Project: **Smart Home Simulator**

Project Report

STUDENT NAMES:

Gregory Tucker - 40092432
Aymen Metallaoui - 40057294
Tyler Znjog - 40005987
Adrien Kamran - 40095393
DeepKumar Patel - 40096716
Minh-Tam Do - 40095639

Tutorial Section HC

Github Repository: **https://github.com/gregtuc/SmartHomeSimulator**

TABLE OF CONTENTS

1.0. Introduction

The rapid development of automation-based products and solutions provides the opportunity to implement technology into houses to increase efficiency. By automating and modernizing systems inside of the everyday house like heating and security, the potential to decrease energy bills and increase security provides a valuable business opportunity. The creation of these software systems, however, is not necessarily easy. The completion of a smart home system requires testing and simulation to perfect the models and software used. For this reason, it is necessary that smart home simulators exist to assist in the development of real-world systems since using a simulation will allow for a more cost-effective approach to testing software.

2.0. Deliverable 1 Scope

The primary objective of the first deliverable is to create the basic external features of a Smart Home Simulator application. Since strong project planning is a mandatory component of the submission, the planning contains five distinct project goals: finding and selecting technologies to be used, learning how to use the selected technologies, division of tasks according to strengths, implementation of the code, and project review. The end result should be a product that satisfies the client's need of being able to change simulation context and parameter variables and be given visual feedback.

A functional software product should be created that contains a basic user-interface, a view of the inputted house text file, a modifiable simulation clock, the ability to modify user profiles, the ability to modify outside temperature, and the ability to set a house location for the active user or other people objects. The software product should be modular and easily used in a plug-and-play fashion during future deliveries when the actual simulation aspect of the application is developed.

The source code of the application must be created in Java, the system architecture must be of the Model-View-Controller (MVC) format, and house-layouts must be stored as a text file and be displayed two-dimensionally. Deliverable requirements should be completed by October 20, 2020. It is assumed that users have Java installed on Windows/Linux/MacOS, have sufficient storage to install the application, and are capable of installing and understanding the program themselves.

3.0. Functional Requirements (Use Cases).

3.1. Create a Profile

Use case name	Create a Profile.
Level	User-level
Brief description	Profiles help to store user preferences and separate permissions. The user must be able to create a profile and specify what type of account they are creating.
Preconditions	The User has accessed the main dashboard of the application.
Triggering event	The “Profile” button in the simulation tab was pressed.
Main flow	1. The system opens a tabbed Profile window. 2. The User selects the tab labelled “Create Profile”. 3. The User enters the information and submits the form. 4. The system stores the new profile in the active session and signs in the new profile. 5. The Profile window closes.
Extensions	During step 3 of the main flow the user submits a blank form: 1. An alert box appears informing the user that they cannot submit a blank form. 2. The User presses the confirm button in the alert box. 3. Continue from step 3 of the main flow.
Postconditions	The new profile is saved. The new profile is accessible in the “Edit Profiles” and “Login to Existing Profiles” tabs. The new profile is signed in.

3.2. Edit Existing Profiles

Use case name	Edit Existing Profiles.
Level	User-level
Brief description	It is mandatory that profiles be modifiable. This includes the ability to delete a profile altogether. The ability to delete a profile is the only ability in terms of editing for the first delivery, as other features have not yet been developed.
Preconditions	The User has accessed the main dashboard of the application.
Triggering event	The “Profile” button in the simulation tab was pressed.
Main flow	1. The system opens a tabbed Profile window. 2. The User selects the tab labelled “Edit Existing Profile”. 3. The User selects the profile to delete from the drop-down and submits the form. 4. The system removes the profile from storage and closes the window.
Extensions	During step 3 of the main flow, the user deletes the signed-in profile: 1. The system sets the active user to null. 2. Continue from step 4 of the main flow.
Postconditions	The deleted profile no longer exists in the “Edit Profiles” and “Login to Existing Profiles” tabs.

3.3. Login to an Existing Profile

Use case name	Login to an Existing Profile.
Level	User-level
Brief description	Users can store modified settings and permissions on their profile. It is mandatory that users be able to return to their created profiles after signing out.
Preconditions	The User has accessed the main dashboard of the application.
Triggering event	The “Profile” button in the simulation tab was pressed.
Main flow	1. The system opens a tabbed Profile window. 2. The User selects the tab labelled “Login To Existing Profile”. 3. The User selects the profile to login from the drop-down and submits the form. 4. The system finds the profile in storage and signs it in 5. The Profile window closes.
Extensions	During step 3 of the main flow the user selects an already signed-in profile: 1. An alert box appears that says the selected user is already signed in. 2. The User closes the alert box. 3. Continue from step 5 of the main flow.
Postconditions	The name of the signed-in user is shown at the top of the Simulation box on the main dashboard.

3.4. Modify the Time of the Simulation

Use case name	Modify the Date and Time
Level	User-level
Brief description	Simulation variables such as temperature require a time parameter to function properly. The time parameter acts as a reference point for the various methods to function properly. To give users the ability to properly test the simulator, they need to have the ability to start, stop, and modify the system clock.
Preconditions	The User has accessed the main dashboard of the application.
Triggering event	The “Time” button in the simulation tab was pressed.
Main flow	1. The system opens a window with fields for specifying the time to set the Clock to. 2. The User enters the information and submits the form. 3. The system updates the current time and closes the window.
Extensions	During step 2 of the main flow the user entered an invalid time: 1. An alert box appears informing the user that they entered an invalid time. 2. The user closes the alert box. 3. Continue from step 2 of the main flow.
Postconditions	The time entered by the user is shown at the top of the Simulation box on the main dashboard.

3.5. Upload a House Layout File.

Use case name	Upload a House Layout file
Level	User-level
Brief description	The user can upload a wanted House Layout into the system
Preconditions	1. The User is in possession of a House Layout in text file displayable in 2 Dimensions. 2. The user has access To a House layout Tab in the Dashboard.
Triggering event	The user presses the “Upload Layout” button.
Main flow	1.The system opens a new window that prompts the user to write down the path of the wanted layout as a text file. 2. The user writes down the path of the layout file in the available space and presses “Confirm”. 3. The window closes. 4. The System loads the wanted Layout into the system and generates it.
Extensions	The user enters a non-valid house layout (not a text file) path; 1. An alert box appears and informs the user to enter a valid file path. 2. The user closes the alert box. 3. The main flow continues at step 2.
Postconditions	The wanted House layout is generated and is now shown in the House View tab.

3.6. Change the Location of Users

Use case name	Change the Location of the Users
Level	User-level
Brief description	Users can change their location during the simulation.
Preconditions	User simulation must be on and the user must be logged-in
Triggering event	The “Location” button in the simulation tab was pressed
Main flow	1. The system opens a tabbed Location window 2. The User selects the tab labelled “Location of currently logged-in user”. 3. The User enters the information and submits the form. 4. The system stores the new location in the active session. 5. The Location window closes
Extensions	During step 3 of the main flow the user submits a blank form. 1. An alert box appears informing the user that they cannot submit a blank form. 2. The User presses the confirm button in the alert box. 3. Continue from step 3 of the main flow.
Postconditions	The location of the signed-in user is shown at the top of the simulation box on the main dashboard.

3.7. Modify the Exterior Temperature.

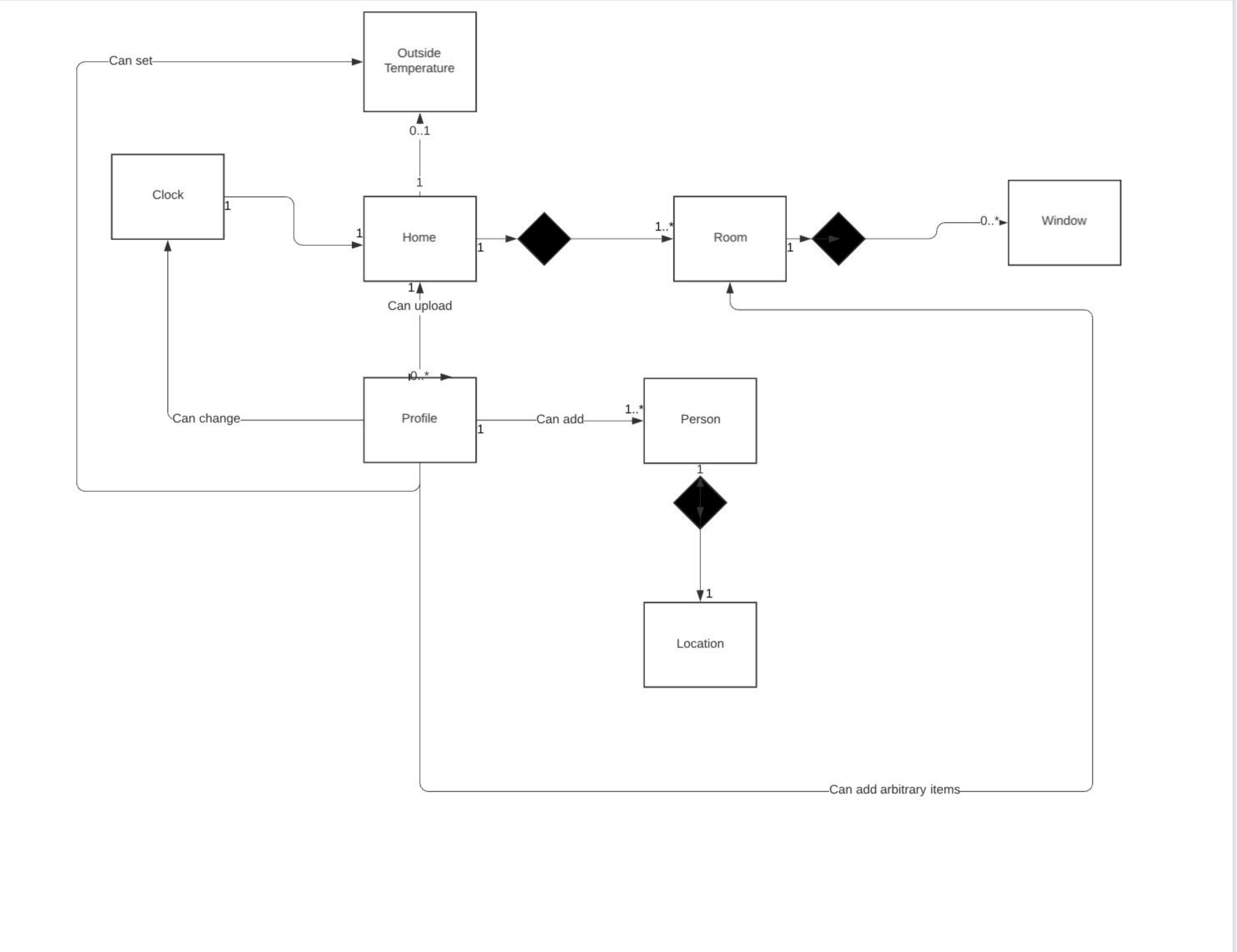
Use case name	Modify the Exterior temperature
Level	User-level
Brief description	The user can change the temperature outside the house which is an important factor that influences the inside temperature and the behavior of users.
Preconditions	The user has opened the application and has access to the main dashboard and locates the simulation tab.
Triggering event	The user presses the “Temp” button on the simulation tab.
Main flow	1.The system opens a new window that prompts the user to write down a desired temperature. 2. The user writes down the desired temperature in the available space and presses “Confirm”. 3. The window closes. 4. The System has updated the desired temperature and the prompt tab is closed.
Extensions	The user enters a blank value of temperature; 1. An alert box appears and informs the user to enter a valid temperature 2. The user closes the alert window. 3. The main flow continues at step 2.
Postconditions	The outside temperature updates as desired and is indicated in the simulation tab’s info section.

3.8. Lock and Unlock the Movement of Windows by Adding Arbitrary Objects.

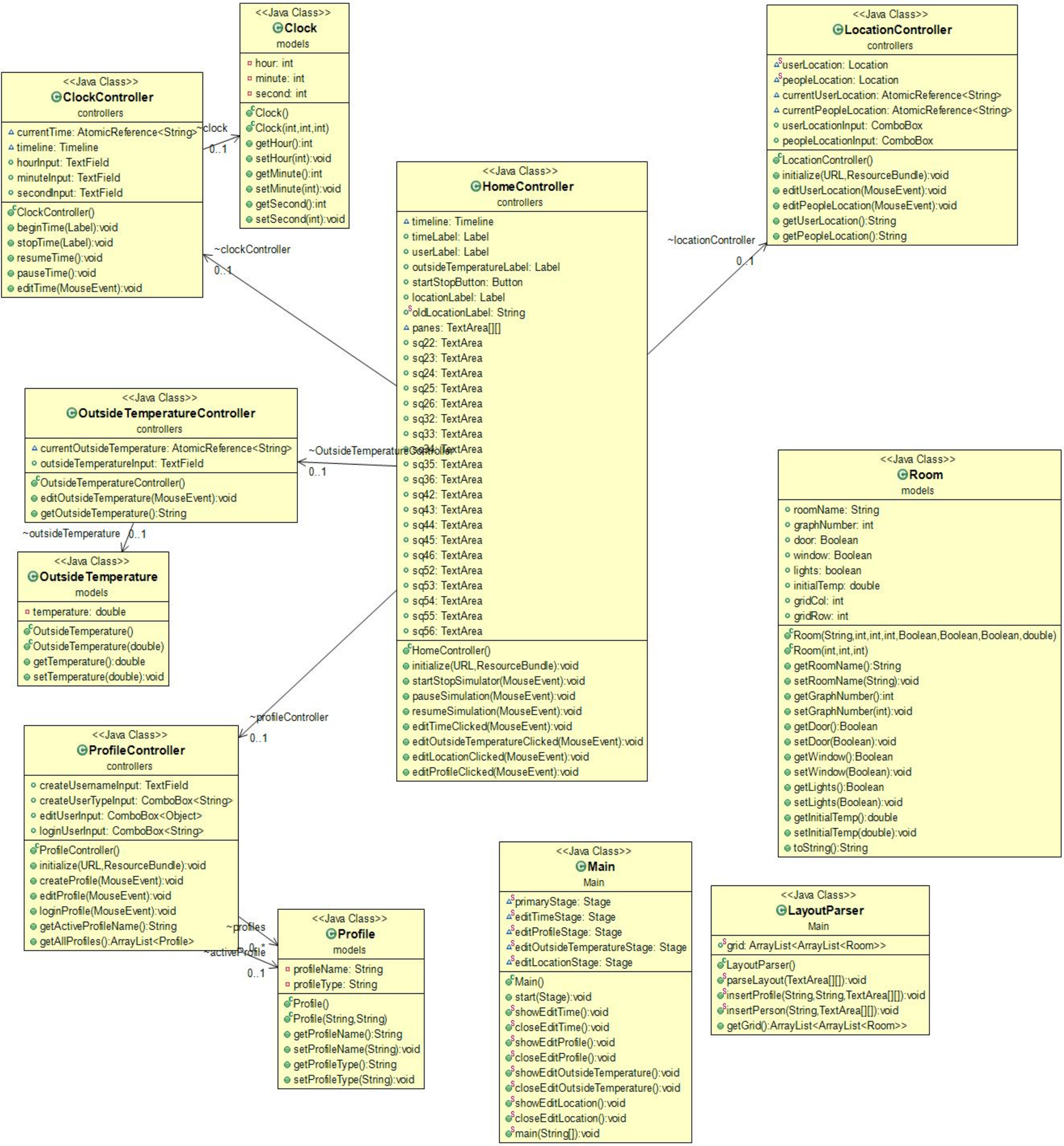
Use case name	Block window movement by putting an arbitrary object.
Level	User-level
Brief description	The user blocks/unblock window movement by adding/ removing arbitrary objects .
Preconditions	The user is in a select room and has access to the Items tab.
Triggering event	The user presses “add Item” button .
Main flow	1.The system opens a new window that asks the user an item name and if they are placing it in front of a window . 2. The user writes down the item’s name and chooses to put it in front of the room’s window or not . 3. The user presses “confirm”. 4. the window closes. 5. The item is added.
Extensions	The user has already entered an item value in front of the window 1. An alert box appears informing the user that an item is already in front of the window. 2. The user presses the confirm button in the alert box. 3. Continue from step 2 of the main flow. The item is now added in the select room and location.
Postconditions	The item is now added in the select room and location, in the item tab.

4.0. Diagrams

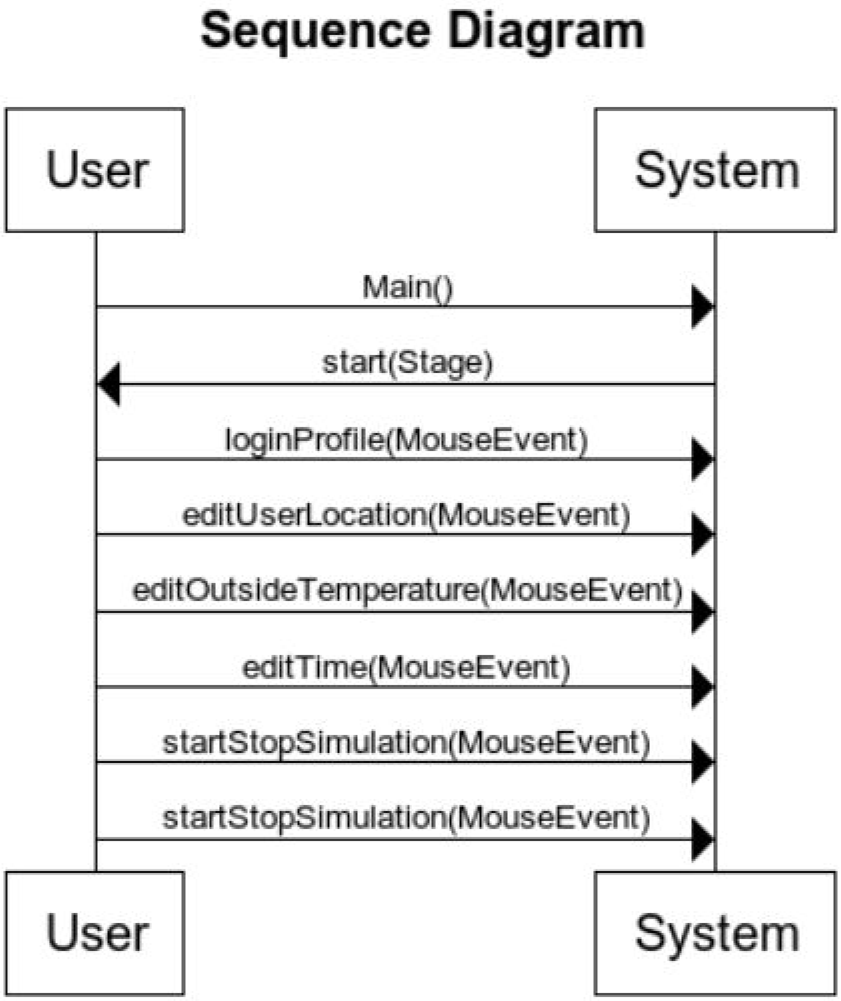
4.1. Domain Model



4.2. Class Diagram



4.3. Sequence Diagram


Sequence Diagram
User
System
Main()
start(Stage)
loginProfile(MouseEvent)
editUserLocation(MouseEvent)
editOutsideTemperature(MouseEvent)
editTime(MouseEvent)
startStopSimulation(MouseEvent)
startStopSimulation(MouseEvent)
User
System