Mail-Direct Analysis and Design

Assignment 5- OSS Individual Project by Randell Carrido

Use Case Descriptions

Primary Actor: TCNJ Resident

Goal in Context: To notify residents whenever mail is placed in their mailboxes.

Preconditions: Server, mobile app, and Arduino system must be fully configured, appropriate usernames and passwords must be obtained.

Trigger: The resident receives a text notification, a push notification, or an email telling him/her that mail is currently in their mailbox or they manually check the app.

Scenario:

1. TCNJ resident logs on to the Mail-Direct app.
2. TCNJ resident enters username.
3. TCNJ resident enters password (at least 7 characters in length).
4. System either prompts user to input again if invalid input or brings the user to the main page.
5. At the main page, some signal/icon is shown in the center (green dot for “mail in the mailbox” and a green dot for “no mail in the mailbox”).
6. Resident sees a green dot and goes to pick up the mail.
7. Once mail is picked up, sensor detects nothing in the box and sends “no mail” data to the server.
8. App displays a red dot for “no mail.”

Exceptions:

1. ID or passwords invalid or unrecognized.
2. Arduino unable to connect to Wi-Fi.
3. Set phone number and email for text, push notifications, and email notifications are invalid.
4. Sensor unable to detect mail.

Frequency of Use: Once a week

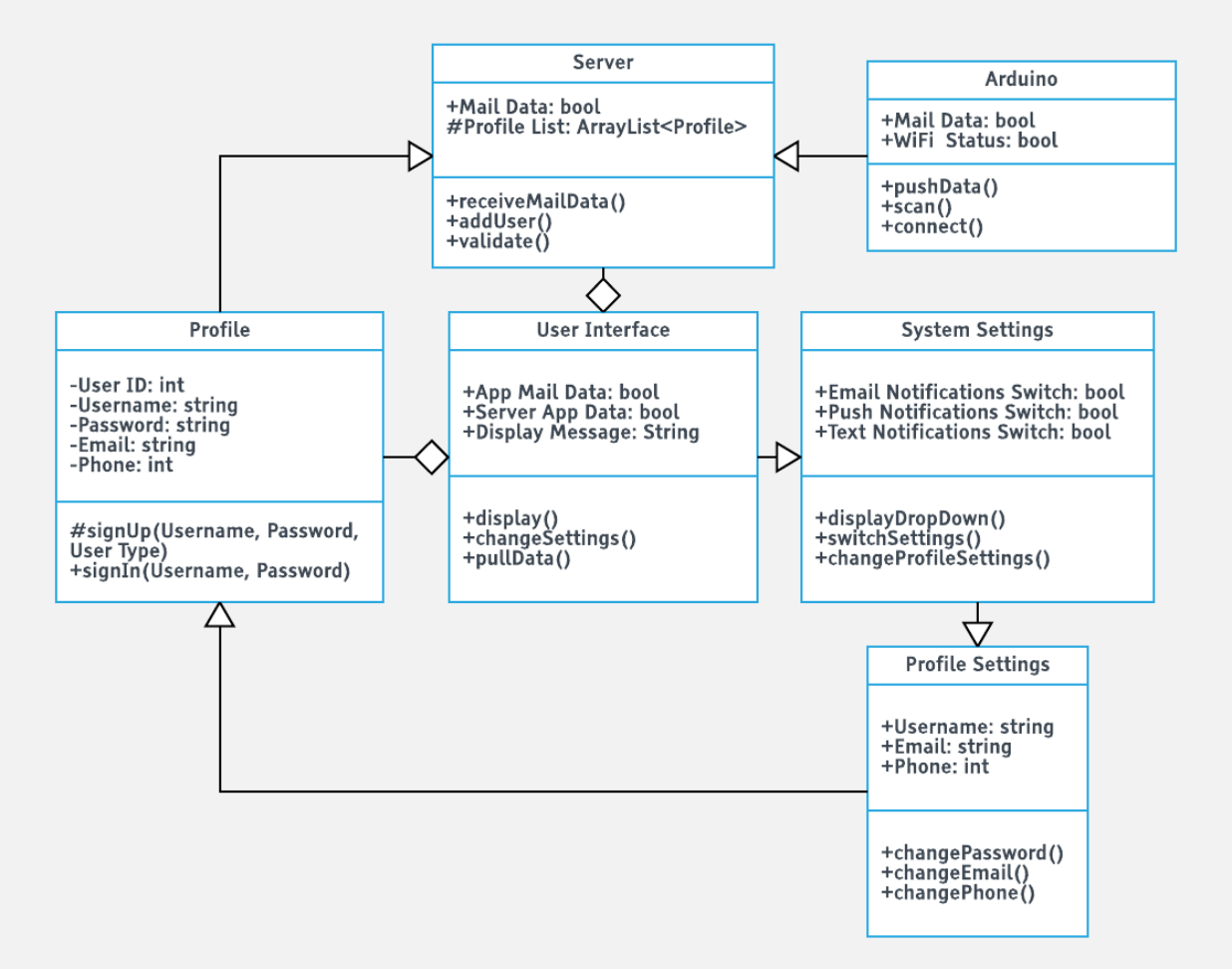
Secondary Actors:

1. System Developers/Admin
2. Arduino System

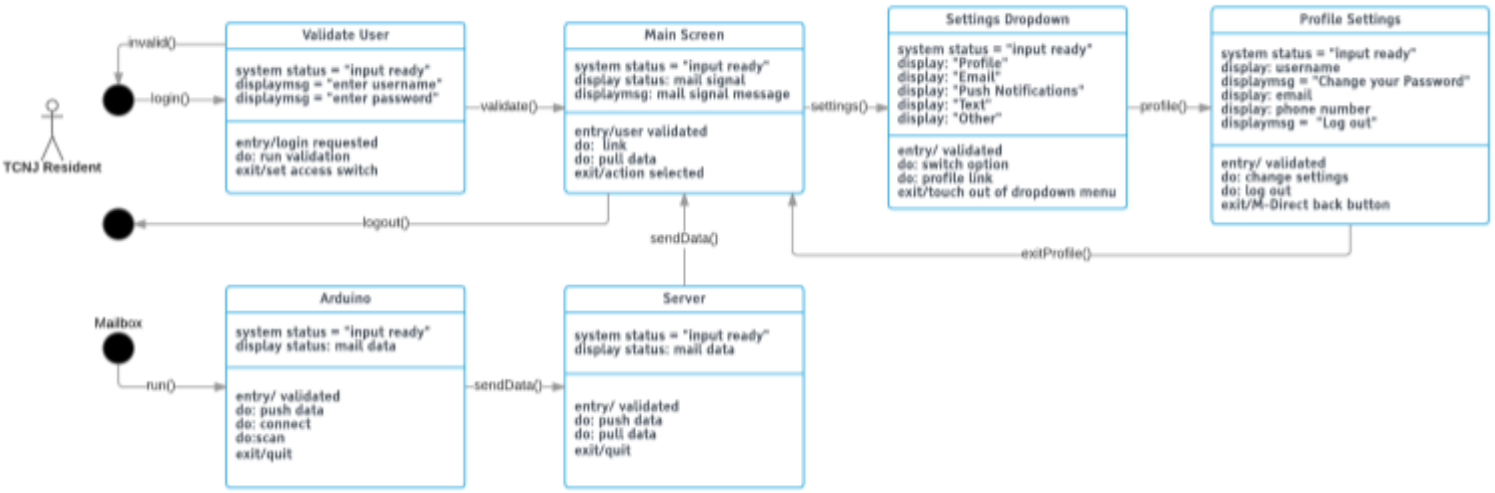
Open Issues:

1. What is the most optimal sensor to use for such a system?
2. Would the information given by the app create a possible security or privacy risk?
3. Can there be a way to automatically check for issues within the Arduino?

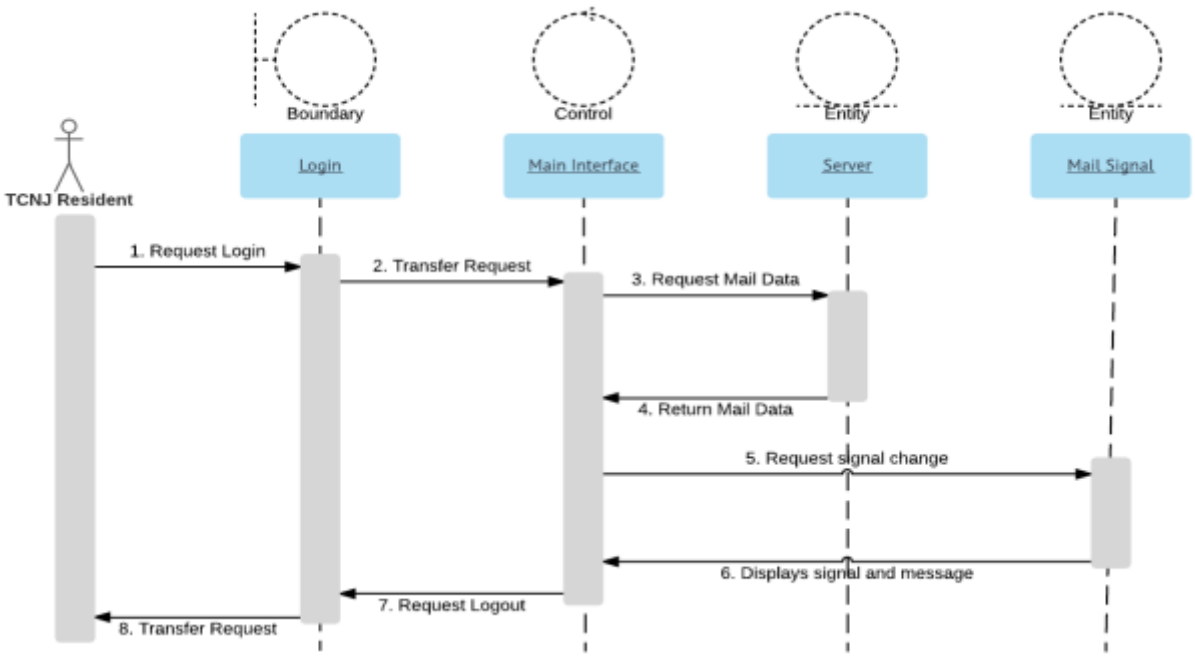
Detailed Design Class Diagram



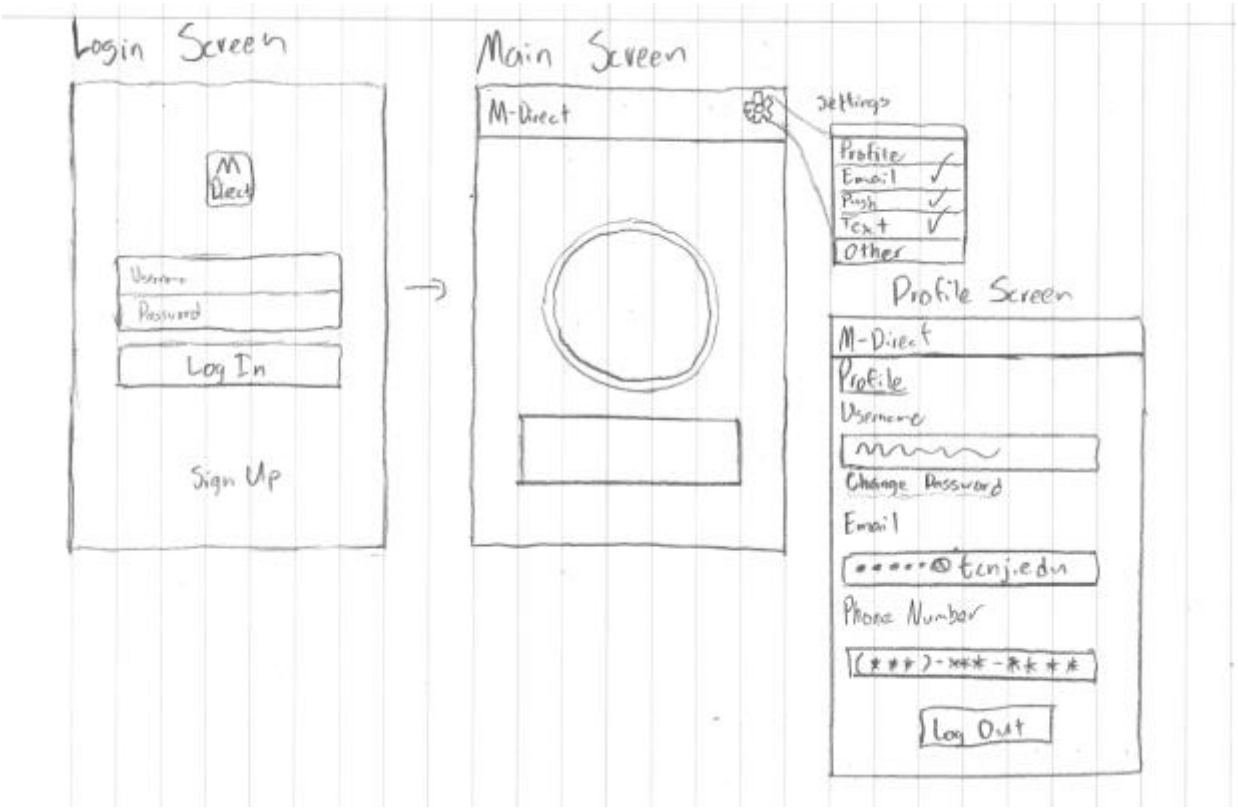
State Chart



System Sequence Diagram



User Interfaces



The main objective for the overall interface design was complete simplicity. To begin, the login screen is very straightforward, allowing the user to log in with a username and password or sign up. The Main Screen features a big circle in the middle which is referred to as the “mail signal” in which it turns green if there is mail in the mailbox and red if there is no mail in the mailbox. Under the mail signal will be a box that displays either the message “you have mail” or “no mail” according to the data received from the server from the Arduino sensors. On the upper right corner is a gear icon which is a dropdown menu for the settings. Here you can choose to allow email notifications, push notifications, or text notifications. You can access your profile settings through here, which has its own separate interface. It displays your Username and allows you to change your password, email, or phone number. You can also logout from this interface. It will be possible to implement a good interface based on the eight golden rules. For shortcuts, frequent users will be automatically logged in when opening the app. For informative feedback, users are told a certain message when requests are successful such as login or sign in requests and changing settings. For consistency, we would have to make an optimal implementation to have it consistent. Like informative feedback, design dialogues to yield closure will be included for when a request is done. For simple error handling, the system will be designed to be able to handle invalid login info, invalid inputs in the profile section, and invalid sign up info. For easy reversal of actions, the only time that would be necessary is

inputting email or phone number. In that case, the reversal of actions just requires retyping/easy adjusting. To reduce short-term memory load, we have the app be as simple and straightforward as possible, only having a signal and a message in the main screen. In all, with the ease of use, there will be guarantee of internal locus of control.

Back-End

To hold the different information of multiple accounts, I will be implementing an ArrayList of Account objects where each object will hold the following: user ID # (number generated through order of people signing up), username, password, email (optional), and phone number (optional). All this data within each object will be either set to private or protected to promote privacy and security through information hiding and encapsulation. This data structure will be stored within the server of the system. In the server itself, there will be a set Boolean variable that is used to hold the mail presence data. This data is periodically changed by the Arduino system and is periodically sent to the user's mobile device through the app. For traversing through the ArrayList, I can utilize the Java Collections Class Library methods, specifically using the indexOf() method.

Test Case Design

Functionality Tested	**Inputs**	**Expected Outputs**
Login Rejection	Incorrect Username or Password	Does not allow login
Login Success	Correct Username and Password	Allows login
Signup Reject	Already used username	Asks for another username
Signup Success	Unique username	Allows signup with username
Arduino system	Mail put or taken out of box	Sent Boolean signal to server
Mobile App Connectivity	Info stored in server	Mobile App receives info

Due to budget and time constraints, we will be testing the system of only a single mailbox which is shared by a single townhouse.

GitHub

Repository Name: MailDirect
Full Pathname: Randellcar/MailDirect