FridgeLocker

Mobile Middleware 2014
Project Report

Anna Kruglaia
simoroshka@gmail.com
014262834

Ayesha Ahmad
Ayesha.Ahmad@helsinki.fi
014109366

Ibbad Hafeez
Ibbad.Hafeez@helsinki.fi
014257292

Project: https://github.com/ayef/FridgeLockers.git

FridgeLocker – The App

The FridgeLocker App is an activity monitoring app developed for Android mobile phones with the Sensordrone. The target audience for this app are people who want to lose weight by limiting their food intake. Some people find it very hard to motivate themselves to limit their diet. During the day time, family members, friends and other activities can motivate them to stick to their routine but there are many cases where such a person is caught eating during the night hours when everybody is asleep. To counter such a situation, the FridgeLocker app has been created using a Sensordrone device which is placed inside the refrigerator. The FridgeLocker app is polling the Sensordrone and will detect that the refrigerator has been opened by the user. The App will then take several actions based on the user's settings. The list of implemented features is shown in the table below:

Feature	Status	Description
Communication of app with sensor drone	✓	Sensor drone API is communicating with drone. Temperature, humidity and luminance values are being retrieved and displayed.
Data upload from app	✓	Data being stored in SQLite database on Android device. This data can be shifted to the cloud in further development.
Communication with web service- Twitter	✓	Service for sending tweet to the web server developed and working. Some improvements required (see 'further work' section).
Data analysis – Statistics	✓	Basic data analysis: Using the local SQLite DB which contains entries for two users 'Alice' and 'Bob', we provide a simple aggregation of data: **user |date | number of violations** grouped by the user and date.
Use of Mobile Phone Sensor	✓	The App rings the user's alarm for some seconds when it detects a fridge violation
Background Service for polling sensordrone	✓	Service polls sensordrone at some user defined interval of time. Upon detection of violation it performs actions such as ringing alarm, sending a shaming tweet, and logging data.
Turning the background service on or off	✓	A useful feature is the ability to turn off the background polling service, thus stopping the monitoring of the fridge.
User-defined settings	✓	SharedPreferences used to store user's preferences. Additional settings that could be added: User's FridgeLocker nick

With some further development this application can be used in several other scenarios where some event is to be monitored e.g. parents can use it to monitor their kids sneaking in the refrigerator at night. In a departmental store, it can be used to get stats of some cold storage compartment and logs the time when it was opened, what was the temperature, humidity level and for how long the storage compartments was kept open, how temperature dropped during this time etc. In case of home kitchen, this app can be used to communicate with a Sensordrone installed near the stove or oven which sends the is polled by the user's phone which then generates and alarm in case some temperatures are exceeding or time for cooking at specific temperature is done. Hence this app is designed to be flexible enough for making slight modifications to be used in different scenarios.

Further Work

There are several improvements that can be made in the App that were planned but could not be completed in the given timeframe.

1. The user interface needs to be improved with larger fonts and formatting of the display. The connected/not connected informational messages are not being updated.
2. The user's preferences are stored in SharedPreferences but not all of these preferences are read from the file – there are still some hardcoded values in the code that must be replaced with values from SharedPreferences.
3. If the user changes the Settings, they need to exit the settings page and re-enter the settings page before the Settings' changes take effect. We planned to add a Sharedpreferences update listener to fix this.
4. Secret tokens for the Twitter user account are hardcoded in the code. The interface for allowing the user to authorize FridgeLocker to post on the user's Twitter account has been fully developed in a separate app but not integrated with FridgeLocker. That twitter app is attached as 'MangoTweet.rar' file
5. The post tweet functionality causes the 'Violation' activity to start which lets the user know they've tweeted. This is useful for debugging but should be removed on App release as it is an irritating feature.
6. Posting on the Facebook page of FridgeLocker was planned but not implemented. It should be relatively easy to implement as it is technically similar to the tweeting functionality
7. FridgeLocker as a Social Networking App: A FridgeLocker community that can share information and support one another

FridgeLocker Application Design

The basic working of FridgeLockers app uses the Sensordrone. The App Sensordrone will measure light, temperature and humidity values after every few seconds when polled by the mobile device. When a fridge violation is detected (the fridge has been opened during banned times), the mobile device can do several things: 1. Ring the alarm, 2. Tweet a shaming message on the user's behalf on their twitter feed, and 3. Store the data in a Server. Here are some screenshots of the App interface:

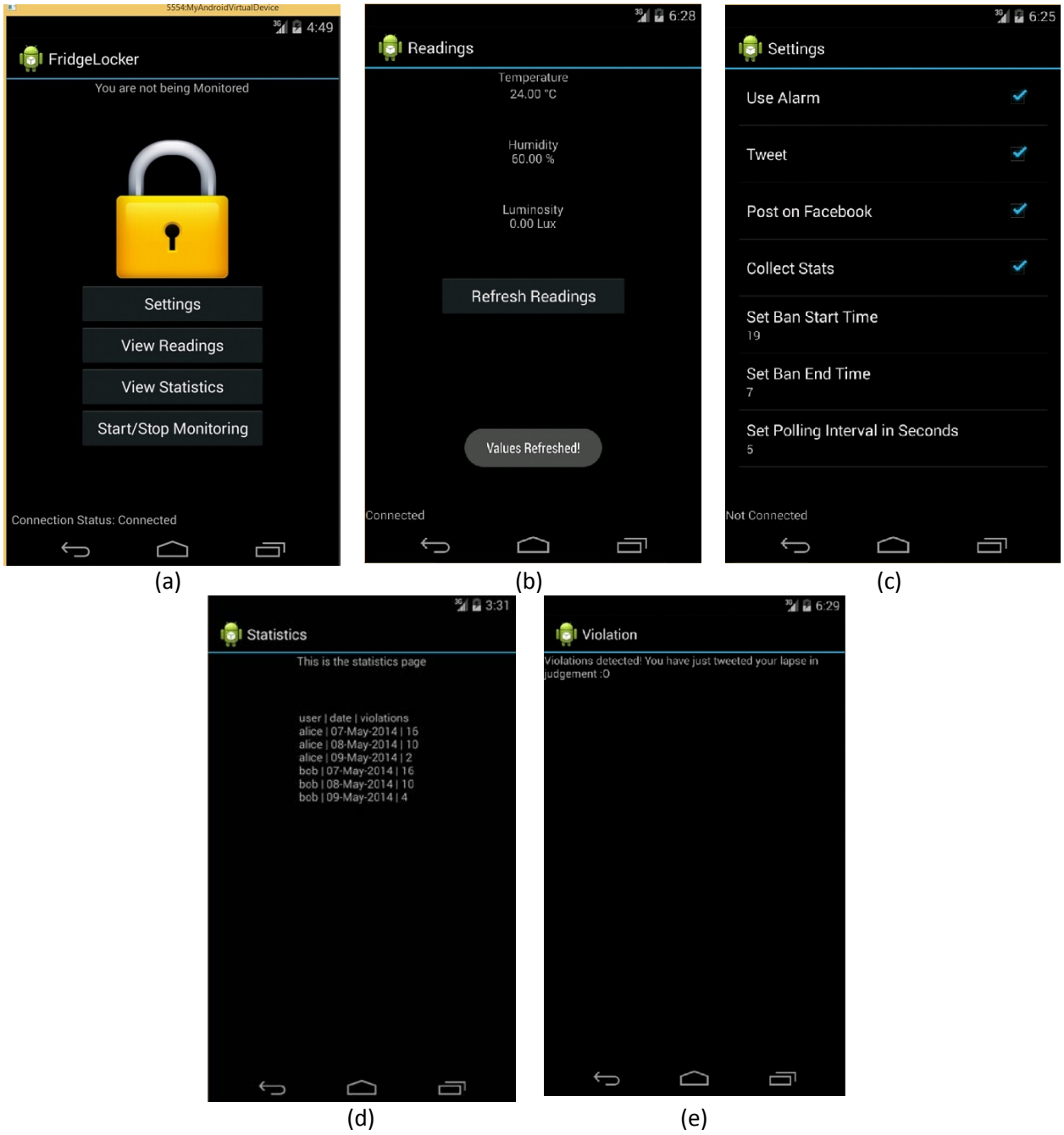

Figure 1: a) The main menu b) Readings page c) Settings Page d) Statistics Page e) Page that appears on tweet

Since all of the tweet application could not be integrated into FridgeLocker, here are some screenshots from the Tweet app which is complete: Figure 2 shows the working of MangoTweet – the application that was planned to be fully integrated into FridgeLocker, also developed by a member of this group. First the user logs in the application and authorizes it to post tweets on his behalf from application. User authentication is implemented using OAuth request and communication is done over secure https

connection. After logging in, user tweet from application can be seen in Figure 2. Test account is used for this task but user can login any account he wants to.

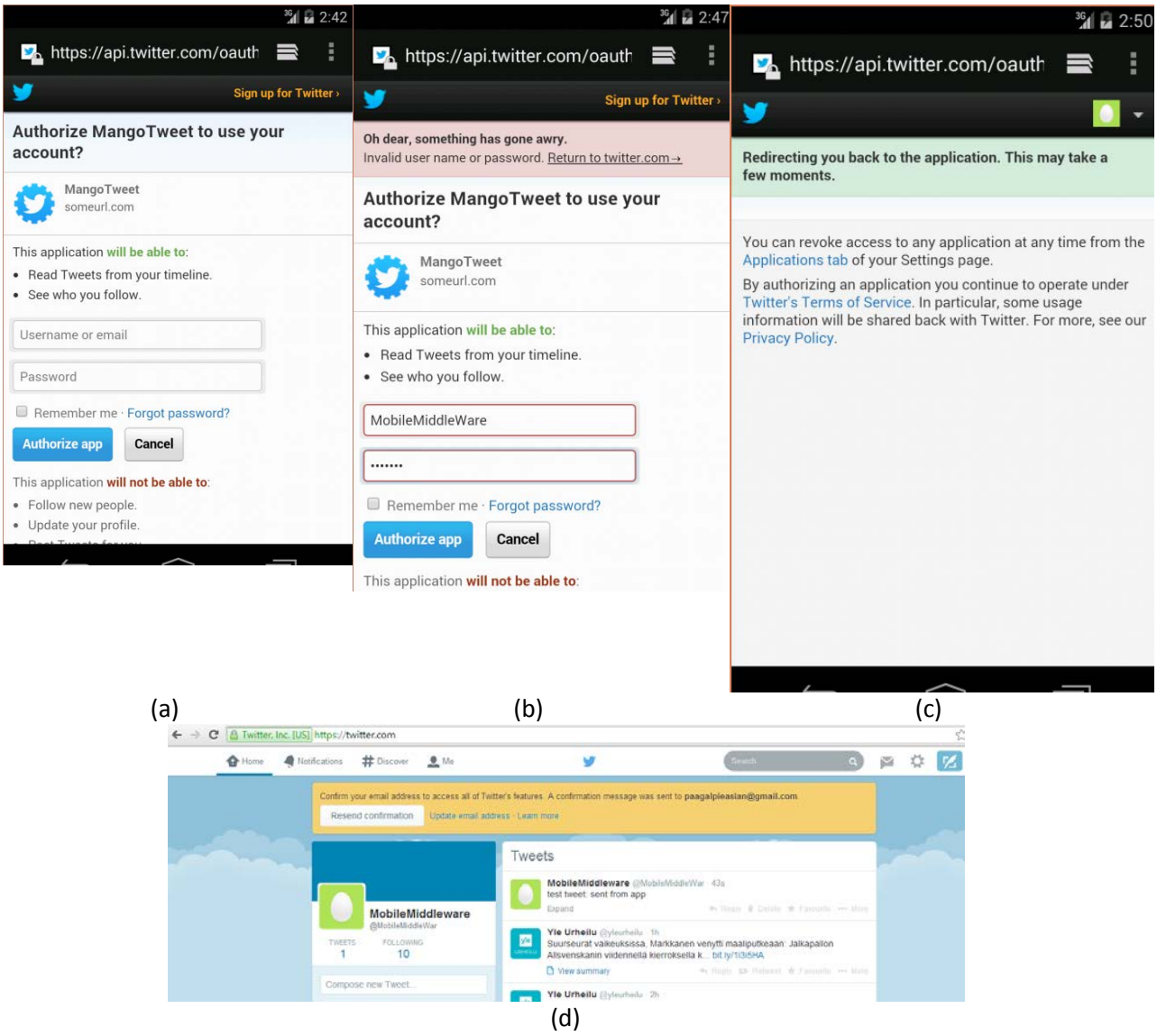

Figure 2: a) User binds the profile to twitter b) User authorizes the app to tweet on his behalf c) User account logged in d) Tweet posted