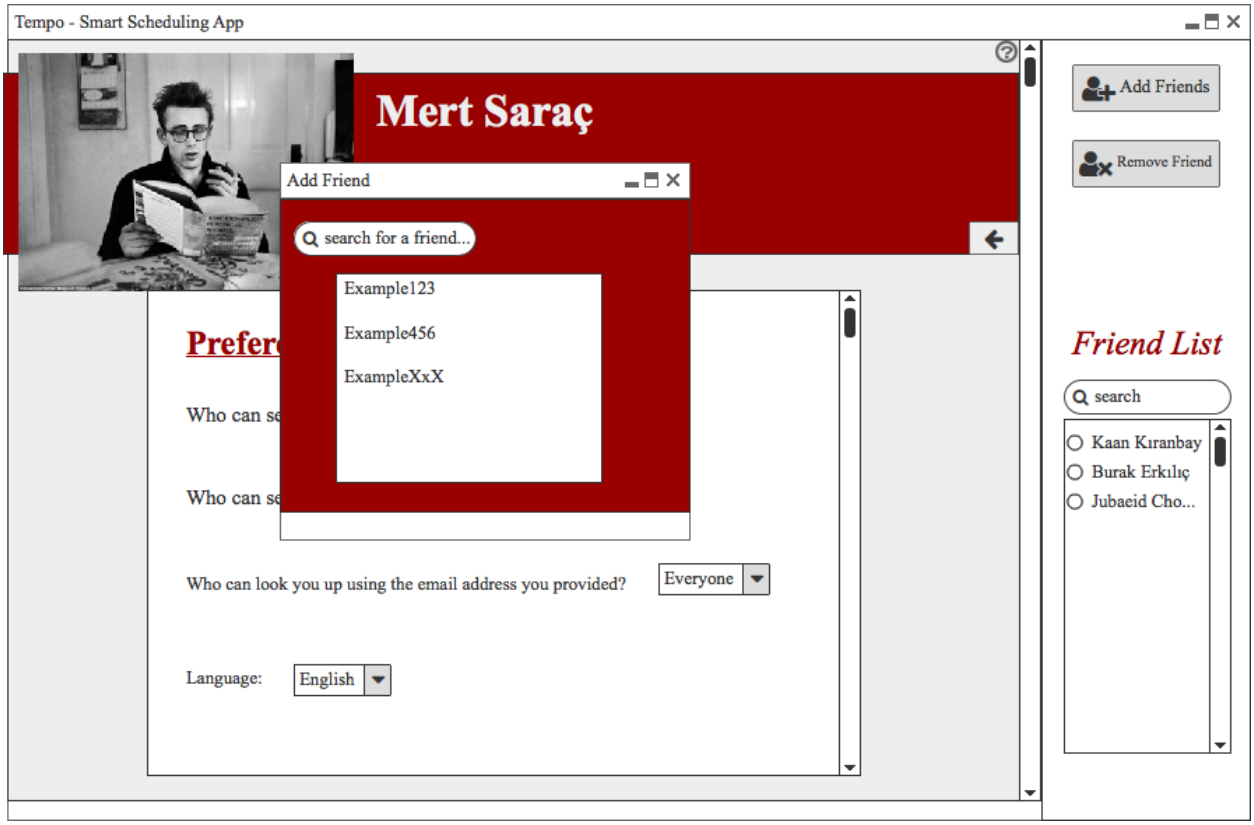
If user clicks menu button on the calendar page, Profile and Preferences pages show up. Profile page contains some personal information about user and they can be changed by user.

5.2.7 Preferences



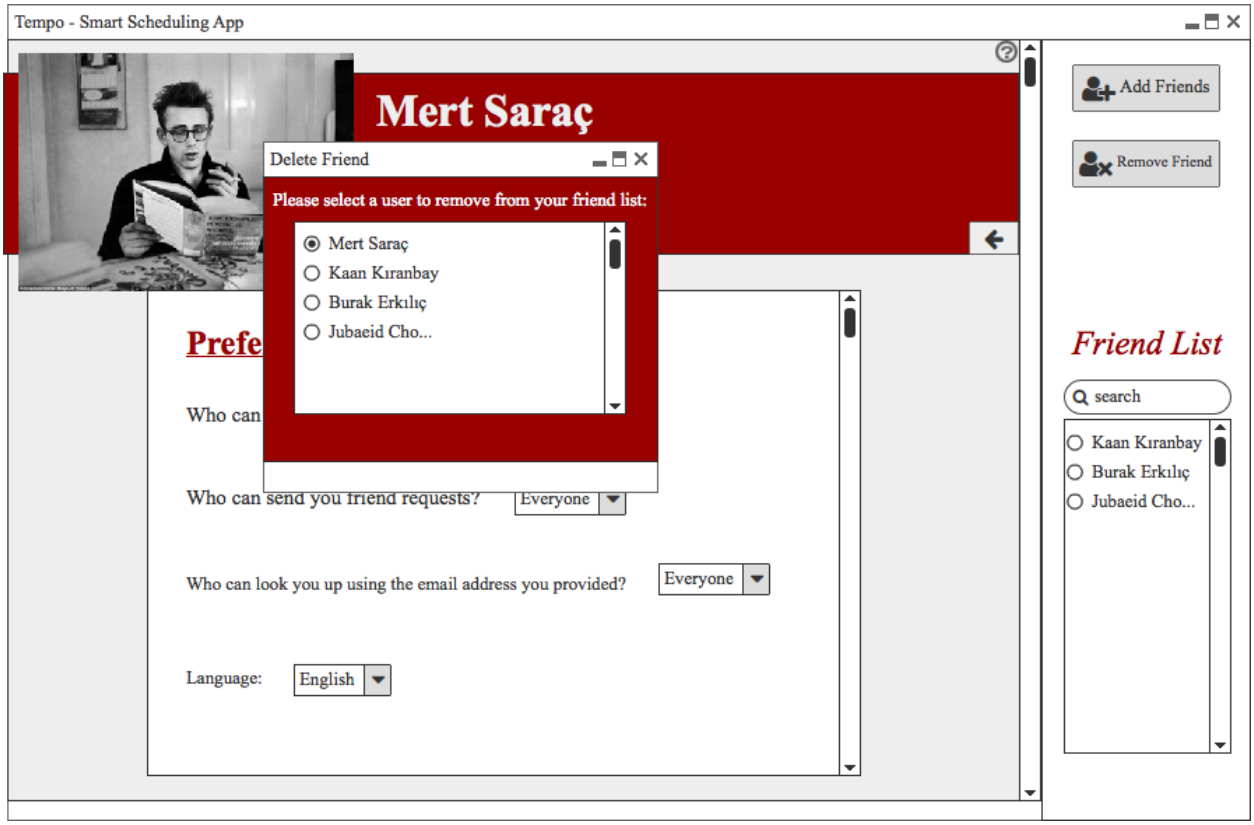
Preferences pages contains some settings like language and privacy. Next to this page, there are 2 buttons that are “Add Friends” and “Remove Friend”.

5.2.8 Add Friend



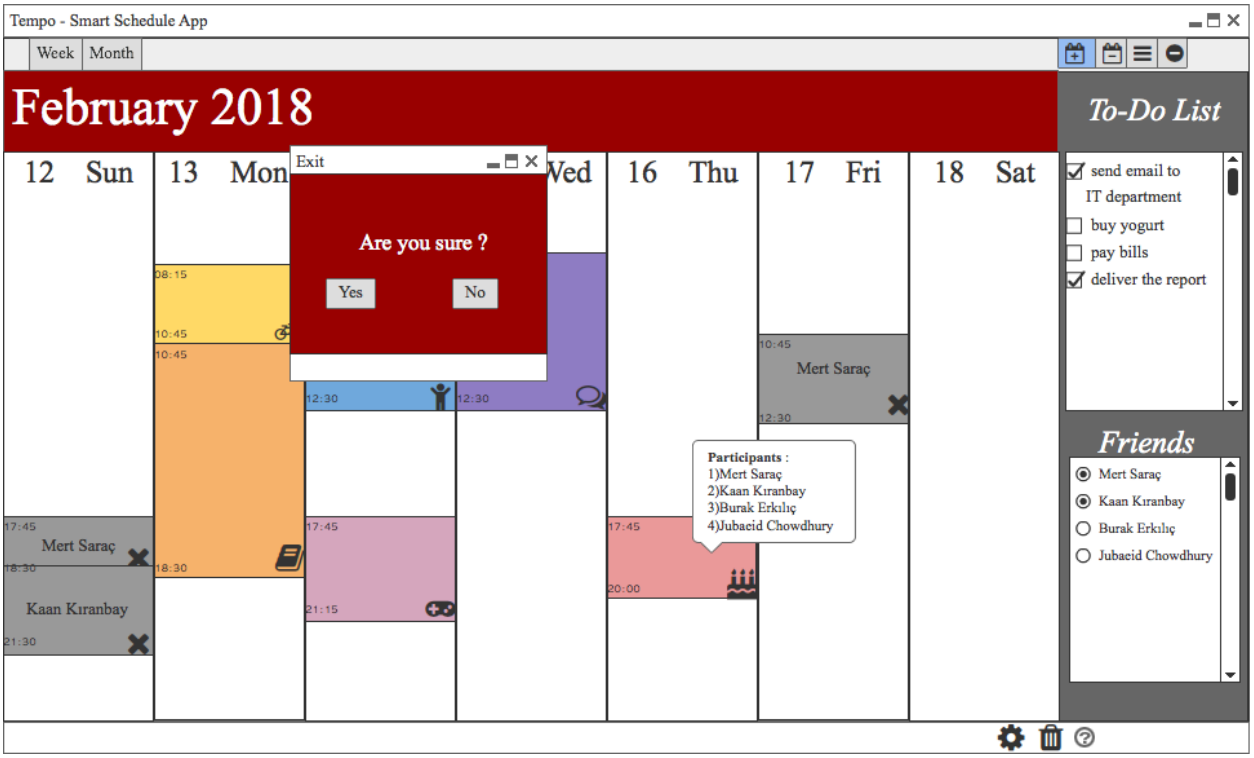
By using "Add Friends" button, user can search for a friend by entering other users' username or their real name.

5.2.9 Delete Friend



If user click “Remove Friend” button, a window that contains friend list and prompts “Please select a user to remove from your friend list.”. By selecting them with radio buttons, user can select friends. User can also go back to main menu by clicking button with left arrow.

5.2.10 Exit Window



If user wants to exit, there is a exit button that is next to menu button. After clicking it, a window pops up and asks for it. If user click “Yes”, program is terminated but if “No” button is pressed, main menu shows up again.

6. References

[1] Object-Oriented Software Engineering, Using UML, Patterns, and Java, 2nd Edition, by Bernd Bruegge and Allen H. Dutoit, Prentice-Hall, 2004, ISBN: 0-13-047110-0.