Software Design Document

Uplan

Richard Hartnett, Joe Menzie, Zach Matthews, Steve Lucey

1. Introduction

The software product we are developing is a self-planning planner called UPlan. How it works is first the user will create an account. When creating this account they will fill in certain preferences such as sleep schedule to help the application schedule efficiently based on when the user goes to sleep. The next thing the user will have to do is add events. These events could be classes or other events going on in a specific week. Next the user will enter their assignments as they come in. When entering these assignments they will be asked to enter how long they believe it is going to take, when the due date is, and whether it is a high priority assignment or not. UPlan's algorithm will take these assignments, compare them to the classes or other events the user has for the day, and plan a study schedule based on the free time available. This study schedule will include which assignments to work on that day and for how long. The idea is, the user will use this generated schedule and roughly follow it to keep them on track for their day to day and long term assignments.

The user will be able to edit these events and assignments to alter the time they occur or other settings related to them. One very important aspect the user will need to know for the application to work correctly is at the end of each day or couple days, they will need to go back and update how much time they have left for each assignment. This needs to be done because a lot of time when someone starts an assignment they predict it will take a certain amount of hours but in reality to will take maybe half as long or twice as long. To factor in for this inaccuracy, the user will need to continuously update how far they have come and how much more they have to do.

2. Design Considerations

This section will cover the assumptions and constraints involved with the making of our UPlan program. This will also discuss the correct environment that the system can operate in.

2.1 Assumptions

In order for our system to run to the best of it abilities we need to assume that people will follow their created schedule moderately well or readjust their schedule as it changes. If they do not go to class consistently or vary their sleep schedule, then the program will not be correctly organizing the client's assignments. In order for the program to run efficiently we also need to assume that the client will keep giving the program feedback and refreshing when they want to change the schedule they have. If the client does not refresh the program, then it will not correctly calibrate the work needed to be done on each assignment.

2.2 Constraints

One possible constraint is the scope of what the project can cover. Clients may expect the program to take in and organize more assignments than it can handle at once. The program can only work assignments into the time slots open between the class schedule and sleep schedule of the assignment.

2.3 System Environment

The environment needed to run our program is a Unix environment. This will be compatible with an Unix based system and some Mac OS systems.

3. Architectural Design

3.1 Overview

The architectural design of the system consists of the GUI, the Client, the file system, and the server. The responsibilities of each component are as follows:

> The main purpose of the GUI is to interact with the user. It displays the data in a calendar format to the user and allows the user to interact with the program. Buttons on the display bring up a page where the user can enter, alter or delete data from his calendar.
>
> The Java Client has the main purpose of communication with the GUI and the server. It takes requests from the GUI and sends them through to the server. When it receives data from the server, the Client formats it to a readable format for the GUI to present to the user.
>
> The database is the primary storage device for the user's information. All relevant information will be sent to the Client upon startup of the program and updated as information is created or modified.
>
> The server receives connections from the client and provides it with information from the database. In addition, the server is where the scheduling algorithm does its work, creating a workplan that it sends to the client and updates in the database.

3.2 Rationale

We are using a GUI instead of a text-based UI because it is essential for the user to be able to see his classes and work blocks on a calendar instead of just in list format. The goal of UPlan is to organize someone's free time who has trouble doing so, and this would be difficult to do without a solid visual representation of their schedule.

As of now, we are using a Java-based GUI and Client since we are most familiar with Java and C, but would much prefer the use of an object-oriented language.

The server is used to run the scheduling algorithm so the local machine doesn't have to do all of the computing work and so that the downloadable program size doesn't get too large. The database is accessed through the server so the local machine doesn't have to store the entire calendar's history using up more space than necessary on the local machine.

3.3 Conceptual (or Logical) View

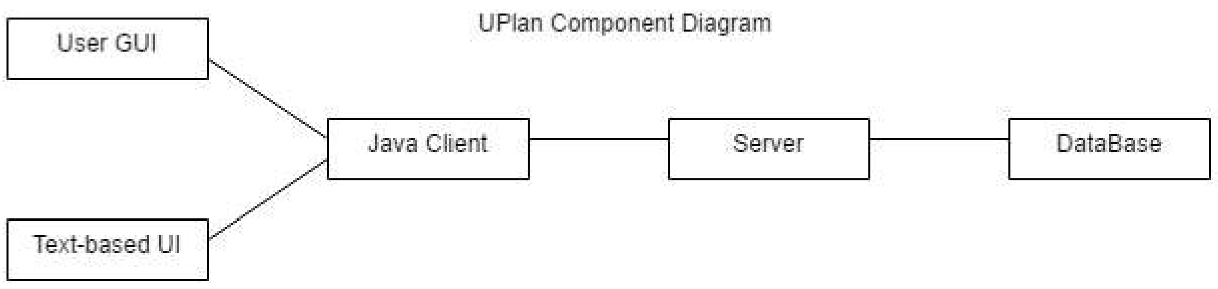

Figure 3.3.1: Component Diagram

4. Low Level Design

This section consists of a description of the low level design of the program's layout. This design is described through class diagrams, sequence diagrams, and state diagrams.

4.1 Class Diagram

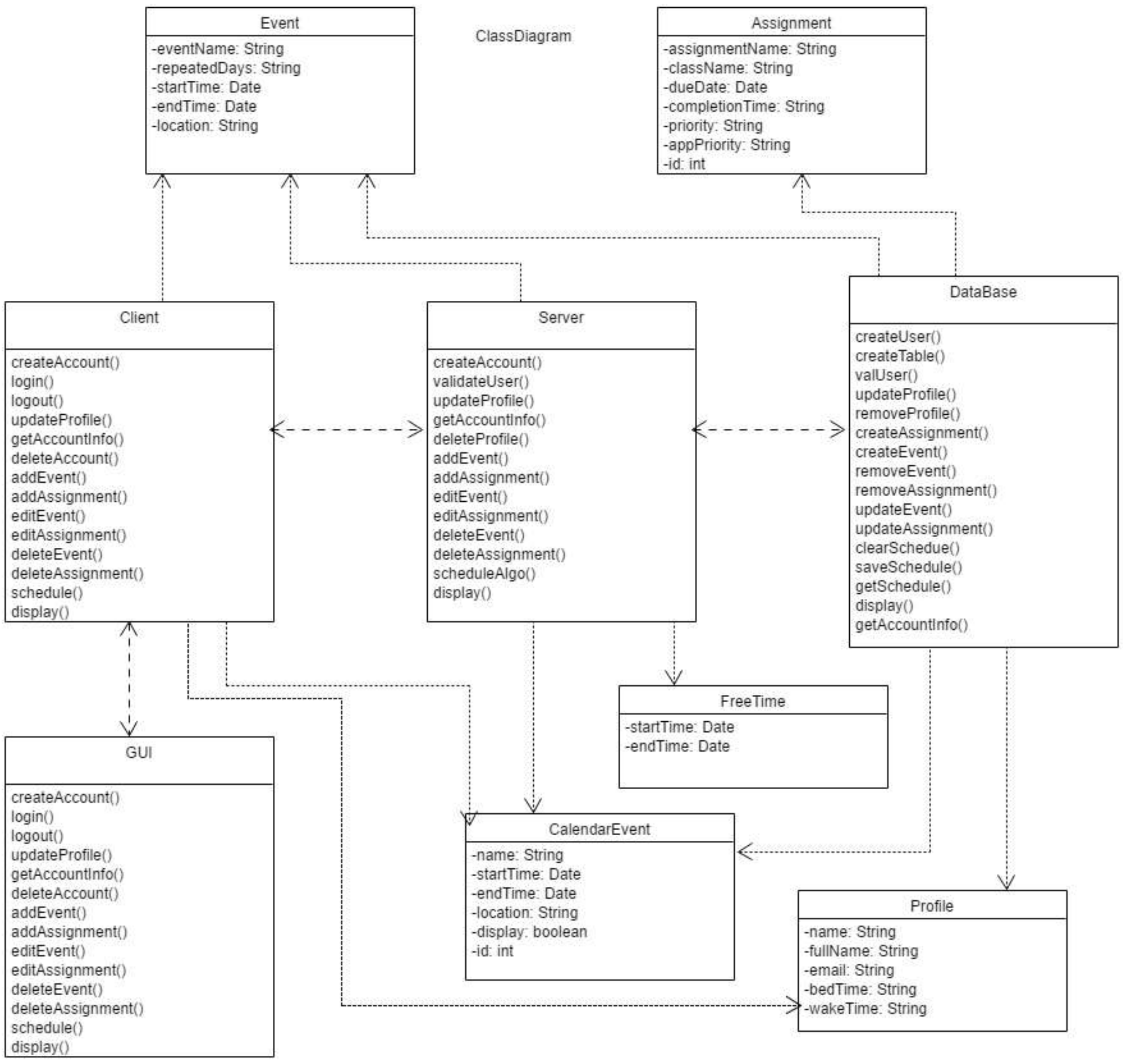

Figure 4.1.1: Class diagram showing all class and associations

4.2 Sequence Diagram

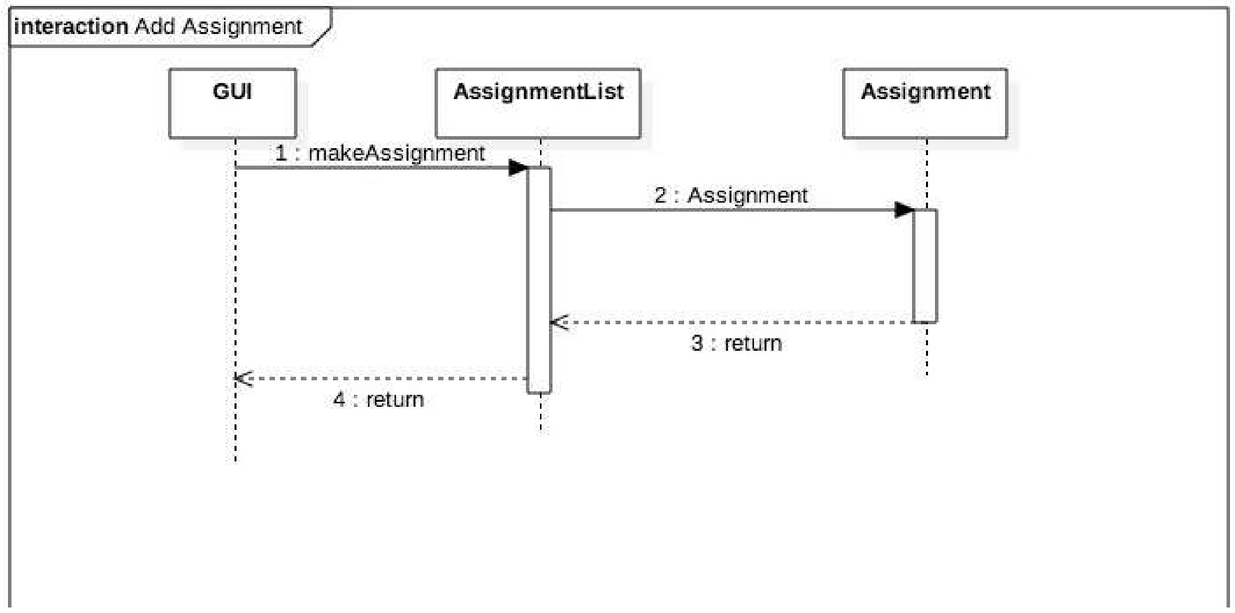

Figure 4.2.1: Sequence diagram for adding an assignment

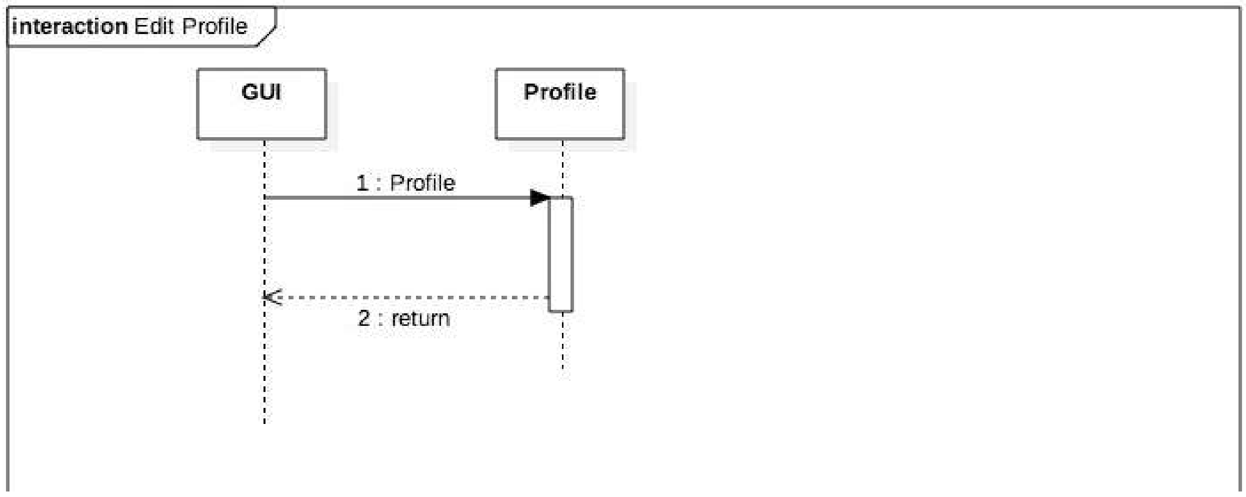

Figure 4.2.2: Sequence diagram for editing user profile

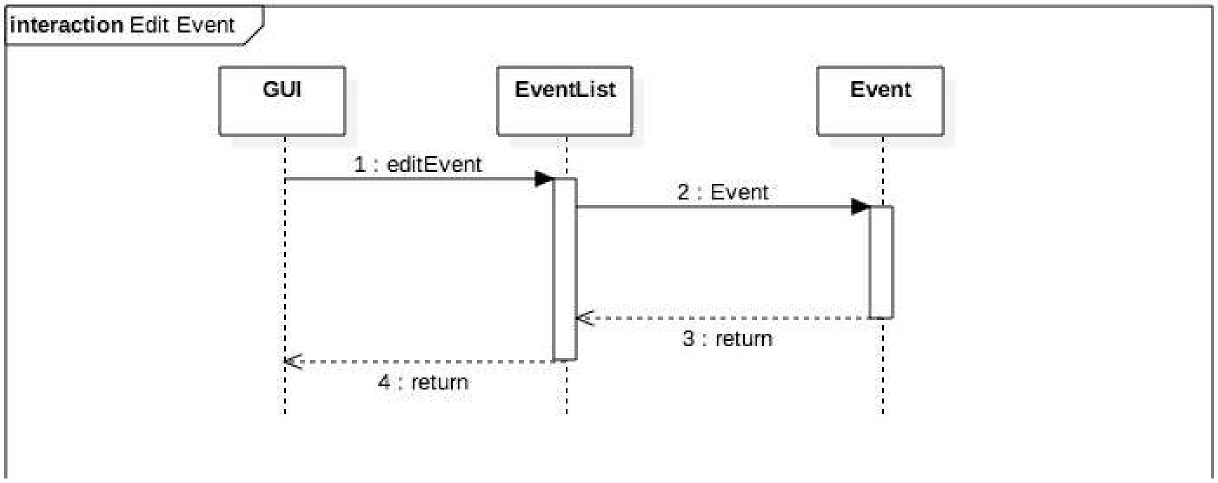

Figure 4.2.3: Sequence diagram for editing an event

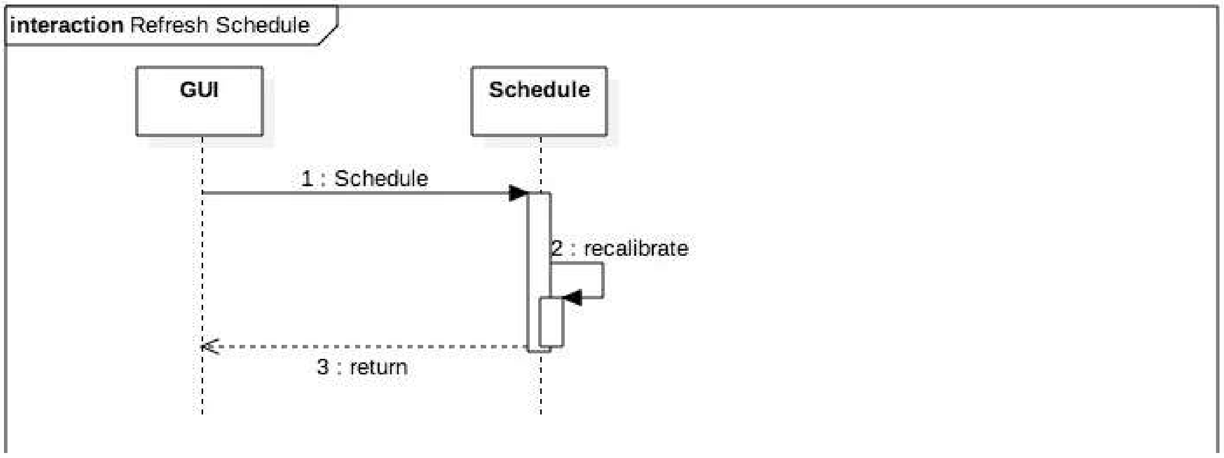

Figure 4.2.4: Sequence diagram for manually refreshing the schedule

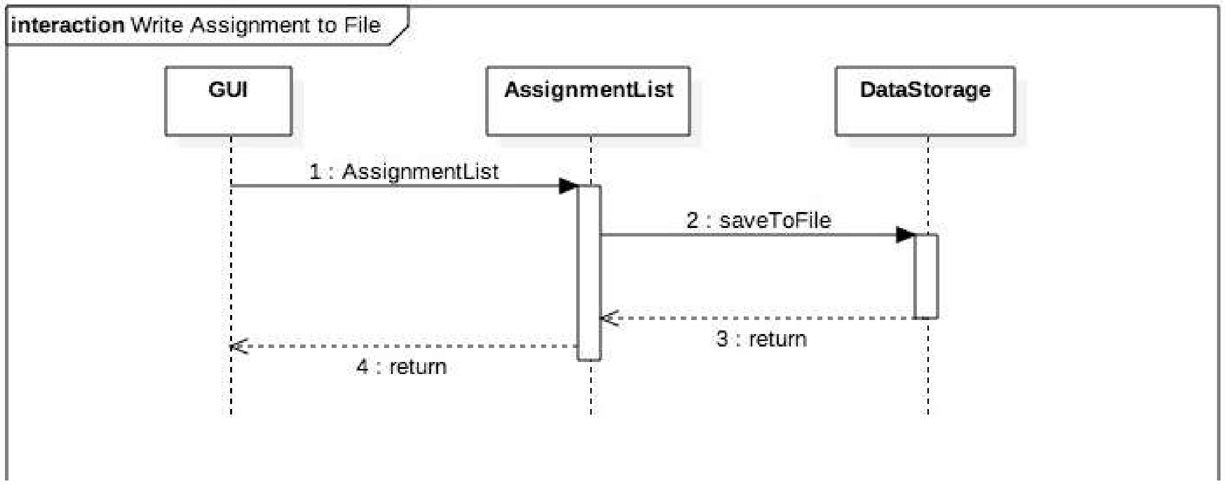

Figure 4.2.5: Sequence diagram for writing an assignment to file

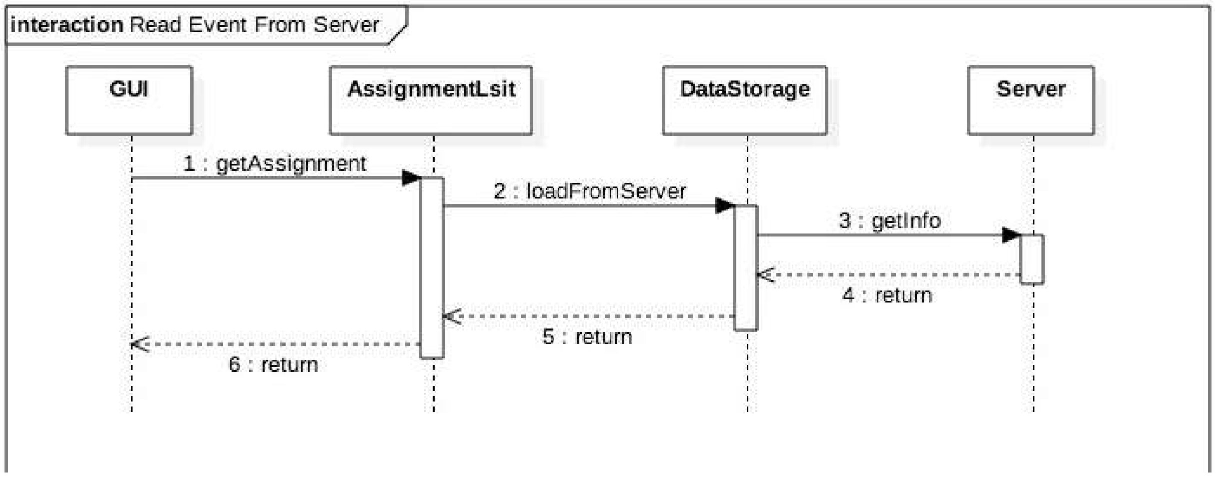

Figure 4.2.6: Sequence Diagram of reading an event from the server

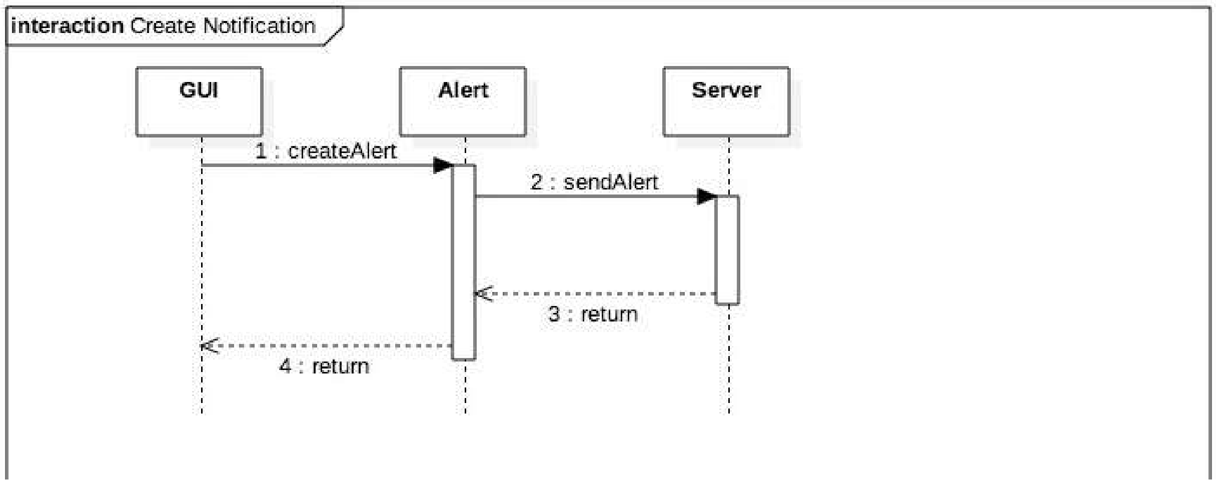

Figure 4.2.7: Sequence diagram of creating a notification

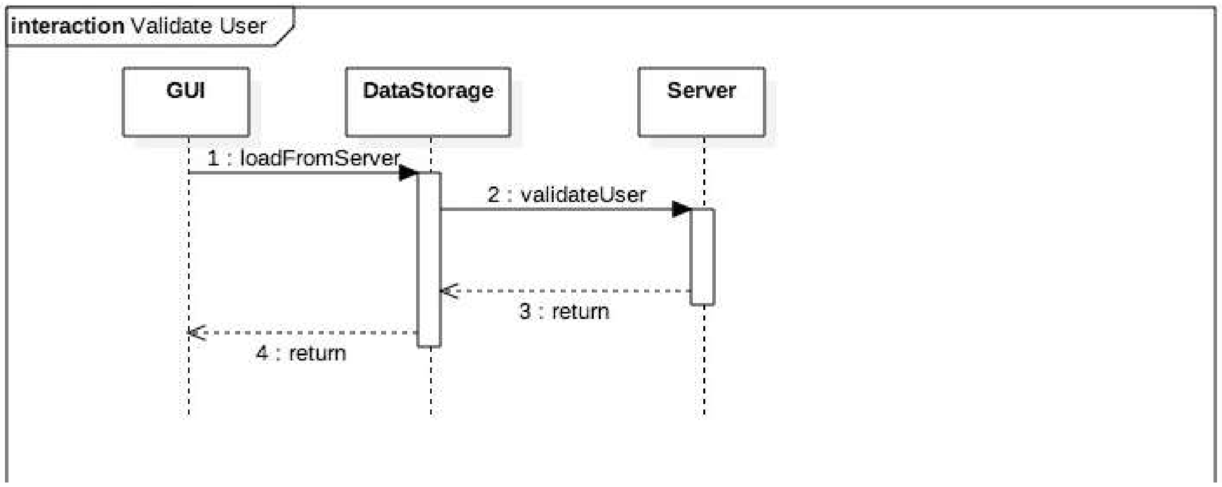

Figure 4.2.8: Sequence diagram of validating a user

4.3 State Diagram

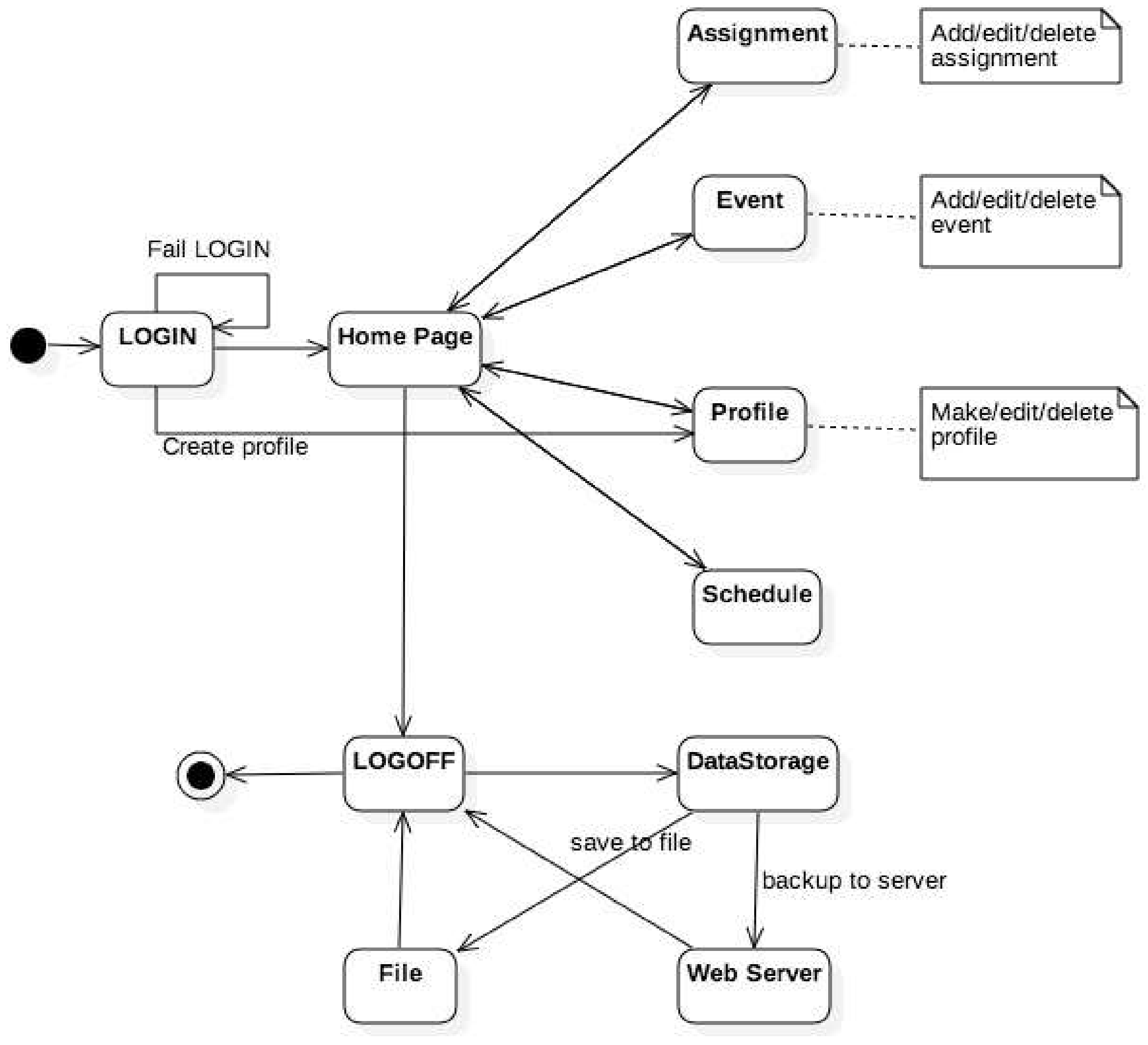

Figure 4.3.1: State diagram showing all possible states for program

5. User Interface Design

The user interface will be in the form of a GUI. The data will be represented in the form of a calendar as events/activities and assignments. On this calendar, there will be options to input, modify or delete data via buttons.

5.1 Login

When the user first enters the program, they will be taken to a screen where they are presented with text boxes for their username/email and their password, then a login button. The login button will not do anything unless the username and password is correct. There is also a create account button that leads to the create account page.

5.2 Create Account

When the user wishes to create an account, this page will prompt them for their name, email, and a username with a corresponding password. The user will also be able to select a preferred bedtime and wake up time from a drop down menu from 7:00am to 11:30pm, at any 30 minute interval. Then the user will click the create account button that will make their account(as long as no other user already has that username) and takes them to the homepage. There is also a cancel button that will lead the user back to the login page.

5.2 Home Page

After login the user is brought to their homepage that consists of a calendar that displays their events and study schedule and also includes a number of buttons. These buttons allow the user to interactively modify their schedule by editing events, adding assignments, etc.

5.3 Add Event

If the user chooses to input a new event, they will click on that button labeled “Add Event” which will bring them to a new window. This new window will include multiple text boxes for the user to enter the information about the event. Once the user has completed filling out the boxes, there will be a save button to return to the calendar. There will also be a cancel button which will erase any information they entered and will return them to the homepage. To display the updated schedule with the new event the user will have to click the refresh schedule button.

5.4 Add Assignment

The user interface for the add assignment action will be very similar to that of the add event/class but the user will be prompted for different information, such as name of assignment, how long they believe the assignment will take them, etc.

5.5 Edit Assignment, Event/Class

The interface for the user editing either an assignment or event/class will be the same as their respective add interfaces. The only difference will be the information previously recorded will be already inputted into the textboxes. The user will have the ability to erase this data and enter new information in and save it.

5.6 Edit Profile

The GUI for the users personal information show the information that the user put in when they created their account, except for their password. There is also two drop down boxes that will allow the users the change their preferred wake up time and bedtime.

5.7 Display Events/Assignments

This page pops up a display window that will list the users assignments and events,(depending on which button they clicked) and show all of the characteristics of each event or assignment. There will also be a close button that will return the user to the homepage.