

Software Engineering - Spring 2019

Report 1

GlassHome

Group 12

Harshil Parekh, Adarsh Gogineni, Shivum Mehta, Shaan Parikh, Andy Guo,

Avi Patel, Nathan Silva, Parth Patel, Kyle Abed

February 24, 2019

sites.google.com/scarletmail.rutgers.edu/softwareengineeringspring2019/

14:332:351 Software Engineering Report 1
Department of Electrical and Computer Engineering
*The State University of New Jersey,
Rutgers*

Responsibility	Harshil Parekh	Adarsh Gogineni	Shivum Mehta	Shaan Parikh	Andy Guo	Avi Patel	Nathan Silva	Parth Patel	Kyle Abed
Project Management	40%	12%	12%	9%	9%	12%	3%		3%
Section 1: Problem Statement					25%		25%	25%	25%
Section 1: Glossary of Terms	100%								
Section 2: Enumerated Functional Requirements		33%	33%			33%			
Section 2: Enumerated Nonfunctional Requirements		33%	33%			33%			
Section 2: On-Screen Appearance Requirements				100%					
Section 3a: Stakeholders					100%				
Section 3b: Actors and Terms		100%							
Section 3c: Use Cases: Casual Description	25%	25%			25%	25%			
Section 3c: Use Cases: Use Case Diagram	20%	20%		10%	20%	20%		10%	
Section 3c: Use Cases: Traceability Matrix		100%							
Section 3c: Use Cases: Fully-Dressed Description		100%							
Section 3d: System Sequence Diagrams	100%								
Section 4a: Preliminary Design			50%	50%					
Section 4b: User Effort Estimation							50%		50%
Section 5a: Domain Model			33.33%				33.33%		33.33%
Section 5b: System Operation Contracts	50%				50%				
Section 6: Project Size Estimation		50%				50%			
Section 7: Plan of Work	50%				50%				
Revisions		20%		60%		20%			

Table of Contents

1.	Section 1: Customer Problem Statement	4
1.1.	Problem Statement	4
1.2.	Glossary of Terms	7
2.	Section 2: System Requirements	10
2.1.	Enumerated Functional Requirements	10
2.2.	Enumerated Nonfunctional Requirements	11
2.3.	On-Screen Appearance Requirements	12
3.	Section 3: Functional Requirements Specification	15
3.1.	Stakeholders	15
3.2.	Actors and Goals	15
3.3.	Use Cases	16
3.3.1.	Casual Description.....	16
3.3.2.	Use Case Diagram.....	17
3.3.3.	Traceability Matrix.....	21
3.3.4.	Fully-Dressed Description.....	23
3.4.	System Sequence Diagrams.....	24
4.	Section 4: User Interface Specification	26
4.1.	Preliminary Design	26
4.2.	User Effort Estimation	36
5.	Section 5: Domain Analysis	38
5.1.	Domain Model	38

5.2.	System Operation Contracts	41
6.	Section 6: Project Size Estimation.....	43
7.	Section 7: Plan of Work.....	46
8.	References	48

Section 1: Customer Problem Statement:

Problem Statement:

Residential Family Home:

Home Monitoring System Team,

I currently live in a residential area in the suburbs with my wife, my three children (ages 16, 10, and 5), and my mother in law who is not very well coordinated due to old age. We also have 2 well-behaved dogs. With that being said, you can only imagine how hectic our home becomes on a daily basis trying to run around and get everything done. Between my wife and I going to work, the kids going to school, taking care of my mother in law, as well as a never-ending list that goes on and on, it is extremely difficult to ensure nothing goes wrong in the process. Like most parents, our main concern during all of this pandemonium is the safety and security of our family.

We are burdened with a lot of extra and unnecessary stress that stems from trying to remember if we did something important or not. Many little things can be easily overlooked due to the constant chaos going on. Little things such as remembering if we left the stove on, if all of the doors and windows are closed and locked, or if we shut off the lights become very difficult. Other things such as checking the basement for water leakage or flooding due to rain, or making sure the heat/AC is turned off before we leave seem minimal but are actually major safety concerns. A stove being left on could, in an extreme case, burn our entire house down. Leaving windows and doors unlocked increases the likelihood of a burglar successfully invading our home. On a less serious note, even the money being saved from ensuring the heat/AC is turned off is something we value tremendously. We are looking for a solution that will put our minds at ease and let us continue on with our day without having to worry about the security and safety of our family or home.

Ideally, for us, a system that can be implemented into our mobile devices would be the most beneficial. We would like some sort of infrastructure to be set-up within our home that passively checks for all of the aforementioned problems. My wife and I will then be able to have customized settings on our mobile application that alerts us of potential problems based on our desired preferences. For example, I can turn on “monitoring” when I leave the house (would be even better if the application recognizes I left automatically) and be notified if the stove is still on after 10 minutes. Knowing this, I could go back home (or call somebody who is there) to turn the stove off. This would help us keep our family safe from any possible home issues.

A reliable system of this nature would be invaluable to my family. Knowing I will be alerted anytime there is an issue will allow me to focus more on the things that really matter in life.

Corporate Lab:

I lead a research laboratory where safety is our number one priority. Part of my job is supervising all of the lab technicians to make sure they practice safe work habits and closely follow the rules. Unfortunately, I am only one person and the company employs many lab technicians. It is difficult to keep an eye out on everyone while simultaneously catching every single safety measure they might forget. To remedy this problem, I am in the market for a sort of security system to help me monitor everyone's work and keep our laboratory a safe workspace.

Some experiments our lab technicians carry out require heating up different liquids and chemicals. Believe it or not, I have caught technicians leaving burners on after using them. Besides the obvious fire hazard, this is extremely problematic considering the fact that technicians sometimes work with flammable substances. Another problem we have is with the storage of all these materials used in experiments. We have this huge storage facility where everything the technicians need can be found. On occasion, materials are mishandled by either our lab technicians or our suppliers and we'll have stuff leaking. This is a big issue because if some extremely reactive chemical spills into another reactive substance, then not only are the technicians in danger, but our equipment in storage can also be destroyed. While we're on the topic of our storage facility, another problem we have is unauthorized personnel getting access to storage. We have a few select technicians in charge of the storage facility. They keep it clean, organized, and retrieve materials that any other technicians request. The problem, however, is that the storage technicians often leave the door open or unlocked when they go to retrieve materials for other technicians. In the past, we've had impatient lab technicians go into storage to grab what they needed rather than waiting for a storage technician to help them. This is obviously a big issue as these technicians are breaking the rules and are entering a restricted area. What's even worse is that they will move around chemicals, which messes up the organization system of the storage technicians. We are concerned not only about reinforcing the storage facility, but also our records room. We have someone to oversee the records room, but they're always in and out of there retrieving and storing documents for technicians. The door leading into where our documents are stored ends up open and unlocked most of the time. It is a big concern of ours that someone might try to steal information from our laboratory as we have many competitors in our business.

This only scratches the surface of the problems we have in our laboratory. Ideally we are looking for a monitoring system that: monitors the use of the burners and notifies us if they have been left on for long, monitors our storage facility for any leaks/flooding and notifies us where it is occurring, monitors our doors to the storage and records rooms and notifies us if it is open or unlocked. These specific issues are of highest priority to us and we would invest a great amount into anything that would help us solve them.

School/ College:

I am the team director for undergraduate campus housing. My job is to make sure the living conditions and dorms are capable of handling students living necessities. Students come from different places and have so many unique hobbies and traits so we want them to feel welcomed and at home on campus. In every dorm we have showers, bathrooms, heating, and cooling. We also have safety features such as fire alarms and carbon monoxide alarms. To increase the safety of our dorms we have security locks that only open with certain verifications. As the director for campus housing, my team has been working to make sure every student's life is as smooth and comforting as possible.

As we continue to add more features and safety precautions in every building, we have come up with an adversity. With the added features, our staff has several applications that they have to go back and forth from. One application for the fire alarms, another for the security system, and others for every parts of the student dorms. Our staff wants to monitor all the activities in one screen. Not to mention the students will find it much more comforting if they can have access and knowledge about their own building. On top of this, we would want to know what in the building is malfunctioning, has been broken, or is in use.

There is a certain company that has everything we are looking for to solve all these problems. A company is selling their home automation system product. With this product we want to know all of the safety precautions in each building. If the fire alarm is on, the security system is not locking people out or locking people in. There is no plumbing errors. We want to know everything about our dorms in one screen. Also we would want the company to make a notification if something does go wrong what is it and how to attack the problem. Either calling the fireman, police, or a plumber.

Restaurant:

Hello Home Monitoring System team, I am a small restaurant owner who is interested in your product. As a small business owner I am always making decisions that I believe will benefit my business. I am also, however, the only one who keeps track of all the problems that happen around my restaurant. Some problems go unnoticed until it is too late. And as a small business owner, my main incentive is to increase profit while diminishing loss.

There are many problems I face daily and I believe your system will benefit me. One such problem is checking for water leakage. As an owner of a restaurant it is important to make sure that there is no water leakage near any of my food. Many times when we notice a water leakage in the freezer, it is already too late. This is because we only go into the freezer when the kitchen is out of a certain product, such as lettuce for example. By the time we discover a leakage in the fridge, our food products are already damaged. This leads to a great loss of money for us as well as a safety hazard for our customers. We want to use healthy ingredients in our products and also decrease food waste. We face another common problem during closing time. We always have to make sure that our stove is turned off. On busy nights, however, it is very easy to forget this simple task. I always find myself back at the restaurant after it is closed to

ensure that everything is properly handled. This not only wastes my time, but causes a great deal of worry for me on a nightly basis. If I do not do this, this can easily lead to a fire and can greatly damage my restaurant.

Another great fear of every restaurant owner is the chance of a burglary. Every night there is a chance that someone could easily break into my restaurant and steal valuable items. With a system notifying me of any break ins, I would be able to quickly reach out and call the police department.

Having a system that can detect all of these issues and provide immediate feedback would be very nice to have. Instead of subscribing to multiple systems that only control individual aspects, I would be able to have one central system that monitors everything. Things such as home security and water leakage detection are not currently sold in one package, which means I have to purchase two different detection systems. With a centralized system like yours, I will be able to easily detect all types of problems in my restaurant without spending money or time on multiple products. I will get all my feedback in one device with a system that will notify me when there is a water leak, as well as if there is a break in. I will be able to fix numerous problems at once before significant damage is done. This will help me save money as well as keep my restaurant safe and secure.

Glossary of Terms:

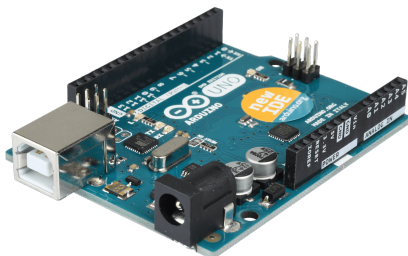
Amazon Web Services (AWS)

A secure cloud services platform, offering compute power, database storage, content delivery and other functionality to help businesses scale and grow

Application Programming Interface (API)

A set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other service

Arduino Uno



Picture of Arduino Uno board

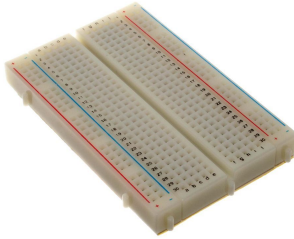
A microcontroller board equipped with sets of digital and analog I/O pins that may be interfaced to various expansion boards and other circuits

AWS Instance

A virtual server for running applications on the AWS infrastructure

Bread Board

A board for making an experimental-model of an electric circuit



Circuit Board

A thin rigid board containing an electric circuit. A printed circuit.

Database

A collection of information that is organized so that it can be easily accessed, managed, and updated

Internet of Things (IOT)

A system of interrelated computing devices that transfer data over a network without requiring any human interaction



Mobile Application

A type of software application designed to run on a mobile device

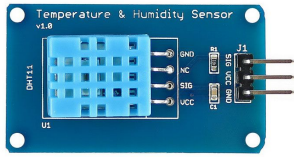
Raspberry Pi

A low cost Linux and ARM-based computer on a small circuit board



Picture of Raspberry Pi Model 3 B+

Sensor



Picture of Arduino compatible temperature and humidity sensor.

A device to detect or measure a physical property and records, indicates, or responds to it

Server

A computer that serves information to other computers, called clients. These computers can connect to each other through a local area network (LAN) or a wide area network (WAN).

User Interface (UI)

The means by which the user and the computer system interact, in particular the use of input devices and software

WiFi Module

An independent self-contained unit capable of accessing WiFi network

Section 2: System Requirements:

We have extracted and analyzed the functional and nonfunctional system requirements from the Product Statements that our customers have given us. Using User Stories, we are able to make a clear and concise table to show exactly what our system aims to accomplish.

The most important task for any software to accomplish is to satisfy the requirements of the customer. Using what the customers have given us in the product statements, we were able to extract the major requirements for our specified system. These requirements however, are not in any order of importance to the customer. For us developers though, priorities in requirements matter. Hence, we need to prioritize all the requirements that were extracted and order them from highest to lowest priority in development. This will moreover help in the development of the software.

With prioritization in mind, we have created the table below. In the table there is a column for priority, which ranges from 1 - 5, with 1 being optional and 5 being absolutely mandatory.

Enumerated Functional Requirements:

ID	Priority	User Story
ST-1	5	As a user, I can check if the stove is on/off.
ST-2	5	As a user, I can check if my doors and windows are properly locked.
ST-3	2	As a user, I can check if my lights are on/off.
ST-4	5	As a user, I can check whether or not my basement has flooded.
ST-5	1	As a user, I can look at reliable businesses near me to solve my problem.
ST-6	5	As a user, I will be alerted if anything dangerous or damaging happens when I am not at home.
ST-7	1	As a user, I can turn off the monitoring system.
ST-8	5	As a user, I can check to see if anyone has broken into my house.
ST-9	3	As a user, I can use this system on a cellular device
ST-10	5	As a user, I can view a live camera feed of my home when anything is detected.

The priority of the functional system requirements has considerations of practicality and safety in them. Keeping these two principles in mind, we chose checking if the stove is on/off (ST-1), locked/unlocked doors and windows (ST-2), checking if the basement is flooded (ST-4), checking whenever anyone has entered the house (ST-9), and viewing a camera feed of dangerous/damaging occurrences (ST-6, ST-11) as the top priorities. Checking if the the doors are locked and if anyone has entered the user's home or business could also pose an important safety factor regarding potential thieves. We also included mobile integration for the system (ST-10) and being able to control the system remotely from any wifi destination (ST-8) as priorities three and four respectively.

The rest of the functional system requirements are identified as a lower priority, but are in no means unnecessary for the function of the system. Making sure the lights in the home or business are off (ST-3) may also be an important safety concern. Lastly, we marked off being able to turn off the monitoring system (ST-7) and looking at reliable businesses to solve the problem (ST-5). Being able to turn off the monitoring system allows for privacy and safety for users in case of any system failures that may result due to safety hazards. Additionally, the practicality of being able to identify reliable businesses should be a functional concern for our product.

Enumerated Nonfunctional Requirements:

ID	Priority	User Story
ST-11	2	As a user, I should understand how to use the system easily. The user interface should be easy to read and be usable for anyone.
ST-12	5	As a user, I should be able to check the status of my home at all times, so there cannot be any downtime for the application where my house is left vulnerable.
ST-13	4	As a user, I should be able to access the application from anywhere.
ST-14	4	As a user, I should be able to use this system on any Android device.
ST-15	3	As a user, I should be given quick and informative directions in case of an emergency.
ST-16	5	As a user, as soon as a problem is detected in my house, I should be alerted of it and be able to view a camera feed so I can address it as soon as possible.
ST-17	1	As a user, I should be able to contact the support team to troubleshoot any possible hardware/software errors.

ST-18	1	As a user, I should be able to easily install the sensors in my home.
-------	---	---

As for the Non-Functional Requirements we broke them down to 4 categories: Usability, Reliability, Performance, and Supportability.

Usability: Usability is the ease of use and learnability of a device and object. A user should be able to access this application through (ST-14). Not only that but the application itself should be easy to learn and utilize.

Reliability: Reliability, or dependability, describes the ability of a system or component to function under stated conditions for a specified period of time. A user should be able to check the status of their home, restaurant, etc. whenever they want (ST-12). As such the user should also be able to access the application from any location as long as they have an Internet connection (ST-13).

Performance: Performance of a computer is essentially estimated in terms of efficiency, effectiveness and speed. If an emergency at a user's home occurs, the said user should be alerted immediately (ST-16). Furthermore, alongside the alert, the user should also be given quick and informative instructions (ST-15).

Supportability: Supportability refers to the ability of technical support personnel to install, configure, and monitor computer products. Technical support personnel should also be able to identify exceptions or faults, debug or isolate faults to root cause analysis, and provide hardware or software maintenance to solve a problem and restore the product into service. The user should also be able to install the sensors easily (ST-18). In case of any technical problems with the application or sensors, there should be a support team that they can contact (ST-17).

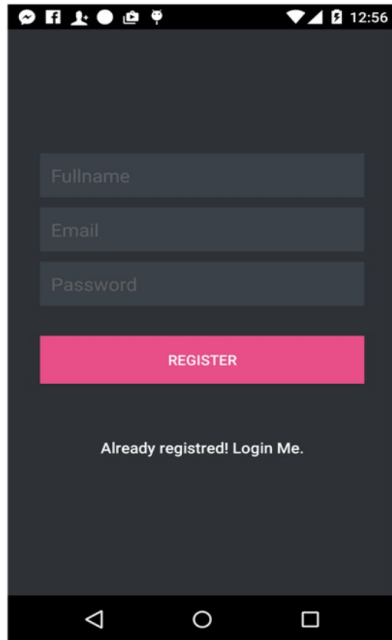
On-Screen Appearance Requirements:

ID	PW	Description
ST-19	5	The interface shall allow the user to input a username and password to securely log into their home server.
ST-20	5	The interface shall show users the status of different parts of the home by clicking different icons/buttons.
ST-21	5	Users should be able to click a button to turn on notifications that send alerts to their phones when something goes wrong.

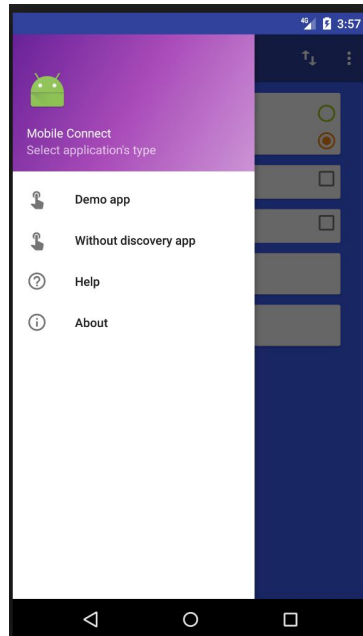
ST-22	3	The interface shall allow a user to switch to an "Account" screen to view database information and the other users logged in.
ST-23	4	The interface shall allow a user to “Favorite” specific appliances to view when the application starts up.
ST-24	3	The interface shall allow user to switch to a "Support" screen to view local businesses using Google Maps.
ST-25	4	The user will be able to click a "Log Out" button to log out of their server.
ST-26	1	The interface shall allow a user to directly call a business from the support section when clicking a "Call" button.
ST-27	3	The interface shall allow a user to call emergency phone numbers.
ST-28	5	The application shall allow a user to view a live camera feed when motion is detected in the home.

The priority of the On-Screen Appearance Requirements is to provide the best possible user experience. Because dependability and safety are crucial aspects of our system, it is important to provide an easy and quick user interface to navigate through the application.

To ensure that this application is safe and secure, a login screen (ST-19) will be the first graphic a user will see on startup, similar to the one shown below. After this, the rest of the application will be easy to navigate through the application using different tabs for each section, each showing important features ranging from ST-20 to ST-27, shown below. Each tab will take the user to the appropriate page in the application.



(ST-19)



(Navigating from ST-20 to ST-28)

Section 3: Functional Requirements Specification

Stakeholders:

1. Internal Stakeholders

- a. Developers : The people who have designed the system and decide how and what requirements to prioritize and fulfill.

2. External Stakeholders

- a. Home Owners: The customers who own the homes that have installed our product for security and protection. They value the reliability, function, cost performance, and quality of our system.
- b. Business Owners: The customers who own businesses like office workplaces that have our product installed for security and hazard prevention. They value the reliability, function, cost performance, and quality of our system.
- c. Restaurants: The customers who are the restaurant owners that installed our product into their workplace for everyday security protection. They value the reliability, function, cost performance, and quality of our system.
- d. Institutions: The customers who own an institution such as a school or college that have our system installed for security protection. They value the reliability, function, cost performance, and quality of our system.

Actors and Goals:

Actors	Type	Goal
Users	Initiating/Participating	To turn on/off the system, download, and use the application to check the status of all the appliances
Sensors	Initiating	To collect various signals and send it to the AWS instance
AWS Server	Initiating	To store data from sensors
Application	Initiating/Participating	To receive data from the AWS instance and alert users about the status of their homes. To allow users to check the status of their homes.

Use Cases:

Casual Description:

Actor	Actor's Goal (What the actor intends to accomplish)	Use Case Name
User	To download the application on any Android Device	Download(UC-1)
User	To use the application with an Internet connection	Connect (UC-2)
User	To disarm the system	Disarm (UC-3)
Sensor	Send a signal when safety or security hazards are detected	Signal (UC-4)
App	To inform the user which support team to contact in case of an emergency	Contacts (UC -5)
App	To notify the user when there is a leak, stove is left on, and/or an intruder enters the home, company, etc.	Notify (UC-6)
User	To create new accounts to access the system	New Accounts (UC-7)
User	To add new sensors to the system	Add (UC-8)
User	To check the status of all the sensors of the system	Check (UC-9)
User	To control how often the notifications are sent	Control Alert (UC-10)
App	To notify the user if there is a network error or the signal of the network is weak	Network Error (UC-11)
User	To unlink user account from the phone application	Log Out (UC-12)

Use Case Diagram:

User Case 1: Download

Related Requirements: ST-9, ST-14

Initiating Actor:User

Actors Goal: To download the application from the Android store and be able to use it on any Android device

Participating Actor: Application

Precondition: User has an updated Android device with an internet connection

Postcondition: The application successfully downloads and connects to a server via Internet

Flow of Events for Main Success Scenario:

- 1. User downloads the application from Android Store
- 2. User makes an account on the application
- ← 3. User successfully connects

User Case 2: Connect

Related Requirements:ST-6, ST-12, ST-16

Initiating Actor: User

Actors Goal: To be able to access the application and receive notifications while away from home via cellular data or any WiFi.

Participating Actor: Server

Precondition: The user has cellular data enabled or is connected to some WiFi and has an account already made

Postcondition: The user will be able to check the status of the home and receive alerts

Flow of Events for Main Success Scenario:

- ← 1. The user connects to the AWS Server
- ← 2. The AWS instance sends data to the application via the Internet
- 3. The application sends the data received by the AWS server to the user as a notification

User Case 3: Disarm

Related Requirements: ST-7

Initiating Actor: User

Actors Goal: The ability to view various appliances in your home and safely shut down the notification of the appliance through user's cellular device

Participating Actor: App

Precondition: The user, as well as the sensors, are connected to the application.

Postcondition: The user has deactivated certain sensors

Flow of Events for Main Success Scenario:

- 1. The user opens the application and checks specific sensors

- 2. User clicks on the sensor he/she wants to disable
- ← 3. The interface has a “disarm” option that the user clicks
- 4. The user clicks “disarm” and receives a notification stating that the sensor is disabled

User Case 4: Signal

Related Requirements: ST-6, ST-12, ST-16

Initiating Actor: Sensor

Actors Goal: To send a signal to the user that informs them when there is a leak, when the stove is left on, and/or an intruder enters the home, company, etc.

Participating Actor: App, User, Server

Precondition: The sensors are continuously checking the status of the home

Postcondition: A signal is sent to the server, and the user receives a notification

Flow of Events for Main Success Scenario:

- ← 1. The sensors picks up a signal that there is a change in the system.
- ← 2. The signal is sent to the AWS cloud server
- ← 3. The application receives the signal from the AWS server
- ← 4. The user is notified through the application

User Case 5: Contact

Related Requirements: ST-5, ST-15, ST-16, ST-17

Initiating Actor: App

Actors Goal: Once the app alerts the user about any possible issues, the app will provide the user with detailed steps on how to solve the issue.

Participating Actor: User

Precondition: A problem occurs and is detected by the sensor and the user is notified

Postcondition: The user is informed and will be able to act accordingly to solve the issue

Flow of Events for Main Success Scenario:

- ← 1. App notifies user on any possible issues(fire, water leak, gas leak, etc.)
- 2. The notification is clicked and user is taken to the application
- ← 3. Depending on the problem, the application will display suggestions on steps to resolve the issue.

User Case 6: Notifications

Related Requirements: ST-5, ST-6, ST-8, ST-9, ST-10, ST-15, ST-16

Initiating Actor: Application

Actors Goal: Notify the users about any signals that have been picked up by the sensors

Participating Actor: User, Server, Sensor

Precondition: The User is logged into the App on the Android device, Sensors are properly connected to the App, User is connected to any WiFi or data network and has notifications on

Postcondition: Notification will have enough info for the User to be able to take appropriate action

Flow of Events for Main Success Scenario:

- ← 1. Sensor picks up a signal
- ← 2. Signal is sent to the AWS Server
- ← 3. AWS Server sends this data to the App
- ← 4. App will send a push notification to the User

User Case 7: New Accounts

Related Requirements: ST-11, ST-13, ST-14

Initiating Actor: User

Actors Goal: To create new accounts to access the system

Participating Actor: Application

Precondition: Having the app on an android phone

Postcondition: Multiple accounts can log in to the application.

Flow of Events for Main Success Scenario:

- ← 1. App displays the login interface.
- 2. User selects 'add a new account' button.
- ← 3. User fills out information.
- ← 4. App sends a confirmation email.
- 5. User confirms account and logs in to the app with the new account.

User Case 8: Add

Related Requirements: ST-18

Initiating Actor: User

Actors Goal: To add sensors to the system.

Participating Actor: Application, Sensors, System

Precondition: Need an account and require the sensors to be properly installed physically beforehand.

Postcondition: The sensor will be able to send alerts to the app.

Flow of Events for Main Success Scenario:

- 1. User logs in to the app.
- 2. User selects to add a new sensor.
- ← 3. System looks for a new sensor signal.
- ← 4. Sensor connects to the system.

User Case 9: Check

Related Requirements: ST-1, ST-2, ST-3, ST-4, ST-8, ST-9, ST-10, ST-12, ST-13, ST-14

Initiating Actor: User

Actors Goal: To check the status of all the sensors in the system.

Participating Actor: App, Sensors, System

Precondition: User must be logged in the app and the sensors must be properly connected.

Postcondition: User will be shown the status of all the sensors through the app interface.

Flow of Events for Main Success Scenario:

- 1. User logs into the app.
- 2. User goes into the dashboard
- ← 3. The status of all appliances at home are displayed

User Case 10: Control Alert

Related Requirements: ST-6, ST-11, ST-13, ST-14

Initiating Actor: User

Actors Goal: To have control over the notifications that are sent and to control their regularity.

Participating Actor: Application

Precondition: App on Android phone, Connected system with other sensors in place

Postcondition: Full adaptive control of the notifications sent to the User

Flow of Events for Main Success Scenario:

- ← 1. User goes to Settings and clicks on the notifications tab
- 2. User can view each sensor and their notification settings
- 3. User can toggle on/off notifications for specific sensors that they do not want to view

User Case 11: Network Error

Related Requirements: ST-6, ST-12, ST-16, ST-17

Initiating Actor: App

Actors Goal: To be notified in case a network error or weak signal may occur.

Participating Actor: User

Precondition: App is downloaded and set up on Android phone, Notification system is properly set up

Postcondition: User is notified of a network error, and can take the appropriate steps to resolve the network connection

Flow of Events for Main Success Scenario:

- 1. The signal weakens or disconnects
- 2. The application detects a network error
- 3. App sends notification to the User

User Case 12: Log Out

Related Requirements: ST-11

Initiating Actor: User

Actors Goal: To remove the user from the application

Participating Actor: Application

Precondition: The user downloads the software and is able to successfully log into the system

Postcondition: The user is now logged out of the system and is unable to see his/her account details

Flow of Events for Main Success Scenario:

→ 1. User goes into the application settings

→ 2. User presses the “Logout” button

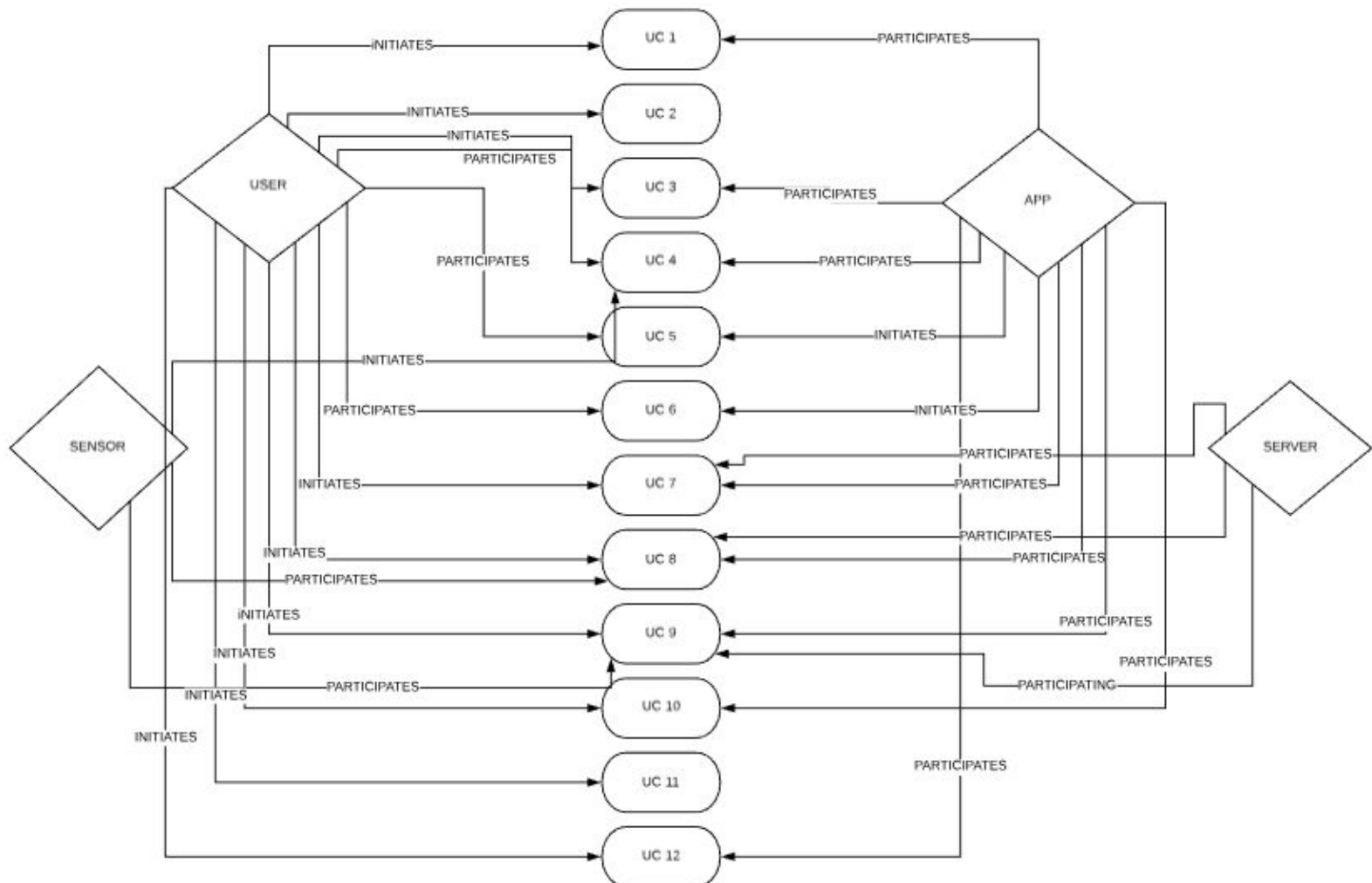
← 3. App unlinks the user account from itself and returns to the “User Registration” screen

Traceability Matrix:

Req	PW	UC 1	UC 2	UC 3	UC 4	UC 5	UC 6	UC 7	UC 8	UC 9	UC 10	UC 11	UC 12
ST1	5									X			
ST2	5									X			
ST3	2									X			
ST4	5									X			
ST5	1					X	X						
ST6	5		X		X		X				X	X	
ST7	1			X									
ST8	4						X			X			
ST9	5	X					X			X			
ST 10	3						X			X			
ST 11	2							X			X		X
ST 12	5		X		X					X		X	
ST 13	4							X		X	X		

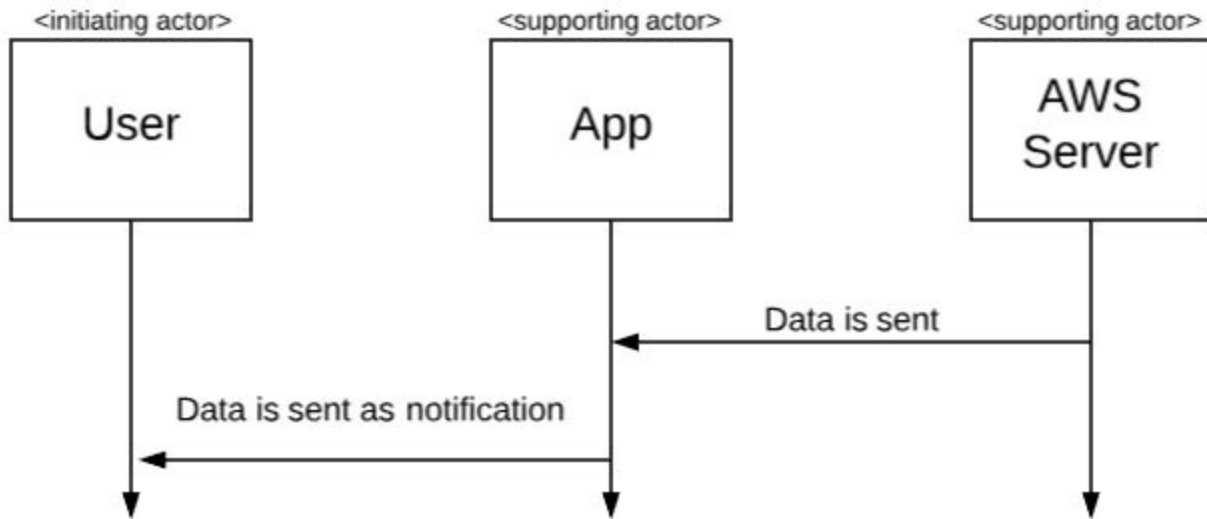
ST14	4	X						X		X	X		
ST 15	5					X	X						
ST 16	5		X		X	X	X					X	
ST 17	1					X						X	
ST 18	1								X				
ST 19													
ST 20	1												
ST 21	1												
ST 24	3												
ST 25	1												X
ST 26	5												
ST 27	2												
Max PW		5	5	1	5	5	5	4	1	5	5	5	1
Total PW		9	15	1	15	12	27	10	1	42	15	16	3

Fully-Dressed Description:

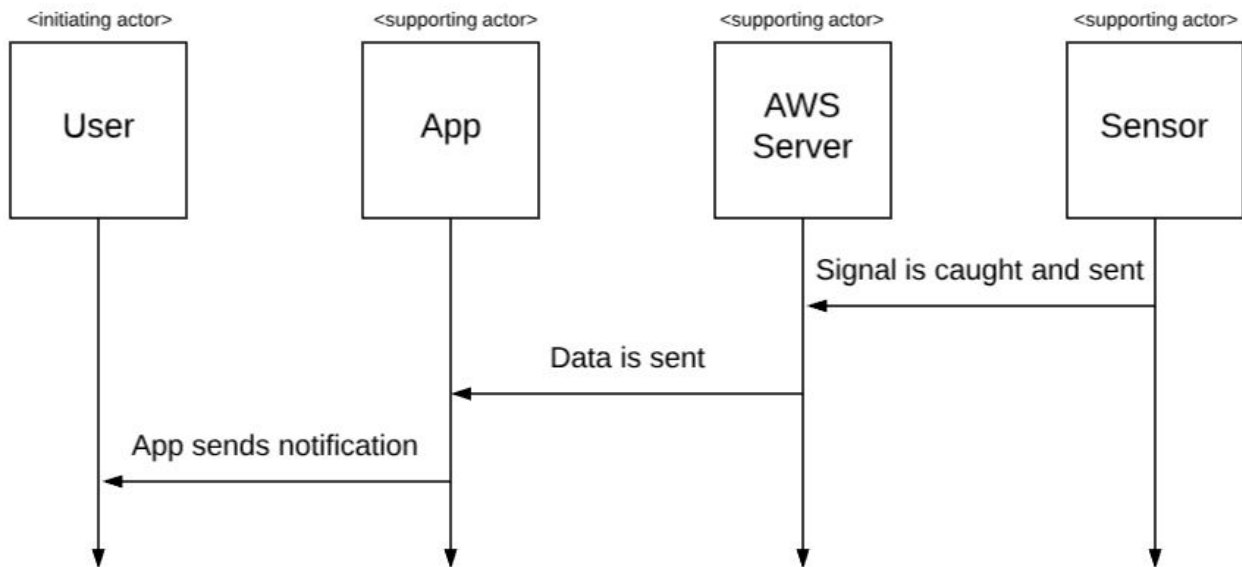


System Sequence Diagrams:

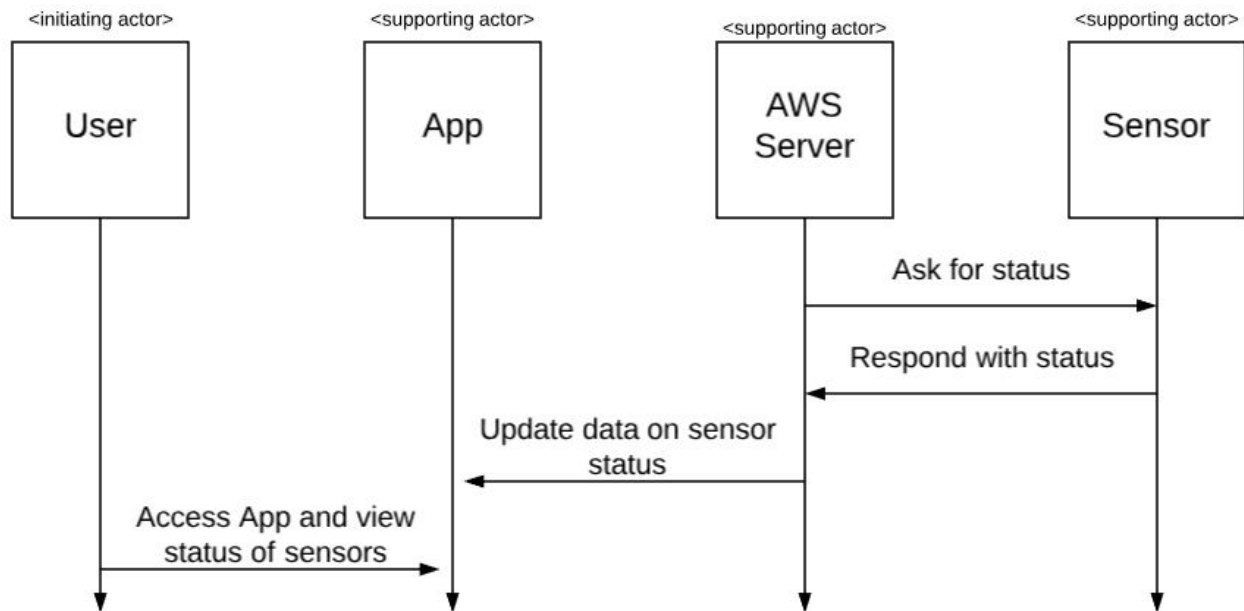
Connect sequence diagram UC-2:



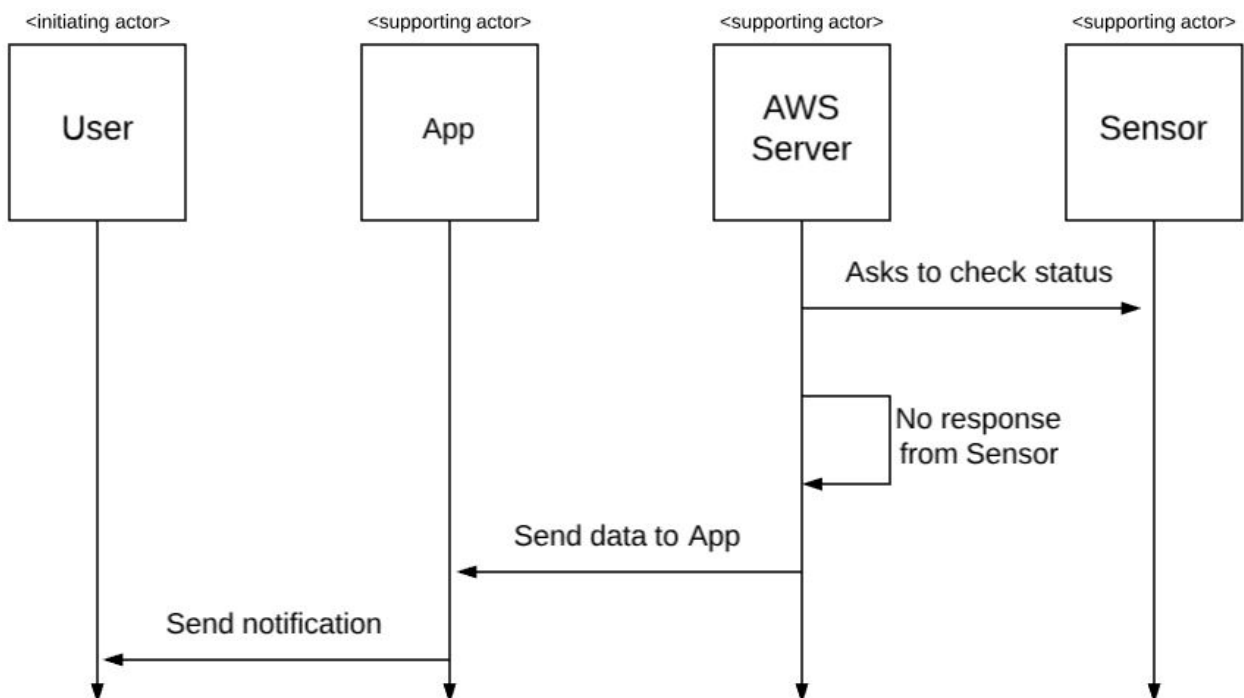
Signal sequence diagram UC-4:



Check sequence diagram UC-9:



Network Error sequence diagram UC-11:



Section 4: User Interface Specification:

Preliminary Design:

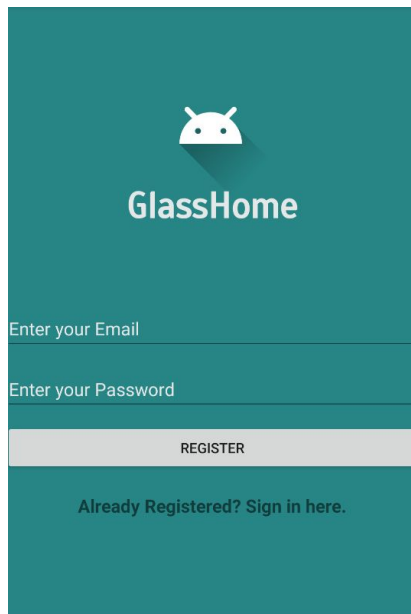
In this section we will give you a detailed explanation of our design, with a step by step guide to run our system. We aim to make it simple and easy to understand, but also complex in the features and information provided to a user.

Access

To access our application, a user will need to download the application from the Android Store. Once the user has downloaded the application and registered an account, he/she will be able to log in by entering a username and password.

New Accounts

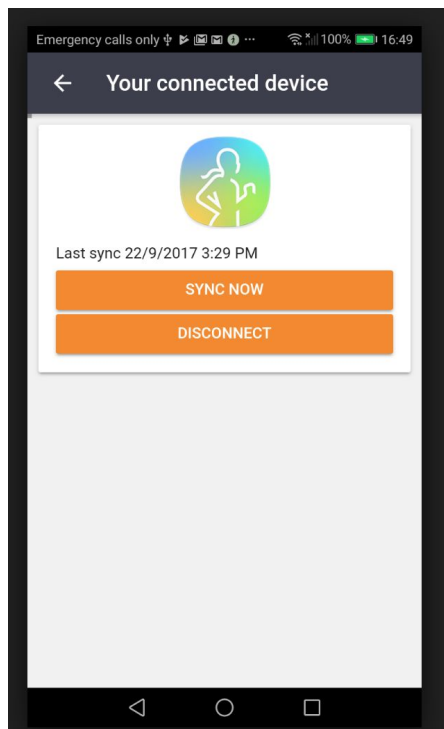
To create a new account, a user can open the application once it is downloaded and fill out a username and password. The username is the email of the user, and once the user has done this, he/she can click “Register User” as shown below. This ensures the user’s security because he/she will have to log in with a private password to access his/her personal system. The application will then send a confirmation email once registered, where he/she can confirm their account. Once confirmed, the user can log in any time.

The image shows a user registration screen for an application named "GlassHome". The screen has a teal background. At the top, there is a white Android robot icon and the text "GlassHome" in white. Below this, there are two input fields: "Enter your Email" and "Enter your Password". A light gray button labeled "REGISTER" is positioned below the password field. At the bottom, there is a link that says "Already Registered? Sign in here." in white text.

User registration

Connect

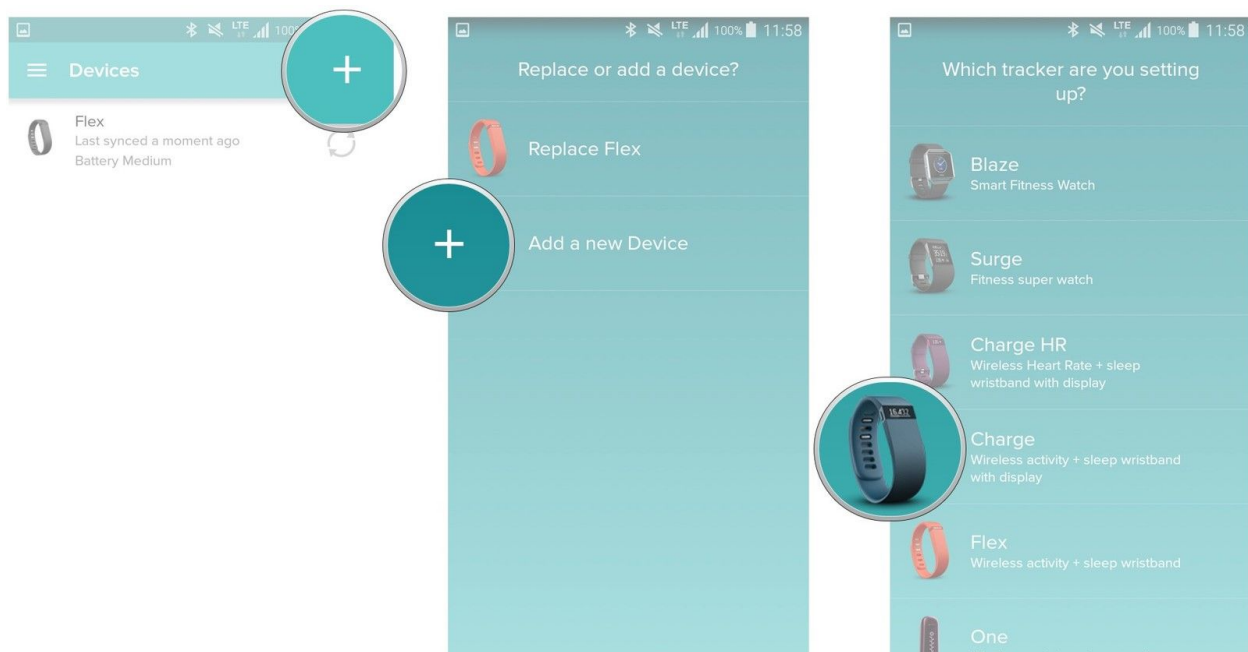
To receive notifications and view the status of their appliances, a user will need to be connected to either a WIFI connection or be connected to Cellular Data. The internet connection will be necessary to access the AWS Server where the sensor data is collected, and then sent to the application. The sensors placed at home will also need to be properly connected to a WIFI connection to send data. When connected, the user will receive notifications on his/her phone that display specific alerts based on the appliance that needs to be checked or monitored. A notification when properly connected to the internet will show up on the user's phone as shown in the pictures below.



Successfully Connected Device

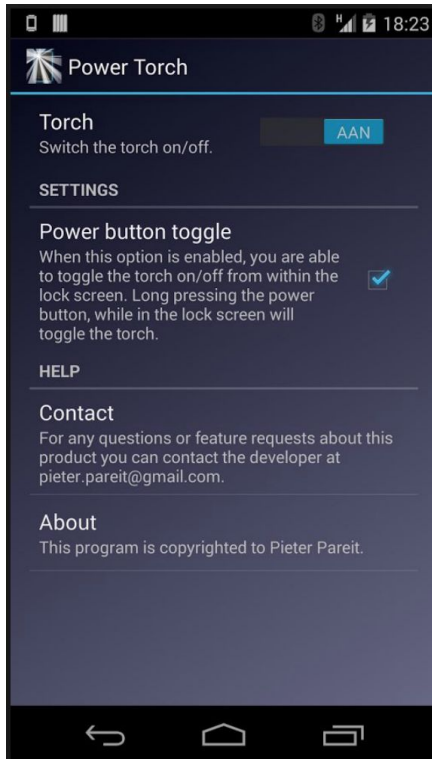
Add

This use case allows the user to quickly add a new sensor to their set of existing sensors, or to add their first sensor to the list. Once logged into the application, a list (empty at first) will be presented to the user, and the user can click on the top right-hand corner at the “+” symbol in order to add a sensor. The user will then be presented with a new screen that scans the existing server and lists all possible sensors that the application can connect to. Once the user clicks the appropriate sensor they want connected, the screen will go back to the original interface with a new sensor added to it. Users can then repeat this process for multiple sensors or keep any amount of sensors they already registered.



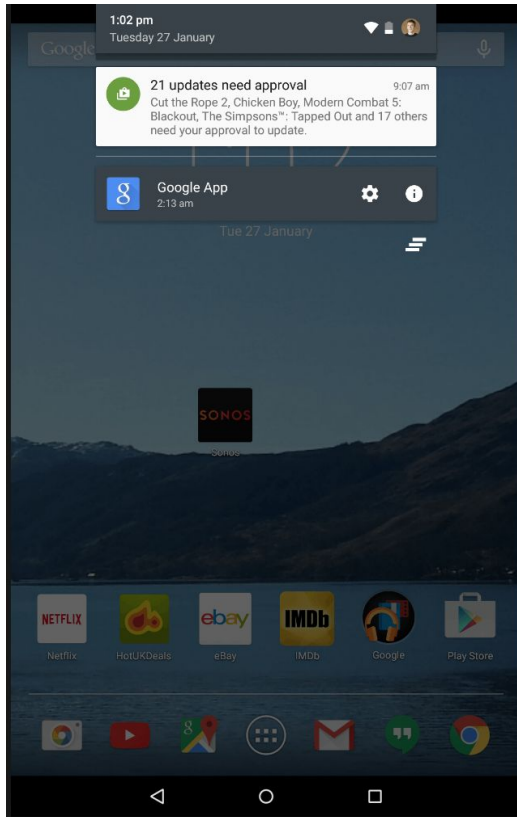
Disarm

Once the application is successfully connected to the sensors, the user can click the “Sensor” section of the application and click on specific sensors that he/she wants to view. Once clicking the sensor, the user can click the button that says “Disarm”. This will disable the specific sensor that the user selects. The disarm option will look similar to the picture shown below, where the user can toggle the top button to disable a specific sensor.



Signal

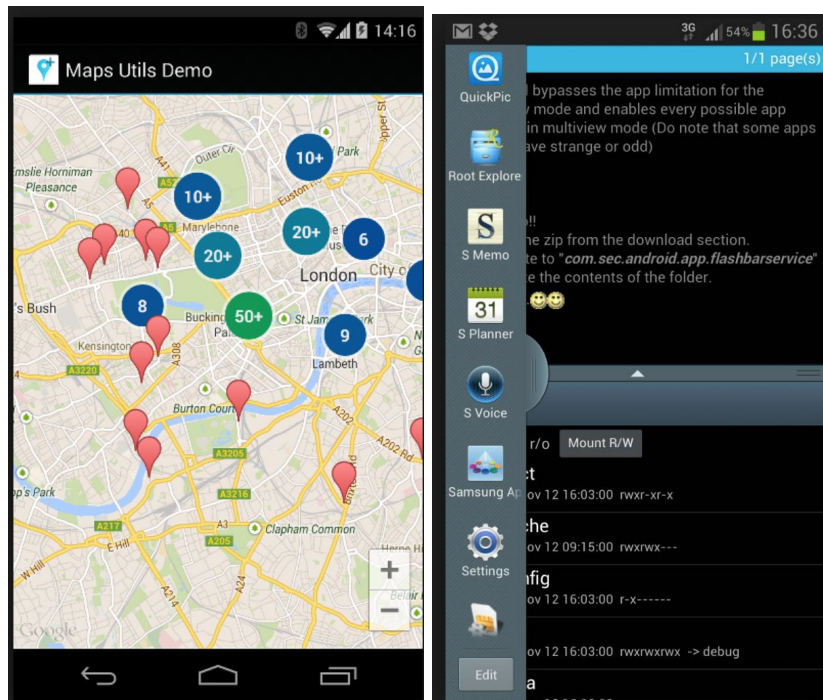
When the sensors in a home detect changes such as high humidity or a carbon monoxide presence, they will be sent to the AWS Cloud Server. The user should have push notifications enabled on his/her phone. When the AWS Server picks up a presence of any hazardous event, it will send the signal to the Android application, which will then send a push notification to the user's phone. The user will be able to click the notification and take the appropriate actions required. This should look like the image shown below.



Contact

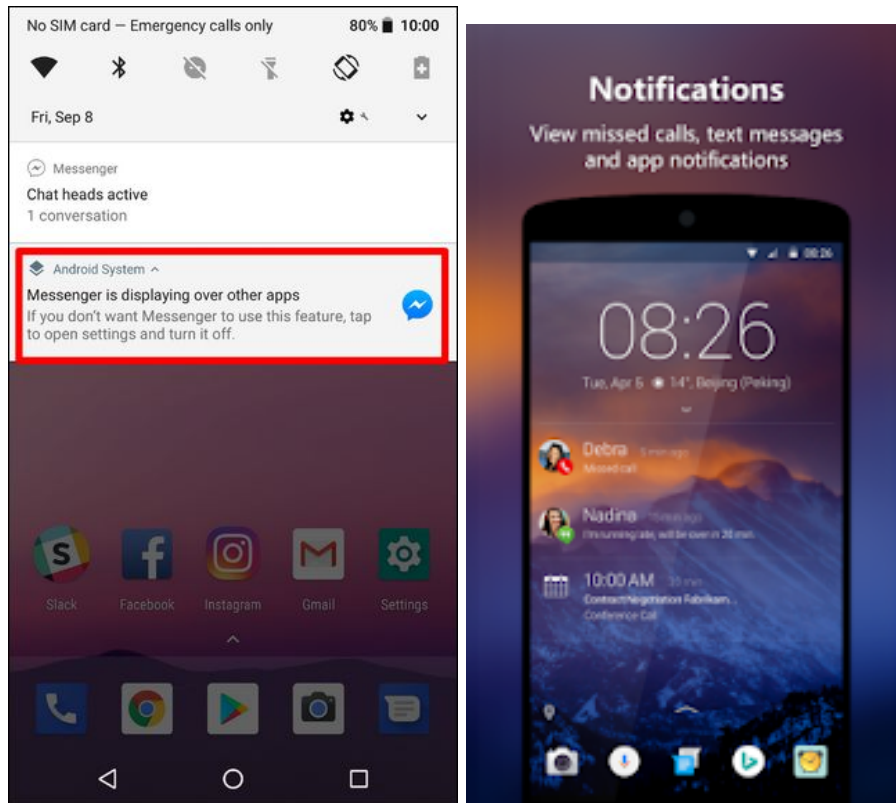
When the user is notified about a problem, he/she can click the notification and will be directed to the application. Once the application is open, the user can click the “Support” tab in the application and it will take the user to a page that shows suggestions on how to resolve different types of issues. The user can click on the button under the specific sensor he/she wants to view, and a page will open up showing the contact information of various services and also instructions on solving frequent issues. A Google Maps API will display the location of these services, and the user can tap on an icon to click a specific service.

The page will look similar to the images on the next page:



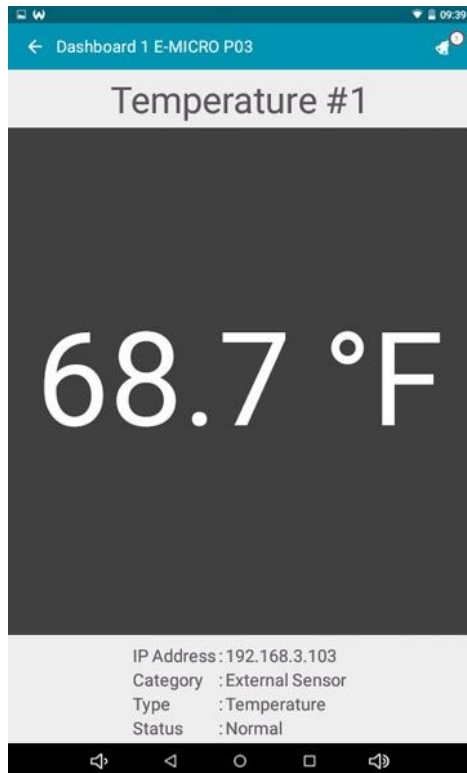
Notify

This user case is initiated by the application where the sensors are connected to. The user should receive notifications if any of the sensors have an alert that they are programmed to announce. The user must enable notifications from the application by toggling it on or off through Settings. Once selected as On, the user's device should receive a push notification reading "Home Security Alert" followed by the specified sensor-specified alert to the lock or home screen. Clicking on the notification should allow the application to open up and present the user with a summary of the alert and a 'help' indicator in order to call/schedule the necessary assistance needed.



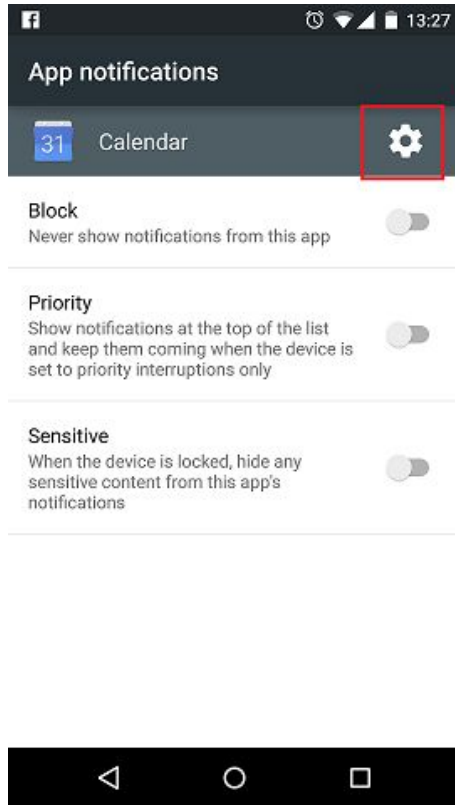
Check

This user case allows the users to check on the status of all the sensors registered to the application. Once a user opens the application and is presented with a list of sensors, they can then click on a given sensor and have a new page opened. On this page, the status of the sensor should be displayed as specified by the sensor. The status depends on the sensor's purpose, as the door sensor would display if the door were open or not and the water leak sensor would display if there was a water breach or not. The 'check' user case allows the sensor information to be presented to the user on top of any and all notifications that may be displayed.



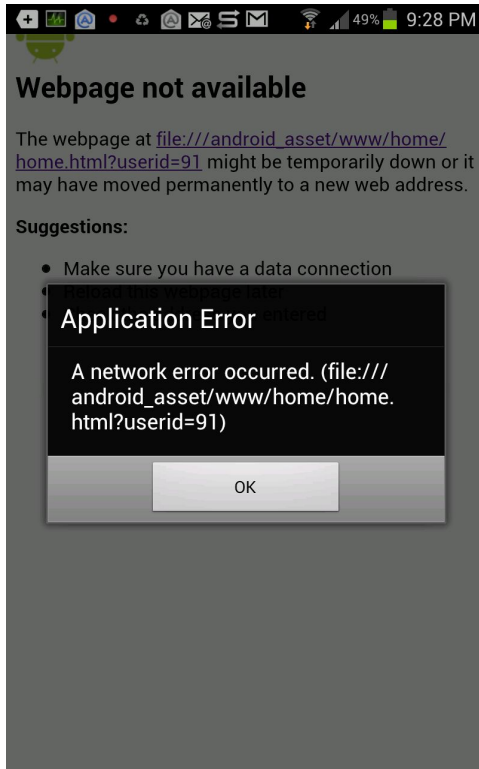
Control Alert

This user case allows the user to specify notifications in the settings section of the application. The user can head over to the settings button located on the top left of the app and then be presented with a notification preferences section. This allows the user to control what specific sensors they want to receive notifications for, essentially allowing more customizability to the user.



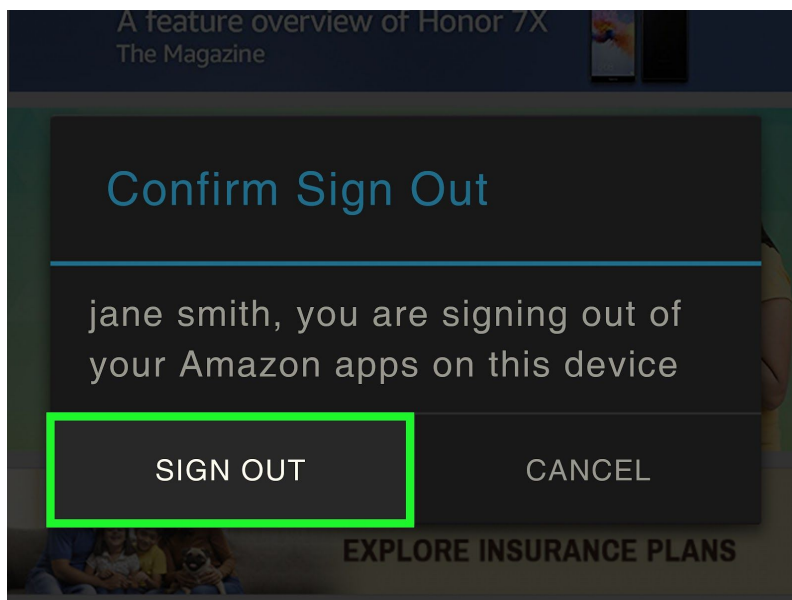
Network Error

This user case presents the user with a notification if the wifi signal in the host location goes out. If the sensors lose connection to the application due to its wifi connection being disabled, the application will present the user with a notification telling the user that the connection has been lost. The user, in this case, would have to go back to the host location to fix the issue.



Log Out

This user case is located in the settings page of the application. The user can head over to this by clicking the settings icon and then accessing the 'logout' button. This will prompt the user to then be presented to the original login screen when they downloaded the app so that they could login using a different username or stay logged out.



User Effort Estimation:

- Logging into account
 1. Navigation: 1 button tap, assuming application is already open, the “User Registration” screen will be the first thing the user sees.
 - a. Tap “Already Registered? Sign in here.” button
 2. Data Entry: 3 button taps in addition to the button taps corresponding to the length of the email and password.
 - a. Tap the “Enter your Email” text field
 - b. Enter your email address
 - c. Tap “Enter your Password” text field
 - d. Enter your password
 - e. Tap the “Login” button
- Creating a new account
 1. Navigation: 0 button taps, assuming application is already open, the “User Registration” screen will be the first thing the user sees.
 2. Data Entry: 3 button taps in addition to the button taps corresponding to the length of the email and password.
 - a. Tap the “Enter your Email” text field
 - b. Enter your email address
 - c. Tap “Enter your Password” text field
 - d. Enter your password
 - e. Tap the “Register User” button
- Connecting/Syncing to the network
 1. Navigation: 1 button tap, assuming the “Your connected device” screen appears directly after logging in or registering.
 - a. Tap “Network Settings”
 2. Action: 1 button tap, assuming the network being connected to is the same one the sensors are connected to.
 - a. Tap “SYNC NOW”
- Disconnecting from the network
 1. Navigation: 1 button tap, assuming “Your connected device screen appears directly after logging in or registering.
 - a. Tap “Network Settings”
 2. Action: 1 button tap
 - a. Tap “DISCONNECT”
- Adding a new sensor
 1. Navigation: 0 button taps, assuming application is already open and the user is already logged in, the “Sensor List” screen will be the first thing the user sees.

2. Action: 2 button taps to finish registering a new sensor.
 - a. Tap the “+” button
 - b. Tap desired sensor from list
- Disarming Appliance
 1. Navigation(1): 2 button taps, assuming application is already on “Sensor List” screen when the application opened and user logged in or registered
 - a. Tap sensor that is associated with a specific appliance
 2. Action(1): 1 button tap, assuming the appliance is currently on
 - a. Tap the “Disarm” button (shuts down appliance)
 3. Navigation(2): 0 button taps, assuming a notification appears letting the user know an appliance is still on
 4. Action (2): 1 button tap
 - a. Tap the “Disarm” button (shuts down appliance)
- Checking status of a registered sensor
 1. Navigation: 0 button taps, assuming application is already open and the user is already logged in, the “Sensor List” screen will be the first thing the user sees.
 2. Action: 1 button tap to open the “Details” screen of the desired sensor.
 - a. Tap desired sensor from list
- Modifying the notification settings
 1. Navigation: 2 button taps, assuming application is already open and the user is already logged in, the “Sensor List” screen will be the first thing the user sees.
 - a. Tap the “Settings” button
 - b. Tap the “Notifications” button
 2. Action: 1 button tap to modify notification settings of a specific sensor.
 - a. Tap to toggle notifications for any given sensor
- Logging out of account
 1. Navigation: 1 button tap, assuming application is already open and the user is already logged in, the “Sensor List” screen will be the first thing the user sees.
 - a. Tap the “Settings” button
 2. Action: 1 button tap to disconnect user account from application.
 - a. Tap the “Logout” button

Section 5: Domain Analysis:

Domain Model:

Concept Definitions

Responsibility Description	Type	Concept Name
RS1: Interacts with all of the subsystems of the home security system	D	Controller
RS2: Verifies the credentials of the User in order to get the User successfully logged into the app	D	IDChecker
RS3: User interacts with this and enters a UserID and Password in order to be logged in	K	IDEntry
RS4: User opens the main menu of the Status app to check each sensor status	K	StatusDisplay
RS5: Responsible for acquiring the status of the sensors by speaking to the sensors directly	D	SensorOperator
RS6: Obtains the status of each of the sensors in order to efficiently display the status to the User	D	StatusDisplayList
RS7: Alerts the user of a potential problem or security issue received from the sensor	K	Alert
RS8: Notifies the user of a home hazard such as a gas leak, a water leak, etc.	K	Notification
RS9: Connects the sensor controller to the application controller	D	AWS

Association Definitions

Concept Pair	Association Name	Association Description
Controller ↔ IDChecker	Verifies identity	Verifies User credentials when logging into the app
IDChecker ↔ Controller	Passes check	Passes the success or failure back to the controller of identity verification

Controller ↔ StatusDisplayList	Obtain/Refresh from query	Obtain the status display list in order to efficiently relay the information to the user
StatusDisplayList ↔ SensorOperator	List query	Lists all the sensor status types
AWS ↔ Controller	Connects	AWS server connects the sensor controller to the application controller
SensorOperator	Request from sensor	Requests and obtains the status of the sensors from the sensors themselves

Attribute Definitions

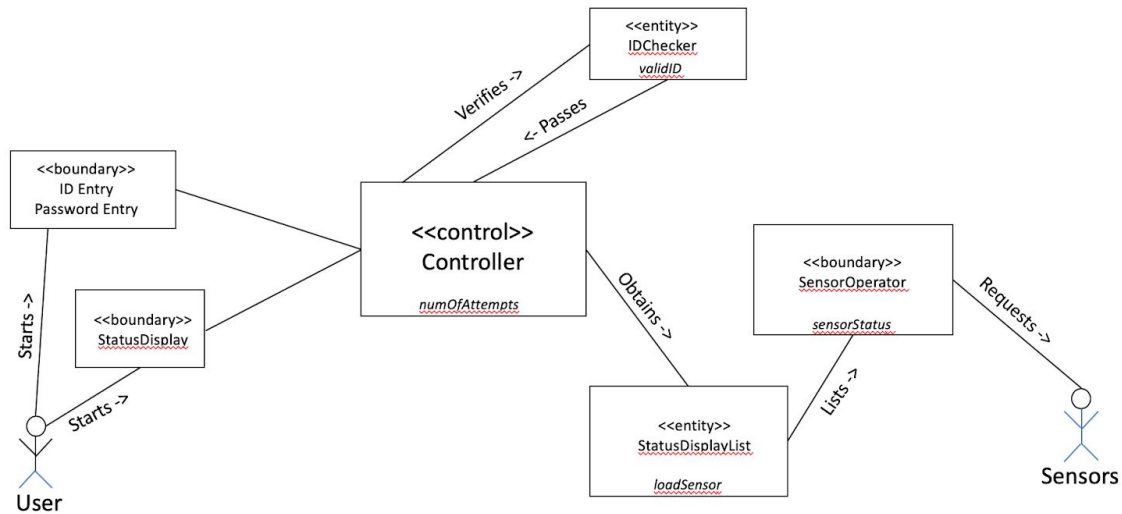
Concept Name	Attribute Name	Attribute Description
Controller	numOfAttempts	Used to determine the number of attempts the user inputted as identity verification
Controller	listUpdate	Updates the lists on the sensor statuses
Controller	listRefresh	Periodically refreshes the sensor list with new statuses
Controller	receiveStatus	Used to receive an incoming status notification
Controller	sendStatus	Used to send the notification to the User
Controller	isAlert	Determines if the notification is an emergency alert
Controller	sensorCheck	Check the sensors for alers or notifications
IDChecker	validID	Matches the ID and password with an ID - password combination

		already present in the database
IDChecker	sensorRequest	User request to check status of sensors
Notification	sendNotify	Pushes notification to the User
App	displayStatus	Checks to see if the display is accurately showing sensor statuses
App	isNotify	Determines if the application needs to send out a notification or alert
App	alertStatus	Further determines the type of alert to best help the User
StatusDisplayList	loadSensor	Responsible for putting the sensor statuses acquired into list form
SensorOperator	sensorStatus/sensorConnection	Distinguishing sensor statuses as acquired by the sensor

Login and Check:

The login and check domain model of the Home Security and Safety System is shown as follows. The user is responsible for initiating one of two possible interactions: logging into the Status app, or opening the main menu of the Status app to check the sensor status. This then goes through a common controller to check the number of attempts that the user tried to login to the app with. It then verifies the credentials using an IDChecker entity and if it passes, it obtains the Status Display List on the dashboard of the Home Security application.

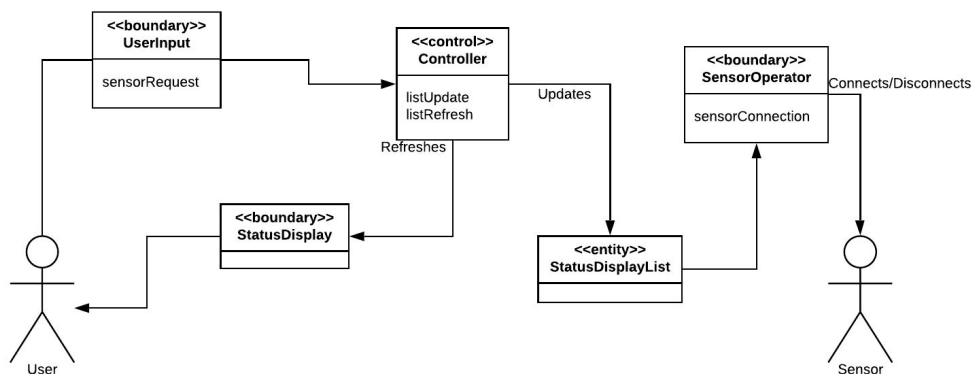
Periodically, the list must refresh in order to display an accurate reading of the sensor status. This is done by allowing the sensor operator to request an accurate reading from the sensors themselves. The attributes pertaining to these concepts are as follows: the IDChecker contains a validID parameter to check a credential match; the controller contains a numOfAttempts attribute to make sure the user does not exceed a certain number of ID-password attempts; the Status Display List contains a loadSensor attribute in order to efficiently load existing sensors; and the Sensor Operator contains a sensorStatus attribute to correctly obtain a sensor status for the application to list.



Add/Remove Sensor:

Assuming the user is logged into their account, this is how the domain model for modifying the sensor list works: First the user will indicate which sensor they want to remove/add. The Controller then updates the StatusDisplayList by removing/adding the particular sensor. Next, the SensorOperator disconnects from/connects to the desired sensor. Finally, the Controller refreshes the StatusDisplay using the new StatusDisplayList.

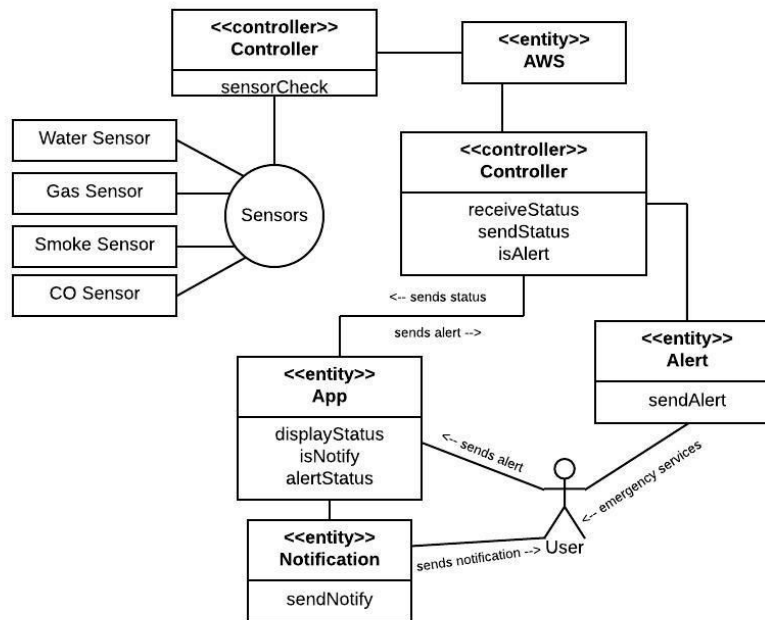
Regarding the attributes: the UserInput will have a sensorRequest attribute which indicates to the Controller what sensor they want to add or remove; the Controller will have a listUpdate attribute which will modify the StatusDisplayList when the user adds or removes a sensor, as well as a listRefresh attribute which will refresh the StatusDisplay after using the listUpdate attribute; the SensorOperator will have a sensorConnection attribute which will establish or destroy a connection to a particular sensor.



System Emergency Alert/Notification:

Assuming the user is logged into their account and connected to the various sensors through the network, the below diagram represents the domain model for emergency alert/notification of the system.

The four (4) sensors (water, gas, smoke, carbon monoxide) are all constantly checking for their respective issues. This data is then linked into AWS which will be accessible through the applications controller. The controller manages the data send to and received from the user-facing application along with the alert system. The controller will send the status of the sensor to application which in turn determines if the user needs to be notified or not. If so, the application sends the user a notification. When a user receives a notification, they have the option of telling the application to alert emergency services (police, fire dept., etc). From there, the application will send this alert to the controller. The controller then triggers the “isAlert” function which contacts the appropriate service. Emergency services can then take action from there in helping the user.



System Operation Contracts:

Name:	Connect
Responsibilities:	This operation must connect the user to the AWS server through the app.
Notes:	UC-2
Output:	A link is established between the App and the AWS Server.
Pre-Conditions:	App already downloaded and installed on the Android device. Account already set-up.
Post-Conditions:	App retrieves the data from the AWS server

Name:	Signal
Responsibilities:	This operation must send a signal from the sensor to the user by using the AWS server and the app.
Notes:	UC-4
Output:	The sensor is linked to the app by the AWS. This link allows the transfer of data.
Pre-Conditions:	A sensor is installed and connected to the AWS server
Post-Conditions:	Signal is captured and send from the sensor to the server

Name:	Check
Responsibilities:	Must update the current status of the sensors and display on the App for User to view.
Notes:	UC - 9
Output:	AWS Server requests status update from the Sensor, and the Sensor will respond with an updated status.
Pre-Conditions:	AWS Server and Sensor are already connected to each other.

Post-Conditions:	AWS Server will share the updated status.
-------------------------	---

Name:	Network Error
Responsibilities:	Must update the user if the sensors are down due to network error/ outage.
Notes:	UC - 11
Output:	AWS Server requests status update from the Sensor. After interval, no response recorded. AWS Server describes this occurrence as network error.
Pre-Conditions:	AWS Server and Sensor are already connected to each other.
Post-Conditions:	AWS Server will keep checking with the Sensor during intervals for an updated status.

Section 6: Project Size Estimation:

Actor Weight:

Actor	Description of relevant characters	Complexity	Weight
User	User is interacting with the system via an android app (when logging in/out, checking the sensors, adding additional sensors, adding users)	Complex	3
Sensor	Sensors pick up various signals around the house and send them to the server database.	Simple	1
Server	Stores data from the signals and the camera (if a break in occurs the camera recordings will be stored)	Average	2
Application	Application is a centralized interface that provides the user with the ability to check the status of their homes as well as sends notifications to the user when the sensor picks up a signal.	Average	2

$$\text{Unadjusted Actor Weight (UAW)} = 1 \times (\text{simple}) + 2 \times (\text{average}) + 1 \times (\text{complex}) =$$
$$\text{UAW} = 1 \times 1 + 2 \times 2 + 1 \times 3 = 8$$

Use Case Weight:

Use Case Description	Use Case Name	Category	Weight
Simple Interface, 3 steps for the main success scenario, 2 participating actors (user, application)	Download (UC-1)	Simple	5
Simple Interface, 3 steps for the main success scenario, 2 participating actors (user, server)	Connect (UC-2)	Simple	5
Average Interface, 4 steps for the main success scenario, 2 participating actors (user, application)	Disarm (UC-3)	Average	10
Average Interface, 4 steps for the main success scenario, 4 participating actors (sensor, user, application, server)	Signal (UC-4)	Average	10

application, server)			
Average Interface, 4 steps for the main success scenario, 2 participating actors (app, user)	Contacts (UC -5)	Average	10
Complex Interface, 4 steps for the main success scenario, 4 participating actors (app, user, server, sensor)	Notify (UC-6)	Complex	15
Average User Interface, 5 steps for the main success scenario, 2 participating actors (user, application)	New Accounts (UC-7)	Average	10
Average User Interface, 4 steps for the main success scenario, 3 participating actors (user, sensors, server)	Add (UC-8)	Average	10
Complex User Interface, 3 steps for the main success scenario, 4 participating actors (user, sensors, server, application)	Check (UC-9)	Complex	15
Simple User Interface, 3 steps for the main success scenario, 2 participating actors (user, application)	Control Alert (UC-10)	Simple	5
Simple User Interface, 3 steps for the main success scenario, 3 participating actors (user, server, application)	Network Error (UC-11)	Simple	5
Simple User Interface, 3 steps for the main success scenario, 2 participating actors (user, application)	Log Out (UC-12)	Simple	5

$$\text{UUCW} = 5 \times \text{simple} + 4 \times \text{average} + 2 \times \text{complex} = 5 \times 5 + 4 \times 10 + 2 \times 15 = 95$$

Technical Complexity Factor:

Technical factor	Description	Weight	Perceived Complexity	Calculated Factor (Weight' Perceived Complexity)

T1	Distributed, App based system	2	3	$2 \times 3 = 6$
T2	Users expect exceptional performance, notification of break-in, leaks, etc. should be instant	1	5	$1 \times 5 = 5$
T3	End-user expects efficiency	1	3	$1 \times 3 = 3$
T4	Internal processing is relatively simple	1	1	$1 \times 1 = 1$
T5	No requirement for reusability	1	0	$1 \times 0 = 0$
T6	Ease of install is moderately important (can be installed by the company technician)	0.5	3	$0.5 \times 3 = 1.5$
T7	Ease of use is very important	0.5	5	$0.5 \times 5 = 2.5$
T8	Portability of use (use on any smartphone connected to the internet)	2	5	$2 \times 5 = 10$
T9	Sensors should be easy to replace if they become faulty	1	3	$1 \times 3 = 3$
T10	Concurrent use is required; multiple users can access at once	1	4	$1 \times 4 = 4$
T11	Security is a concern	1	3	$1 \times 3 = 3$
T12	Only the user/s are able to access the account	1	0	$1 \times 0 = 0$
T13	No unique training needs, should be easy to set-up and use	1	0	$1 \times 0 = 0$
Technical Factor Total:				39

$$TCF = 0.6 + (TF/100) = 0.6 + .39 = .99$$

$$UCP = (UUCW + UAW) \times TCF \times 1 = (95 + 8) \times .99 \times 1 = \underline{101.97} \text{ or around } \underline{102}$$

Section 7: Plan of Work:

1) Goals after the first report 1 until 1st Demo:

Our first main deadline coming up will be the 1st Demo, that will be in 4 weeks

For a successful demo, we need the basics of our project running to show a proper output.

The 3 main parts of our project include design of sensors, hosting AWS Server, and App Development.

We connect each of these parts together using WiFi.

To have a working system created, we have set deadlines for the next 4 upcoming weeks:

Previous Weeks: Sensors have been purchased and ordered, all group members understand the project and what must be done for successful completion.

1st Week: Sensor Design, build the sensor prototype using sensor and WiFi module on Arduino board

2nd Week: AWS Server Connectivity, host the server and be able to connect it to the Sensor to store the data that it receives.

3rd Week: App development, get basic App interface working, connect to the AWS Server.

4th Week: System Linked, all parts of the project are connected and basic design of each model is completed. Have test Demo ready.

2) Goals after 1st Demo till the end of semester:

- Each group will be adding more sensors and grouping them together to be able to pick up a more accurate occurrence of an issue and send notification
- Advanced functionality from the app
- Complete synchronization of the system

3) Product RoadMap:

	Sensor Design Mar 1, 2019	AWS Connectivity Mar 9, 2019	App Dev Mar 16, 2019	DEMO 1 Mar 25, 2019	Q2 *	Grouping of Sensors Added Apr 6, 2019	More App Functionality Apr 17, 2019	Final Demo Ready Apr 22, 2019
✓ Product Design								
Sensor Design		AWS Connectivity	Application Development	System Linked/ Demo Ready		More Application Functionality Added	Final Demo Ready	
✓ Hazard Prevention Team								
				Add Other Hazard Sensors				
✓ Home Security Team								
				Add Home Security Sensors				

4) Product Ownership:

We have divided into 2 distinct units, one for dealing with home hazards and one for home security. This is a combined effort and the final product will be the combined work of both groups.

Hazard Prevention Group: Shaan, Shivum, Andy, Nathan, Kyle

Home Security Group: Harshil, Adarsh, Avi, Parth

5) Breakdown of Responsibilities:

Thus far we have spent our time as a team developing a project idea that is ambitious and is a project we are all ready to work hard to accomplish.

We are a team of 9 members, and we have further divided ourselves into 2 more smaller groups. We have split our team into two main groups, each designated to focus on a particular group of sensors. Below are the groups:

Hazard Prevention Group: Shaan, Shivum, Andy, Nathan, Kyle

Home Security Group: Harshil, Sri Jay Adarsh, Avi, Parth

Hazard Prevention Group will focus on the sensors that will prevent any major household hazards. Home Security Group will focus on the grouping of sensors to secure the doors and windows of the home. Each group respectively has been researching and developing ideas on which sensors to incorporate and how they will be set up to be useful.

We have now ordered the parts that we will be needing and will now work on the design aspect of the sensors.

Section 8: References:

- Software Engineering Fall 2013 project - Voice Control Based Home Automation System
- Software Engineering Spring 2012 project - autoHome
- <https://www.lucidchart.com/>
- <https://app.productplan.com/CugVKIXI#>
- <https://www.romanpichler.com/blog/10-tips-creating-agile-product-roadmap/>
- <https://blog.asana.com/2018/08/product-roadmap-tips-templates/>
- <https://www.ece.rutgers.edu/~marsic/Teaching/SE/proposal.html>
- <https://www.ece.rutgers.edu/~marsic/Teaching/SE/report1.html>
- <http://www.cs.ucc.ie/~adrian/cs560/UML4%20SysSeqDiag%20Contracts.pdf>
- https://www.androidcentral.com/sites/androidcentral.com/files/styles/xlarge/public/article_images/2016/02/fitbit-add-add-device.jpeg?itok=VdWTM11N
- https://lh3.googleusercontent.com/FIC7m6lZ9oZfV0w6hO1NzfgrY-ZCF8fx-jpSU28lbKGC8AxCji-_NdySTV_P-TrIxc8=h310
- https://cdn57.androidauthority.net/wp-content/uploads/2015/09/Screenshot_2015-09-14-17-59-06.png
- <https://en.wikipedia.org/wiki/FURPS>
- <https://app.productplan.com/CugVKIXI#>
- <https://www.romanpichler.com/blog/10-tips-creating-agile-product-roadmap/>
- <https://blog.asana.com/2018/08/product-roadmap-tips-templates/>