Information Systems 1D Report - Team 2D

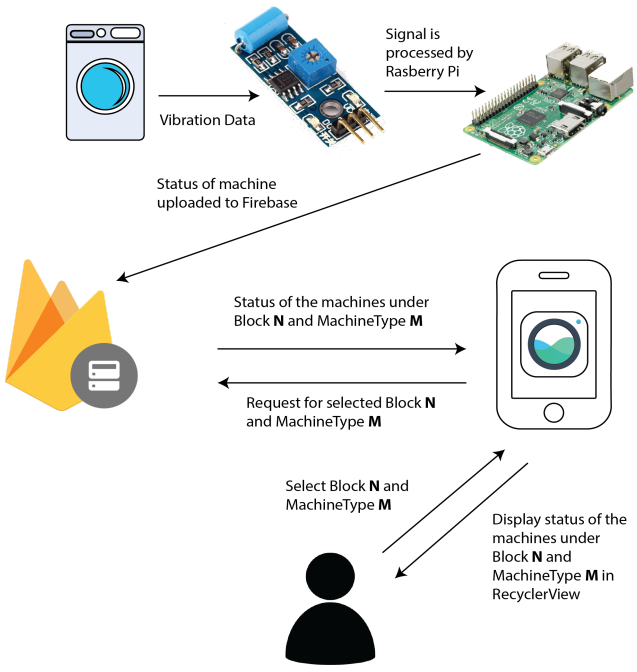
1. Background

Hostel residents depend on communal laundry machines to wash their dirty laundry and keep their clothes clean. However, there are limited machines available and the whole cleaning and drying process is long, which often causes machines to be unavailable when they are needed. To give an example, a resident wants to wash his clothes before returning home for the weekend. He heads down to the laundry room but realizes that all of the machines are occupied, which necessitates a rescheduling of his plans since he would have to wait for a machine to be available in order to do his laundry.

Not knowing about the availability of hostel laundry machines may result in unexpected delays in residents' laundry plans, causing time to be wasted. Currently, the only way to know if there are vacant machines, or if not, how much time is left until a machine will be available, is by physically going down to the laundry room to check. This also means that at times, hostel residents might procrastinate on doing their laundry simply due to the hassle of needing to go down to check on the availability of the machines. In light of these problems, we have developed WashWatch, an app that delivers real-time laundry machine information straight into hostel residents' hands, bringing convenience and efficiency to their laundry experience.

2. System Design and Implementation

2.1 System architecture



Accounting for the possibility of hostel facility management replacing machines or changing machine models periodically, we have decided that our machine data collection system would be implemented through an external hardware attachment. This would eliminate the hassle of reprogramming the interface with the new machines whenever machine changes are made.

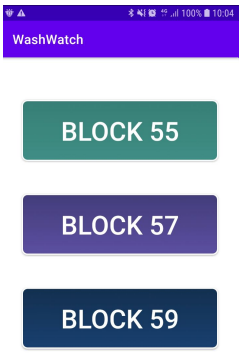
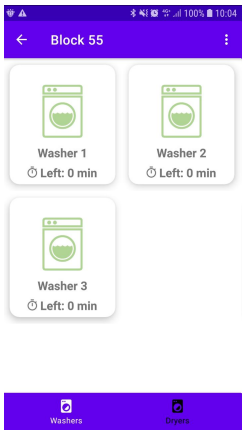
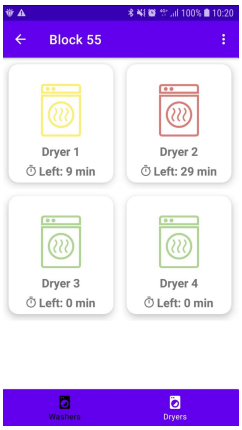
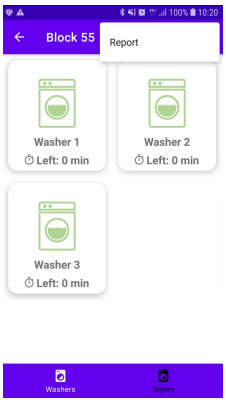
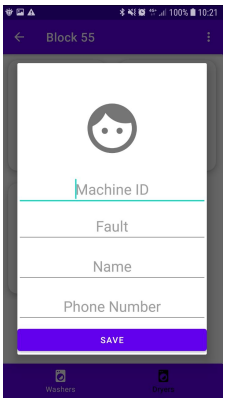
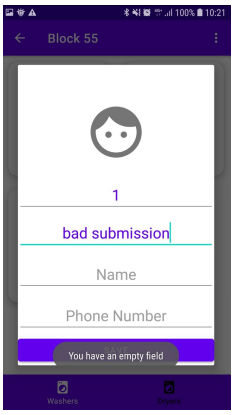
The external attachment consists of a Raspberry Pi hooked up to a vibration sensor to capture the status of the machine. If the machine is running, the vibration sensor will pick up that a new laundry cycle has started and trigger a countdown timer. Since all the laundry machines in SUTD hostel run on a fixed time, the countdown timer will be set to the default laundry cycle time of each machine.

The collected information is stored in Google Firebase to allow for easy retrieval by our app. Using a real-time database, it gives instant updates on the status of the machine in the block selected by the user.

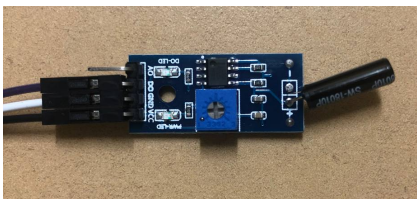
We designed the system architecture to be modular so that if there is a need for future upgrading or migration, such processes can be done easily.

Information Systems 1D Report - Team 2D

2.2 App screenshots

<table>
<tr><td colspan="3">Process to view status of machine for Block 55 for Washers and Dryers</td></tr>
<tr>
<td> 
</td>
<td> 
</td>
<td> 
</td>
</tr>
<tr><td>Main Activity</td><td>Washers in Machine Activity</td><td>Dryers in Machine Activity</td></tr>
<tr><td colspan="3">Process of reporting fault in the machine</td></tr>
<tr>
<td> 
</td>
<td> 
</td>
<td> 
</td>
</tr>
<tr><td>Report Button</td><td>Report Pop Up Form</td><td>Checking User Input</td></tr>
</table>

2.3 Hardware used

<table>
<tr>
<td> 
</td>
<td>  </td>
</tr>
<tr><td>Raspberry Pi</td><td>Vibration Sensor</td></tr>
</table>

2.4 Design patterns used

1. Singleton to instantiate Firebase database

- The constructor of the Firebase database is declared as private. To access the Firebase database, programs use the public `getInstance()` method. If the database has not been created before, it creates a new instance of the database, else it returns the existing database.
- This ensures that there is only one database instance throughout the app. As a result, there is only a single point of access to the database, making it easier to maintain.
- It also allows future developers to modify one Java file to migrate the app to a new database.

2. Adapter and Delegation Design Pattern in RecyclerView
    - `MyAdapter` is used for `RecyclerView` to bind the image and text data obtained from the Firebase database to the view that is controlled by `ViewHolder`. The view item is set as a default machine card, whose machine image, title and time remaining text can be changed based on the data received from the database.
    - This design pattern makes the code modular, reusable and more dynamic as it removes unnecessary repeated code to display multiple instances of a machine card.

2.5 Other key technical features

1. Firebase Real-time Database
    - Firebase is a free and simple NoSQL database to connect to our mobile app. We first integrated the Firebase Real-time Database SDKs and set up the references to query the database. The data is in JSON format and the data is synchronized in realtime to every connected client.
2. Pull Down to Refresh Framework
    - Nested the RecyclerView within a SwipeRefresh layout. When the app is swiped downwards in the Machines page, the onRefresh function is called to update the page, by calling the pullFromCloud function from Firebase. Similar to how a bus arrival app works, this allows the user to refresh the page to get the latest updated time left for the machines to be available.

3. Members' Contribution

Member (Student ID)	Contribution
Wee Neville (1005035)	Making of video to convey the project
M S Subesh Kumar (1005141)	Linking Raspberry Pi with the sensor and the uploading the information to Firebase
Cheong Cher Lynn (1005458)	Presentation, Poster
Dhanush Kumar Manogar (1005010)	Linking Raspberry Pi with the sensor & uploading information to Firebase
Shawn Choo (1005128)	Implemented the RecyclerView
Wang Siyang (1005485)	Ratifying the app for bugs discovered during checkoff

Information Systems 1D Report - Team 2D

Ivan Feng Jun Kai (1005269)	Implementation of Singleton to retrieve information from Firebase

4. Possible Future Work

- Notification system
    - The user could set the app to send a notification when a washer/dryer is available, in the event that all are occupied.
    - The user could set the app to send a notification when a particular washer/dryer has finished a cycle, which would help the user to collect their laundry as soon as it is ready.

5. Conclusion

In today's age of information, smart solutions involving the Internet of Things have been gaining popularity, where technology is increasingly being employed to improve the quality of our lives. Our app, WashWatch, bridges the frustrating gap in the current hostel laundry system by allowing residents to obtain real-time laundry machine statuses at the click of a button, wherever they are. We have developed a minimum viable product (MVP) that has been able to showcase proof of concept of our app. In developing this MVP, we have made numerous design considerations and tapped into our learnings of good design patterns, allowing our app and product to be modular and reusable. This makes it easy for future developments to be made to the app, should anyone decide to pick it up and bring it to the next level. Although there is always room for improvement, WashWatch, with its current functionalities, has had a positive reception among hostel residents, who cite the usefulness of its functionalities in bringing convenience and efficiency to the laundry process and the simplicity of its interface as its greatest selling points. As such, we hope that this project is able to gain enough traction to make a tangible impact on the SUTD hostel community, and beyond.

Link to Github: https://github.com/pinkadotted/WashWatch