|  |
| --- |
| Gurkaran Padda, Ricky Ramnath, Mohamed Kore |

Humber College Institute of Technology & Advanced Learning Parts Crib Management System CENG35-0NB

# Declaration of Joint Authorship

We, Gurkaran Padda, Ricky Ramnath, and Mohamed Kore, confirm that this work submitted is the joint work of our group and is expressed our own words. Any uses made within it of the works of any other author, in any form (ideas, equations, figures, texts, tables, programs), are properly acknowledged at the point of use. A list of the references used is included. The work breakdown is as follows: Each of us provided functioning, documented hardware for a sensor or effector. Student A provided the RC522 RFID sensor. Student B provided Adafruit Ultimate GPS Breakout. Student C provided COM-14662 12-Button Keypad. In the integration effort Student A is the lead for further development of our mobile application, Student B is the lead for the Hardware, and Student C is the lead for connecting the two via the Database.

# Proposal

We have created a mobile application, worked with databases, completed a software engineering course, and prototyped a small embedded system with a custom PCB as well as an enclosure (3D printed/laser cut). Our Internet of Things (IoT) capstone project uses a distributed computing model of a smart phone application, a database accessible via the internet, an enterprise wireless (capable of storing certificates) connected embedded system prototype with a custom PCB as well as an enclosure (3D printed/laser cut), and are documented via this technical report targeting OACETT certification guidelines.

Intended project key component descriptions and part numbers  
Development platform:   
Sensor/Effector 1: RC522 RFID sensor  
Sensor/Effector 2: Adafruit Ultimate GPS Breakout  
Sensor/Effector 3: COM-14662 12-Button Keypad

We will continue to develop skills to configure operating systems, networks, and embedded systems using these key components to allow the parts crib to keep an electronic catalogue of items loaned out to students. Students will be able to use a mobile android-based application to see what items are available and to request them using their student credentials. The Parts Crib staff will be able to use the admin page through a website to manage inventory and track outstanding items. There will be a common datastore (using Google Firebase) for both the android app and the admin panel to ensure that both students and staff are getting the most current information on the state of the inventory. Sensors will enhance the overall process of borrowing items

from the parts crib, the RFID will keep track of data such as student number, name, email addresses and a randomly generated pin, which will allow us to keep track of who takes what from the crib and for easy identification. The RFID will be a much faster and more waste proficient than the traditional paper/student id method that we are currently accustomed to. The Adafruit Ultimate GPS Breakout will allow us to bring students to the location of the Parts Crib. And finally, the COM-14662 12-Button Keypad will server as a multi factor authentication.

Our project description/specifications will be reviewed by, Vlad Porcila, ideally an employer in a position to potentially hire once we graduate. They will also ideally attend the ICT Capstone Expo to see the outcome and be eligible to apply for NSERC funded extension projects. This typically means that they are from a Canadian company that has been revenue generating for a minimum of two years and have a minimum of two full time employees.

The small physical prototypes that we build are to be small and safe enough to be brought to class every week as well as be worked on at home. In alignment with the space below the tray in the Humber North Campus Electronics Parts kit the overall project maximum dimensions are 12 13/16" x 6" x 2 7/8" = 32.5cm x 15.25cm x 7.25cm.

Keeping safety and Z462 in mind, the highest AC voltage that will be used is 16Vrms from a wall adapter from which +/- 15V or as high as 45 VDC can be obtained. Maximum power consumption will not exceed 20 Watts. We are working with prototypes and that prototypes are not to be left powered unattended despite the connectivity that we develop.

# Executive Summary

This document aims to describe a breakdown of the implementation plan of the Parts Crib Management System, which is designed to simply allow students to easily borrow parts from the campus parts crib, at a faster and efficient rate compared to the current implemented system. Our implementation plan is as follows: Students will be required to come to the Parts Crib to have a RFID sticker placed on. This sticker will be uniquely tied to their student number as part of the onboarding process. The student will install the application using the Google Play Store, and must be logged on the Humber network to register their student number on to the service. The first loan by the student will require physical verification of the student ID. Our product should be purchased because it aims to nullify any of the current system problems, all while being cost efficient. Since we are also students at Humber College, aiming to create a better Parts Crib Management System will also benefit us and future students, and the employer can use it as an example to demonstrate what future students from this institution are capable of.

Contents

[Declaration of Joint Authorship 2](#_gjdgxs)

[Proposal 3](#_30j0zll)

[Executive Summary 5](#_3znysh7)

[List of Figures 11](#_2et92p0)

[1.0 Introduction 13](#_tyjcwt)

[1.1 Scope and Requirements 15](#_3dy6vkm)

[2.0 Background 17](#_1t3h5sf)

[3.0 Methodology 21](#_4d34og8)

[3.1 Required Resources 21](#_2s8eyo1)

[3.1.1 Parts, Components, Materials 21](#_17dp8vu)

[3.1.2 Manufacturing 23](#_3rdcrjn)

[3.1.3 Tools and Facilities 27](#_26in1rg)

[3.1.4 Shipping, duty, taxes 29](#_lnxbz9)

[3.1.5 Time expenditure 31](#_35nkun2)

[3.2 Development Platform 33](#_44sinio)

[3.2.1 Mobile Application 33](#_2jxsxqh)

[3.2.2 Image/firmware 37](#_1ci93xb)

[3.2.3 Breadboard/Independent PCBs 39](#_3whwml4)

[3.2.4 Printed Circuit Board 41](#_1pxezwc)

[3.2.5 Enclosure 43](#_147n2zr)

[3.3 Integration 43](#_23ckvvd)

[3.3.1 Enterprise Wireless Connectivity 44](#_ihv636)

[3.3.2 Database Configuration 44](#_32hioqz)

[3.3.3 Security 44](#_1hmsyys)

[3.3.4 Testing 44](#_41mghml)

[4.0 Results and Discussions 45](#_2grqrue)

[5.0 Conclusions 47](#_vx1227)

[6.0 References 49](#_3fwokq0)

[7.0 Appendix 51](#_1v1yuxt)

[7.1 Firmware code 51](#_4f1mdlm)

[7.2 Application code 51](#_2u6wntf)

# List of Figures

[Figure 1 HomeActivity 35](#_3j2qqm3)

[Figure 2 Database 35](#_3j2qqm3)

[Figure 3 Login Activity 35](#_1y810tw)

[Figure 4 Loans Activity 35](#_1y810tw)

[Figure 5 Request Activity 36](#_4i7ojhp)

[Figure 6 Parts List 36](#_4i7ojhp)

[Figure 7 & 8 Timed Request Activity 36](#_2xcytpi)

[Figure 8. Initial schematic. This work is a derivative of "http://fritzing.org/parts/" by Fritzing, used under CC:BY-SA 3.0. 40](#_2bn6wsx)

[Figure 9. This work is a derivative of "http://fritzing.org/parts/" by Fritzing, used under CC:BY-SA 3.0. 40](#_qsh70q)

[Figure 10. Breadboard prototype. 41](#_3as4poj)

[Figure 11. PCB design This work is a derivative of "http://fritzing.org/parts/" by Fritzing, used under CC:BY-SA 3.0. 42](#_49x2ik5)

[Figure 12. Humber Sense Hat Prototype PCB. 42](#_2p2csry)

[Figure 13. Example enclosure. 43](#_3o7alnk)

# 1.0 Introduction

This technical report presents the methods and processes of how we achieved our project result. Knowing who arrives, borrows parts and having easy access to our inventory are key parts of the parts crib management solution. Some of this projects policy require that you track how long students are borrowing parts for or that all students participating return each part at a certain time of day that is specified. This will additionally be able to help the administrative side by verifying that individuals borrowing parts are Humber College students that have an official student ID card. By recording sign-out times, it’s easy to track who borrows parts when and attend to students who are over their time limit or that have not yet returned said parts. Having a GPS sensor to track activity from a certain parts crib location and prioritize orders will also help with an efficient way of managing the parts crib. To further enhance security, the keypad will allow students to enter the specific code that will be generated by the application to ensure verification of Humber students. Finally, the RFID scanner will be a critical component of this project to help keep track of inventory by uploading information to the database for a more accurate count and to easily monitor activity. The application will provide a precise GPS location of the parts cribs, inventory status, and peak times for labs. The application will also allow students to create a cart of parts they will need for their labs and place orders from anywhere with a Wi-Fi connection. This system is meant for loan out companies and small businesses with inventory that will need to be loaned out to employees or customers. Specifically, this system will be prototyped and used by the Humber College Parts Crib.

## 1.1 Scope and Requirements

The parts crib management system will implement a system that will allow the parts crib to keep an electronic catalogue of items loaned out to students. Students will be able to use a mobile android-based application to see what items are available and to request them using their student credentials. An RFID Scanner, GPS Sensor, and a Keypad System will allow us to achieve this. Our goal is to make this a more efficient and easier way of borrowing parts from the parts crib. It is a goal of our group to make this project industry worthy for any application. We are assuming we’re capable of putting RFID tags on everyone’s student ID in the parts management system. We can also assume that the inventory of the parts crib is given to us so we’re able to put it into our database for when it comes to distribution of parts. We’re dependent on the Admins at the parts crib to be able to read the system and be able to handle the intake of item requests given to them by the users.

Report

/1 Hardware present?

/1 Introduction (500 words)

/1 Scope and Requirements

/1 Background (500 words)

/1 References

# 2.0 Background

The Humber College Parts Crib is located in a hallway in the J building at Humber College, it’s situated near the main series of labs, as well as classrooms, used by students of the Electrical/Electronic/Computer Engineering Technology programs. The current lending system was implemented by Vlad Porcila to address issues he saw with the lending process at The Parts Crib. The system requires that students make a request on a network-connected school computer using an application. This request is then made visible in a queue of requests to Parts Crib staff. The student is then expected to come to the parts crib, present their school ID and pick up the requested parts which will have been placed in a clear bag with an RFID tag assigned to that student.

The main concerns that the Parts Crib Staff feel need to be addressed, are the amount of time spent verifying student identification, notifying students when their parts are ready to be picked up (to avoid unnecessary crowding by students who are waiting for their request to be fulfilled), a way to prioritize orders based on when a student might need the parts, and the ability to retrieve historical data.

The challenges faced by the Parts Crib are similar to those encountered when handling lending services at libraries and item tracking at logistics companies. IoT technologies are being adopted at an increasing rate in supply chain management. The key technologies being deployed are RFID tag systems, cloud computing and IoT applications allowing multiple devices to communicate with one another Ben-Daya, Hassini, Bahroun 2017). One of the most common configurations used by companies for RFID tracking of shipments is to have each RFID tag assigned to a case/pallet (Zelbst, Sower 2012. p.73). This is analogous to the bagging system currently used by The Parts Crib, and with information about the items contained in the bag being stored in a cloud database it minimizes the number of RFID tags required.

In order to limit the interaction between a student and the Parts Crib staff during the process of obtaining loan requests we will require students to have a RFID tag affixed to their Humber student ID and the web/android application will generate a 5 digit pin for each request that they’ll be required to enter after tapping their card. This system is known as a *smart card based two-factor authentication scheme* and has the advantage of being easy to scale and cost-effective (Hussain et al. 2019). As the student will return the bag using the RFID tag, the only interaction between staff and students will be the when the staff hand the designated bag of items to the students.

We would like to thank Kristian Medri, Austin Tian for providing us with support during the writing of this report as well as overseeing our work on the application/website and physical prototype and Vlad Porcilo for meeting with us during project milestones to provide further input into features The Parts Crib staff would like to see.

# 3.0 Methodology

## 3.1 Required Resources

Report

/1 Parts/components/materials (500 words)

/1 PCB, case (500 words)

/1 Tools, facilities (500 words)

/1 Shipping, duty, taxes (250 words)

/1 Working time versus lead time (250 words)

### 3.1.1 Parts, Components, Materials

There will be many parts, components, and materials that will be used in this project. Firstly, there will be the sensors which include; the radio frequency identification scanner, global position sensor, and a twelve-button keypad. Each sensor and effector will be a key asset to this project. Some materials that will be used will include, laser cut acrylic, fiberglass epoxy resin with a copper foil bonded on to one or both sides which is what the printed circuit board is made of, and acrylonitrile butadiene styrene plastic for the three-dimensional printed parts that will be used with the acrylic to make a feasible enclosure to house our sensors. For our development platform, we will be using a raspberry pi three model b as well as a Nucleo STM32 microcontroller. The Nucleo stm32 microcontroller will be used alongside with the twelve-button keypad and the raspberry pi 3 model b will be used to accommodate the radio frequency identification scanner, global position sensor and the Nucleo STM32 microcontroller to use the twelve-button keypad. This will all be housed in the enclosure made from acrylic acrylonitrile butadiene styrene plastic. Some extra components will be a registered jack-forty-five Ethernet cable with an Ethernet to universal serial bus adapter. This will allow the Broadcom development platform to connect to a laptop via a remote desktop connection to configure settings for the sensors and effector. Some additional accessories that will enhance the features of the project will include; radio frequency identification tags for attaching to parts to scan in or out for inventory control, and an external active global position sensor antenna that will allow the sensor to have a boosted signal to get a position fix more faster and efficiently. The Two-Factor authentication will be implemented using both a soft and hard entry method. The user will be able to enter the five digits on the android device when prompted or enter them manually using the physical keypad. The keypad will itself be connected to a Raspberry Pi through the proprietary universal serial bus port with the STM32 microcontroller. The radio frequency identification scanner and the global positioning sensor will both be connected to the same Raspberry Pi using its general-purpose input/output pins and will both interface with the Real-Time database which will be used to retrieve both Student Information for radio frequency identification tags and current locations from the on-board global position sensor on their phone. The radio frequency identification scanner information will already have been loaded onto the database at the time of onboarding at the parts crib while the global position sensor coordinates will be pulled from the student’s phone when appropriate. An overall combination of these parts, components, and materials will lead to a successful build and presentation of the project. In order to be able to prototype this project and work on it, we will be using the resources provided by the Humber Colleges Prototype lab.

### 3.1.2 Manufacturing

For the Hardware Production Technology course taken by Computer Engineering Technology students in their 5th semester, students were tasked with designing and fabricating a Printed Circuit Board (PCB) and an enclosure for their sensor/actuator and development board/PCB.

For the PCB, we used Fritzing, an open-source hobbyist CAD program to design the breadboard connections and PCB as well as to produce the gerber files for printing the PCB. These gerber files would then be emailed to the Humber Prototype Lab where they would be laser etched onto a copper-coated plate of fiberglass. Following this, we solder header pins on to our sensors/actuators and header sockets onto the cut out holes on the PCB as required. Before the sensors/actuators are connected and powered on, each socket and pins is tested for continuity and power/ground connections are tested to see if voltage range is acceptable.

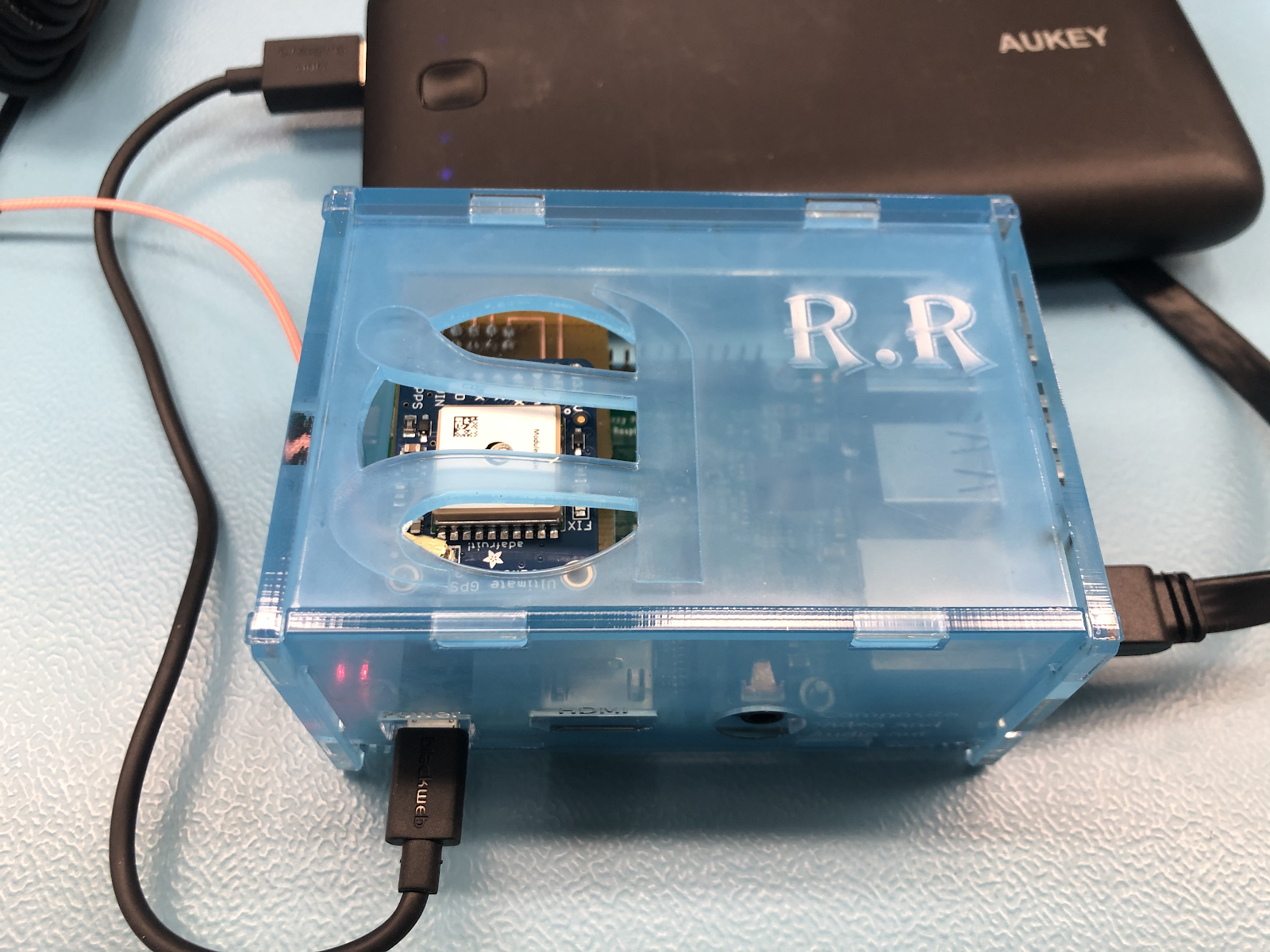
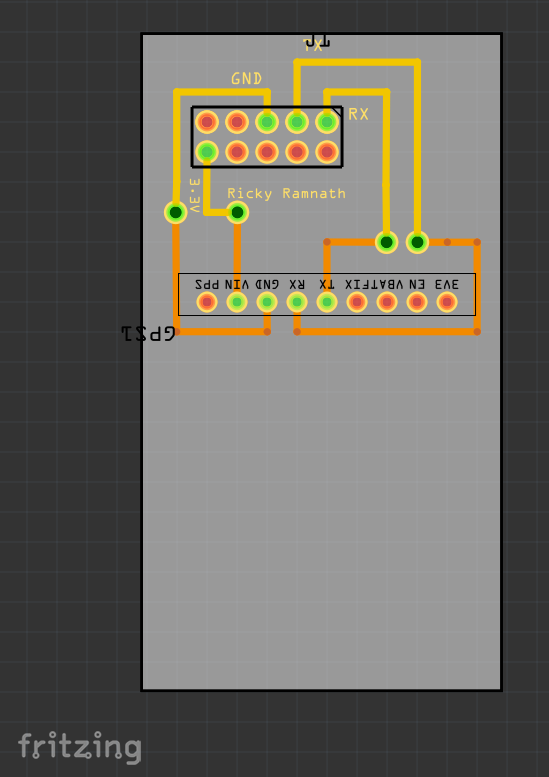
Students were required to attend a workshop in order to utilize the Idea Lab at the Humber College Library which has two Ultimaker 2+ 3D printers available for booking by qualifying Humber College Students. During the workshop we were shown how to generate .stl and .obj files for export and how to import these files in to the bundled software for the 3D printer so that the G-code for printing can be generated. For laser-cutting, .cdr files are generated from drafts made using Corel Draw and sent to the Prototype Lab, where the sides for a constructible case are cut out of a sheet of acrylic (in clear or black).

For the final prototype combining the three sensors/actuators and boards, a main acrylic case is to be produced for the GPS sensor, Raspberry Pi and RFID Sensor. Furthermore, there will be a space to mount the Keypad which will itself be in a 3D printed case.

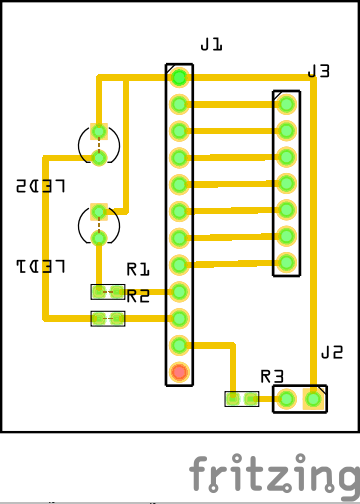
A second PCB had to be produced for the keypad as the copper rings surrounding the holes ripped off as the keypad moved due to being pressed. Foam padding was put on the underside of keypad to keep it stationary. There were several difficulties with the development of the case for the keypad. During the first attempt to print the case at the Idea Lab there was an issue with the filament that prevented the job from being properly finished and due to other students having booked the printers throughout the week and the print job had to be moved to the Prototype Lab. The second printed case was too small (the wall thickness was not properly accounted for) and due to time constraints a suitable case was purchased from Amazon.ca. The first PCB board for the GPS/Raspberry Pi was damaged as the copper lines were burned off after trying to remove excess solder with a de-soldering gun. Furthermore, sizing of initial acrylic case was too small and a new one with the proper dimensions had to be cut. The acrylic case and the PCB for the RFID reader was completed with no significant issues.

Images of each of the PCB designs and cases can be seen below:

GPS:



Keypad:



### 3.1.3 Tools and Facilities

For our parts crib management system, most of the tools required for testing the hardware, are either already given to us in our classroom or we already have it with us at all times in our toolboxes. For testing on our hardware’s PCB Board, we will be using multimeters in order to find any connection continuity errors. Some other components that we may need to borrow for our hardware design, such as the helping hand for holding the PCB board, will be acquired from the parts crib. For the mobile application component of the capstone project, our lab computers already come with the software tools that we need in order to finish and fine tune building our app, such as Android studio and SDK tools which can be used to program and emulate our mobile app and troubleshoot any problems that may occur, also, test how it would feel for a user using our app. The software tool for the hardware enclosure design such as CorelDraw will be used in order to achieve precise measurements that will hold all our hardware components. Also, for the PCB board testing, the software tool we will be using is Fritzing which allows us to create PCB boards and test our connections and export them as pdf files which will be sent to the Prototype Lab facility for cutting. In terms of facilities we will be using, most of our time spent will be in the labs, where we will continue to build upon our project. Other facilities we are considering incorporating into our project will be the Idea lab. The Idea lab at Humber College offers a 3D printing workshop session which we have/will attend in order to understand how to use the 3D printers so we can begin using the lab for 3D prints and testing designs to find out what works/fits well and for other minor improvements we could make to the enclosure. We will be using the 3D for creating the enclosure for our hardware components. Another facility that we will be using is the Prototype Lab frequently in order to create our PCB boards which will adhere to the requires of the lab, being made from copper and having vias the size of 3-5mm. It is likely that problems/errors will occur in the creation of our PCB boards, so visiting the Prototype Lab will be needed in order to make those. The Prototype Lab also offers a laser cutter which uses a multiple of materials such as acrylic, wood, etc. Prototype Lab is also where our collaborator Vlad is stationed, so it gives us the perk of being able to directly show off our progress and gives us the chance to add features/make chances to either the mobile application or the hardware design. A finish product of our project will consider using the laser cutter and all the other tools available in these facilities in order to achieve standards that will please our collaborator’s requests and create a product that can be used for demonstrating our abilities.

### 3.1.4 Shipping, duty, taxes

Shipping for our sensors were at a standard rate. The Adafruit GPS took approximately three days to arrive because it was ordered from the United States. In order to be able to have it cleared and reach its destination, I had to pay some duty fees which costed $7.64 Canadian Dollars. Taxes for this item costed zero dollars. Shipping fees for this sensor costed $22.05. The total price for everything for that sensor was $117.41. This was converted from $88.54 US Dollars. The currency exchange rate at that time was $1 US Dollar for $1.32 Canadian Dollars. The Twelve-Button Keypad took approximately fourteen days to arrive. There was no duty fee for this product. Taxes for this is as follows; goods and services tax were zero dollars, harmonized sales tax was $0.89, and provincial sales tax was zero dollars. Shipping fees for this effector costed $22.05. The total price for everything for this effector was $15.77 CAD. Since this was bought and shipped from Canada, there was no shipping charges and no currency conversions. The radio frequency identification scanner took approximately three days to arrive. This was ordered from Amazon. There was no shipping cost on this sensor because first time customers on amazon received free shipping on their first order. Taxes for this item costed $1.81 which included goods and services, harmonized sales tax, and provincial sales tax. The total price for everything for that sensor was $15.79 CAD. Since this was bought and shipped from within Canada, there was no shipping charges and no currency conversions.

### 3.1.5 Time expenditure

As there are parts of this project that we interact with third-parties within and outside the school, there are periods where we are required to wait for before we can proceed to another stage. In this section we will distinguish how we’ve scheduled our time during periods where we are actively working on completing a milestone (working time) and the intervening time between these active periods (lead time).

As the in-class meetings are in the middle of the week (Wednesday), PCB/3D printing requests are sent out on the Thursday of the previous week at the latest to ensure that components are ready for active development during class time. Items acquisition is restricted when possible to vendors who offer expedited shipping services such as Digi-Key or Amazon Prime.

During our working time, our team focuses on implementing the main breadboard/PCB, soldering pins and sockets and testing the continuity of the connections. The code tested individually on each sensor and the Raspberry Pi are tested with the sensors connected to the main PCB. All critical steps are actively documented and important milestones are photographed.

For lead time periods, since there are two Raspberry Pi 3 B+ boards available to our team, we individually work on connecting our sensors to the boards and testing the serial connections before the final bread-boarding and PCB build. This time is also used to further develop (and reevaluate) our UI/UX with respect to the android application and the website. Furthermore, depending on the workload for a particular week, we also refactor our most critical Android code from last semester and handle locale and language translations (all of the UI will be available in both French and English).

## 3.2 Development Platform

### 3.2.1 Mobile Application

Our Parts Crib Management System mobile application will allow students to conveniently borrow parts from the parts crib and to achieve a few goals in order to meet the satisfaction of our collaborators and create a well-designed app. When building the app, we first brain stormed interfaces/views that the student might want to see and/or will need when it comes to borrowing parts from the Parts Crib, while making it intuitive for the user. Following these conditions, we came up with four main views/tabs that the user should need, and we should keep it no more than four in order to avoid confusion. The four views we decided on for the mobile app was a home tab, a loans tab, a account tab, and a help tab, any other small details that we would need or plan on implementing in the future can be put into the settings menu. The home tab will allow users to view all the items in the Parts Crib inventory and choose what they need to rent out. We have also implemented a “Flooding Alert” in the home tab which alerts users that the Parts Crib is busy and might be out of a certain part, or in the event that is it closed. The next tab is the loans tab, here all of the user’s items that have been signed out will be displayed. We plan on implementing a feature on the Parts Crib employee’s end that will allow confirmation of a returned item to go through, then the item you have returned should disappear from the loans tab. The next tab is the accounts tab, it will contain basic information such as student number and name. We use this information in the firebase for when students are signing out parts so we can tie those parts to a person. It is also here that we will plan on implementing a 4-digit authentication code that will work with our keypad component of the hardware. For the future, we plan on allocating a space in the account tab for overdue items. The final tab is the help tab, here users will find helpful extra features such as the school map, which can help you navigate to the Parts Crib. The map is also planned in the future to be implemented with the GPS Breakout hardware component. The help tab may also be implemented to have a method of contacting Parts Crib employees in the event that users are facing issues with the sign out procedure. The login screen was kept simple with just an email and password field, and the register field was also kept simple except for a few extra pieces of required information such as student number. When an item has been requested, a few fields of information is stored in the Firebase. The important pieces are the quantity of items, the name of the item, along with a unique pin code. All of that is in a contained subcategory, the main block contains the student’s Humber ID, their name, and their email address. We need to keep all of this information tied to unique pins in the event of duplicate order of items. It is also used to keep track of the inventory, so the app prevents parts from being signed out that aren’t available anymore/ have all been signed out.

Status

/1 Hardware present?

/1 Memo by student A + How did you make your Mobile Application? (500 words)

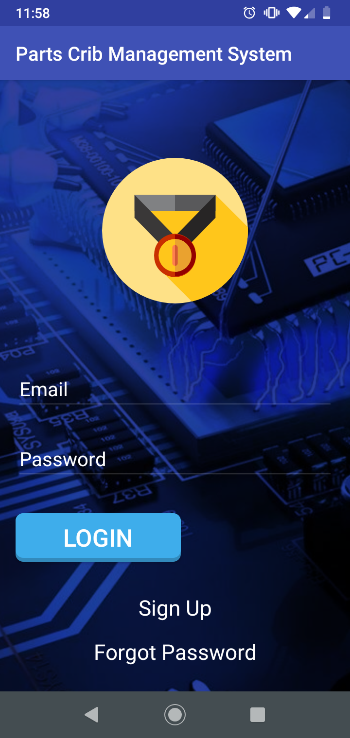
/1 (Login Activity) Figure 3 Login Activity

/1 Data visualization activity

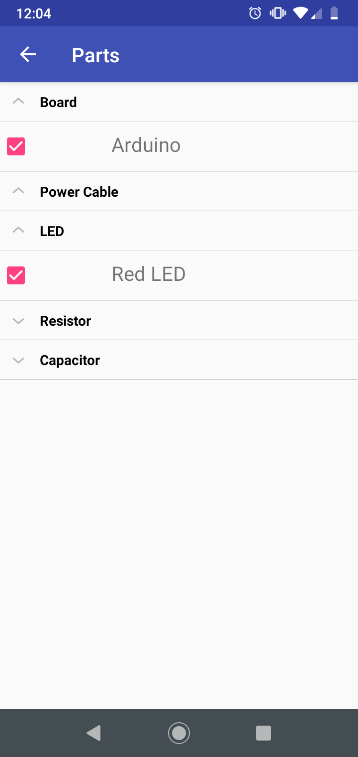
/1 Action control activity

### C:\Users\n01256088\Downloads\Screenshot_20200205-120344.png

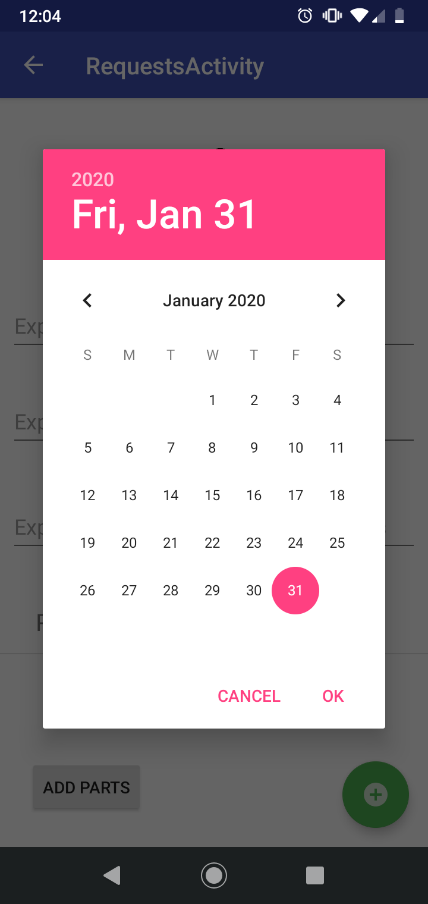
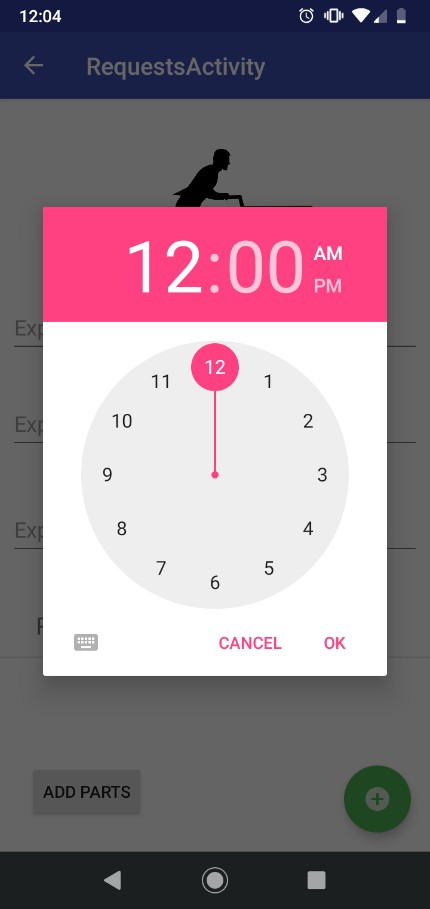
*Figure 1 HomeActivity Figure 2 Database*

*Figure 3 Login Activity*  *Figure 4 Loans Activity*

*Figure 5 Request Activity Figure 6 Parts List*

*Figure 7 & 8 Timed Request Activity*

### 3.2.2 Image/firmware

The firmware used to run this project on is the Raspberry Pi 3 Model B. This is the development platform used to implement this project. The Raspberry Pi will communicate with the sensors and effector using the proprietary general input/output pins. To connect to the Broadcom development platform, you must use either a remote desktop access, or a high definition multimedia interface. To setup the firmware, you will need to download the latest firmware of Raspbian onto your computer, then format an SD card to make it bootable. From there you can simply just drag the .iso file onto the SD card and connect it to your development platform. Follow the on-screen instructions to install the firmware. Once that is done you have successfully installed Raspbian to your raspberry pi. You will now be able to explore and create. To create an image of the Raspberry Pi that will be used in this project, I used a program called Win32 Disk Imager. I have created an image of it and will be able to share it in the future once everything is completed in the project. The code for the Global Position Sensor has been uploaded to the GitHub page of this project and build instructions as well. By following the steps from the build instructions, one can successfully be able to create a working GPS project on a Raspberry Pi. To connect to the Pi wirelessly, you will need to use a program called VNC Viewer. This will allow remote access of the development platform without the use of an ethernet cable or an HDMI cable. To set this up you need to configure the pi to enable VNC on it and download the latest version of VNC viewer from the internet. Internet connection from the pi and the computer must be the same or else you will not be able to access it. This will allow you to have a simple setup without the hassle of carrying an ethernet cable and an ethernet adapter. You will only need your power cable. An issue that was encountered while trying to wirelessly access our Raspberry Pi included; not being able to connect to the Humber College Wi-Fi. I have tried many ways to get it to connect but have had no luck trying to successfully connect it. We have tried many ways to troubleshoot the issue but there was no solution to connect to the Humber College Wi-Fi network. The only way we were able to get it to work was to either connect it to either a home internet connection, or use a hotspot from our mobile devices to get the Raspberry Pi to access the internet while on campus. We are actively trying different methods to get it to work on campus Wi-Fi. Once a solution is found, our GitHub repository and this document will be updated.

Status

/1 Hardware present?

/1 Memo by student B + How did you make your Image/firmware? (500 words)

/1 Code can be run via serial or remote desktop

/1 Wireless connectivity

/1 Sensor/effector code on repository

### 3.2.3 Breadboard/Independent PCBs

Status

/1 Hardware present?

/1 Memo by student C + How did you make your hardware? (500 words)

/1 Sensor/effector 1 functional

/1 Sensor/effector 2 functional

/1 Sensor/effector 3 functional

*Figure 10. Breadboard prototype.*

### 3.2.4 Printed Circuit Board

Demo

/1 Hardware present?

/1 PCB Complete and correct

/1 PCB Soldered wire visible but trim, no holes or vacancies

/1 PCB Tested with multimeter

/1 PCB Powered up

### 3.2.5 Enclosure

Demo

/1 Hardware present?

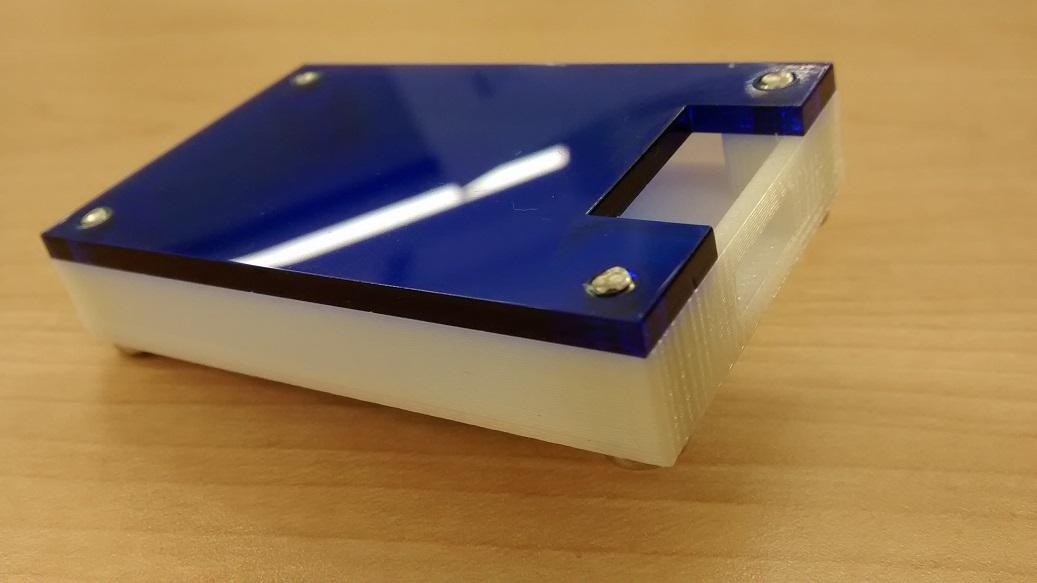
/1 Case encloses development platform and custom PCB.

/1 Appropriate parts securely attached.

/1 Appropriate parts accessible.

/1 Design file in repository, photo in report.

How did you build your Prototype: Case?

**

*Figure 13. Example enclosure.*

## 3.3 Integration

Demo

/1 Hardware present?

/1 Data sent by hardware

/1 Data retrieved by mobile application

/1 Action initiated by mobile application

/1 Action recieved by hardware

Report

/1 Enterprise wireless connectivity (250)

/1 Database configuration (250 words)

/1 Security considerations (500 words)

/1 Unit testing (900 words)

/1 Production testing (100 words)

### 3.3.1 Enterprise Wireless Connectivity

### 3.3.2 Database Configuration

The firmware consists of a single python file run on startup after logging in to the raspberry pi. The third party libraries used are for the sensors are pad4pi to interface with the keypad, adafruit\_gps for the gps system and mfrc522 for the MFRC522 RFID sensor. The library used to interact with the firebase database is pyrebase.

The requirements.txt file to install the libraries has been included.

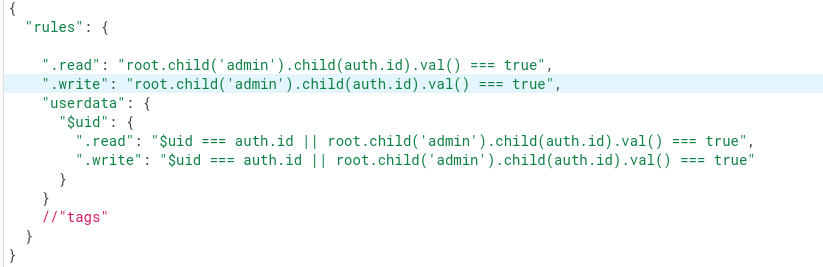




### 

### 3.3.3 Security

The firebase application uses database rules to limit access to the whole database or parts of it based on rules defined in the console. The rules below allow only users belonging to the admin in the group to read and write from the whole database. Further rules on the userdata portion of the database to be modified by authenticated users but only for their own user data.



### 3.3.4 Testing

Unit testing and Production testing.

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 4.0 Results and Discussions

Is your prototype perfect? What did you learn?

The primary functions of the physical prototype are working correctly. While an encasement wasn’t able to be produced, the PCB has no physical issues such as broken solder contacts or unintended connections. The sensors are working as expected, although the GPS sensor required the use of an external antenna to be able to obtain a signal. The most significant hardware issue with our prototype came during the development process when the Nucleo-32 development board that interfaced with the keypad failed to properly communicate with the Raspberry Pi using i2c and eventually failed all together. This required us to directly interface the Keypad with the Raspberry Pi.

Furthermore, as our team began to integrate the prototype into the rest of the system and worked through the sign-in/pick-up flow we discovered some design aspects that could be improved upon. Firstly, there is no way to indicate the success status of an RFID tap or keypad entry to students signing in on the prototype, as there are no programmable lights or speakers built into the Raspberry Pi or included on the PCB. The initial plan was to have the system be Raspberry Pi configured to run headless and the web connected computer at the Parts Crib would provide audible status feedback using the computers feedback, however, our dashboard needs to incorporate an alert system for incoming requests and it became clear that having an audible alert for both sign-ins and incoming requests would overwhelm the staff and make it difficult to distinguish what the alert was for. Additionally, there is no way for users to intuitively reset or correct an incorrect Keypad entry or reset the sign-in process altogether. We discussed assigning one of the unused keys on the Keypad (\* and #) for this purpose but decided it would slow the process down further as there is no way for students to know the current state of the login system (as stated above). Another design obstacle we came across during the development of the platform was the use of the Firebase real-time database to store our data. As the database stores data in the form of documents using json notation, it made it more difficult to store inventory, user and request information. While there are methods to flattening data in noSQL databases, a traditional relational database would have better served the needs of our platform.

# 5.0 Conclusions

In conclusion, while the social distancing requirements and closure of facilities due to the SARS-CoV2 outbreak have created challenges for our team, we’ve still managed to to accomplish much of what we set out to do. The issues surrounding working remotely with team members on different components of the project are relatable to those that arise in larger workplaces where multiple teams have to coordinate with one another to complete projects. We feel that the communication skills developed in producing deliverables over the latter period of this course lend themselves well to the workplace and can be valuable to future employers. Furthermore, the lessons learned during the integration phase of this project taught us the value of taking the fully integrated system (software and hardware) and how it’ll be used by others into account when designing the prototype. Our team would like to thank and acknowledge our instructors Austin Tian and Kristian Medri for providing us with the necessary support to complete this project over the course of our final school year.

Report

/1 Hardware present?

/1 Checklist truthful

/1 Valid Comments

/1 Results and Discussion (500 words)

/1 Conclusion

# 6.0 References

Tejesh, B. S. S., & Neeraja, S. (2018). Warehouse inventory management system using IoT and open source framework. Alexandria Engineering Journal, 57(4), 3817–3823. https://doi.org/10.1016/j.aej.2018.02.003

Ben-Daya, M., Hassini, E., & Bahroun, Z. (2017). Internet of things and supply chain management: a literature review. International Journal of Production Research, 57(15–16), 4719–4742. https://doi.org/10.1080/00207543.2017.1402140

Hussain, K., Jhanjhi, N., Mati-ur-Rahman, H., Hussain, J., & Islam, M. H. (2019). Using a systematic framework to critically analyze proposed smart card based two factor authentication schemes. Journal of King Saud University - Computer and Information Sciences. https://doi.org/10.1016/j.jksuci.2019.01.015

Zelbst, P., & Sower, V. (2012). RFID for the supply chain and operations professional (1st;1; ed.). US: Business Expert Press.

# 7.0 Appendix

## 7.1 Firmware code

### 7.1.1 rasp\_firmware.py

import pyrebase

#import pythonwifi.iwlibs import Wireless

from mfrc522 import SimpleMFRC522

#!/usr/bin/env python

#from gps import \*

import os

import time

import subprocess

import sys

import board

import busio

import time

import adafruit\_gps

import serial

import RPi.GPIO as GPIO

from pad4pi import rpi\_gpio

from mfrc522 import SimpleMFRC522

API\_KEY = os.getenv('API\_KEY')

# Get Environment variables

o

GPIO.setmode(GPIO.BCM)

DEBUG = 1

LOGGER = 1

GPIO.setup(2, GPIO.IN)

GPIO.setup(3, GPIO.IN)

GPIO.setup(4, GPIO.IN)

GPIO.setup(17, GPIO.IN)

GPIO.setup(27, GPIO.IN)

#Read Tag

reader = SimpleMFRC522()

config = {

"apiKey": API\_KEY,

"authDomain": "partscrib-2fe52.firebaseapp.com",

"databaseURL": "https://partscrib-2fe52.firebaseio.com",

"projectId": "partscrib-2fe52",

"storageBucket": "partscrib-2fe52.appspot.com",

"messagingSenderId": "154343080345",

"appId": "1:154343080345:web:3c3060fc9d028b51e7d308"

}

firebase = pyrebase.initialize\_app(config)

email = "partscribmanagement@gmail.com"

password = "dummy123"

auth = firebase.auth()

user\_admin = auth.sign\_in\_with\_email\_and\_password(email, password)

db = firebase.database()

current\_student\_sign\_in = ""

current\_student\_onboarding =""

def gps():

#GPS

# num = int(subprocess.check\_output(['wc', '-l', "gps\_data.txt"])[:-13])

count = 0

try:

# print lines

gps\_data = {"longitude": "43.724330436", "latitude": "-79.605497578"}

db.child("newsBulletin/location/coordinates").update(gps\_data)

except (Exception):

print("Done.\nExiting.")

finally:

GPIO.cleanup()

print("Finished GPS")

entered\_passcode = ""

def digit\_entered(key):

global entered\_passcode

db.child(f"userdata/{current\_student\_sign\_in}/").update({"currentPinEntry": entered\_passcode})

if (len(entered\_passcode) < 6):

entered\_passcode += str(key)

#send to firebase

print("Key Pressed: " + entered\_passcode)

else:

db.child("newsBulletin/signIn/").update({"currentUID": "", "studentSigningIn": "false"})

def key\_pressed(key):

try:

int\_key = int(key)

if int\_key >= 0 and int\_key <= 9:

digit\_entered(key)

except (Exception):

pass

def keypad():

KEYPAD = [

[1, 2, 3],

[4, 5, 6],

[7, 8, 9],

["\*", 0, "#"]

]

ROW\_PINS = [17, 27, 22, 24] # BCM numbering

COL\_PINS = [4, 18, 23] # BCM numbering

factory = rpi\_gpio.KeypadFactory()

# Try factory.create\_4\_by\_3\_keypad

# and factory.create\_4\_by\_4\_keypad for reasonable defaults

keypad = factory.create\_keypad(keypad=KEYPAD, row\_pins=ROW\_PINS, col\_pins=COL\_PINS)

print("Keypad Intialized... Enter keys")

keypad.registerKeyPressHandler(key\_pressed)

def onboardRFID(uid):

print("Tap tag to write UID to rfid chip...\n")

# First read rfid id, text

id, text = reader.read()

# Overtite text to be student number and then update tag table

reader.write(uid)

db.child(f"tags/{id}/").set(uid)

print(f'uid: {uid} written to tag\n')

def onboarding\_stream\_handler(message):

if message["data"]["studentOnBoarding"] == "true":

global current\_student\_onboarding

current\_student\_onboarding = message["data"]["currentUID"]

onboardRFID(current\_student\_onboarding)

# set onboarding to false

# set to false after onboarding complete

db.child("newsBulletin/onBoarding/").update({"currentUID": "", "studentOnBoarding": "false"})

def signIn\_stream\_handler(message):

if message["data"]["studentSigningIn"] == "true":

print("Sign-In initiated....")

global current\_student\_sign\_in

current\_student\_sign\_in = message["data"]["currentUID"]

keypad()

def location\_stream\_handler(message):

if message["data"]["updateLocation"] == "true":

gps()

db.child("newsBulletin/location/").update({"updateLocation": "false"})

def main():

while True:

# Listen for signIn or onboarding event

onboard\_stream = db.child("newsBulletin/onBoarding/").stream(onboarding\_stream\_handler)

signIn\_stream = db.child("newsBulletin/signIn/").stream(signIn\_stream\_handler)

location\_stream = db.child("newsBulletin/location/").stream(location\_stream\_handler)

# Wait for user tap and set signIn to true to begin process for keypad entry

try:

id, text = reader.read()

# Get the userID from tags table based on RFID id from database

user\_id = db.child(f'tags/{id}').get().val()

print(f"Tag belongs to user {text}.")

# update the current user id and singIn status at the location

db.child("newsBulletin/signIn/").update({"currentUID": user\_id['firebaseStudentID'], "studentSigningIn": "true" })

except (Exception):

print("Error: There was an error")

continue

finally:

onboard\_stream.close()

signIn\_stream.close()

location\_stream.close()

if \_\_name\_\_ == '\_\_main\_\_':

main()

### 7.1.2 requirements.txt

adafruit-circuitpython-gps

pad4pi

mfrc522

pyrebase

Demo

/1 Hardware present?

/3 Code runs concurrently for all sensors/effectors

/1 Project repository contains integrated code

Status

/1 Memo including updates

/1 Financial update

/1 Progress update

/1 Modified Code Files in Appendix

/1 Link to Complete Code in Repository

## 

## 

## 

## 

## 7.2 Application code

Code Location: <https://github.com/c3ko/Parts-Crib-Android>

Demo

/1 Hardware present?

/1 Memo by student A

/1 Login activity

/1 Data visualization activity

/1 Action control activity

Report

/1 Login activity

/1 Data visualization activity

/1 Action control activity

/1 Modified Code Files in Appendix

/1 Link to Complete Code in Repository

The Login Activity (along with a registration and forgotten password activity) were already implemented in the CENG 318. Those activities along with the ability to create requests was also demonstrated during the previous android application status update. In the intervening time we’ve further added an activity to input the six digit pin required to retrieve requested parts instead of using the keypad. The activity updates and sets the current pin being required by the parts crib through the Firebase database so that each key press on either the keypad or the phone will be updated throughout the system in real time. Furthermore, we’ve implemented an activity to display the current location of the Parts Crib in real time by reading the GPS coordinates from the GPS sensor from the Firebase database through the android application.

### 7.2.1 KeypinActivity.java

package com.partscrib.partscribmanagementsystem;

import androidx.annotation.NonNull;

import androidx.annotation.Nullable;

import androidx.appcompat.app.AppCompatActivity;

import com.davidmiguel.numberkeyboard.NumberKeyboard;

import com.davidmiguel.numberkeyboard.NumberKeyboardListener;

import com.google.android.gms.tasks.OnFailureListener;

import com.google.android.gms.tasks.OnSuccessListener;

import com.google.firebase.database.ChildEventListener;

import com.google.firebase.database.DataSnapshot;

import com.google.firebase.database.DatabaseError;

import com.google.firebase.database.DatabaseReference;

import com.google.firebase.database.FirebaseDatabase;

import com.google.firebase.database.ValueEventListener;

import com.partscrib.partscribmanagementsystem.model.ExpandableListPartData;

import com.partscrib.partscribmanagementsystem.model.PartExpandableListAdapter;

import com.partscrib.partscribmanagementsystem.model.PartModel;

import android.os.Bundle;

import android.service.autofill.UserData;

import android.util.Log;

import android.widget.TextView;

import org.w3c.dom.Text;

import java.util.ArrayList;

import static com.partscrib.partscribmanagementsystem.Login.USER\_NAME\_MESSAGE;

public class KeypinActivity extends AppCompatActivity {

NumberKeyboard numberKeyboard;

TextView textView;

String keypin = "";

private FirebaseDatabase db;

private DatabaseReference dbRef;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_keypin);

numberKeyboard = (NumberKeyboard) findViewById(R.id.numberKeyboard);

textView = (TextView) findViewById(R.id.keypin\_text);

numberKeyboard.showRightAuxButton();

db = FirebaseDatabase.getInstance();

String user = getIntent().getStringExtra(USER\_NAME\_MESSAGE);

String path = "userdata/" + user + "/currentPinEntry";

dbRef = db.getReference(path);

numberKeyboard.setListener(new NumberKeyboardListener() {

@Override

public void onNumberClicked(int number) {

if (keypin.length() < 6){

setCurrentPin(number);

textView.setText(keypin);

}

else {

}

}

@Override

public void onLeftAuxButtonClicked() {

}

@Override

public void onRightAuxButtonClicked() {

}

});

getCurrentPin();

}

public void setCurrentPin(int number){

keypin = keypin + number;

// Submit pin to firebase

dbRef.setValue(keypin);

}

public void getCurrentPin(){

dbRef.addValueEventListener(new ValueEventListener() {

@Override

public void onDataChange(@NonNull DataSnapshot dataSnapshot) {

keypin = dataSnapshot.getValue(String.class);

textView.setText(keypin);

}

@Override

public void onCancelled(@NonNull DatabaseError databaseError) {

}

});

}

}

### 7.2.2 MapActivity.java

package com.partscrib.partscribmanagementsystem;

import androidx.annotation.NonNull;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import com.google.android.gms.maps.CameraUpdateFactory;

import com.google.android.gms.maps.GoogleMap;

import com.google.android.gms.maps.MapView;

import com.google.android.gms.maps.MapsInitializer;

import com.google.android.gms.maps.OnMapReadyCallback;

import com.google.android.gms.maps.model.BitmapDescriptorFactory;

import com.google.android.gms.maps.model.LatLng;

import com.google.android.gms.maps.model.MarkerOptions;

import com.google.firebase.database.DataSnapshot;

import com.google.firebase.database.DatabaseError;

import com.google.firebase.database.DatabaseReference;

import com.google.firebase.database.FirebaseDatabase;

import com.google.firebase.database.ValueEventListener;

import static com.partscrib.partscribmanagementsystem.Login.USER\_NAME\_MESSAGE;

public class MapActivity extends AppCompatActivity {

class Location {

private String latitude;

private String longitude;

public Location(String latitude, String longitude) {

this.latitude = latitude;

this.longitude = longitude;

}

public String getLatitude() {

return latitude;

}

public void setLatitude(String latitude) {

this.latitude = latitude;

}

public String getLongitude() {

return longitude;

}

public void setLongitude(String longitude) {

this.longitude = longitude;

}

}

Location location;

MapView mMapView;

GoogleMap googleMap;

FirebaseDatabase db;

DatabaseReference dbRef;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_map);

db = FirebaseDatabase.getInstance();

String path = "newsBulletin/location";

dbRef = db.getReference(path);

mMapView = (MapView) findViewById(R.id.mapView2);

mMapView.onCreate(savedInstanceState);

mMapView.onResume(); // needed to get the map to display immediately

dbRef.addValueEventListener(new ValueEventListener() {

@Override

public void onDataChange(@NonNull DataSnapshot dataSnapshot) {

location = dataSnapshot.getValue(Location.class);

}

@Override

public void onCancelled(@NonNull DatabaseError databaseError) {

}

});

try {

MapsInitializer.initialize(getApplicationContext());

} catch (Exception e) {

e.printStackTrace();

}

mMapView.getMapAsync(new OnMapReadyCallback() {

@Override

public void onMapReady(GoogleMap mMap) {

googleMap = mMap;

// For dropping a marker at a point on the Map

//LatLng Humber = new LatLng(43.724330436,-79.605497578);

LatLng Humber = new LatLng(Float.parseFloat(location.longitude), Float.parseFloat(location.latitude));

googleMap.addMarker(new MarkerOptions()

.position(Humber)

.icon(BitmapDescriptorFactory.fromResource(R.drawable.humber))

.title("Welcome to Humber College"));

googleMap.moveCamera(CameraUpdateFactory.newLatLngZoom(Humber, (float) 6.0));

googleMap.setMinZoomPreference(2.0f);

googleMap.setMaxZoomPreference(30.0f);

googleMap.setTrafficEnabled(true);

}

});

}

@Override

public void onResume() {

super.onResume();

mMapView.onResume();

}

@Override

public void onPause() {

super.onPause();

mMapView.onPause();

}

@Override

public void onDestroy() {

super.onDestroy();

mMapView.onDestroy();

}

@Override

public void onLowMemory() {

super.onLowMemory();

mMapView.onLowMemory();

}

private double distance(double lat1, double lon1, double lat2, double lon2) {

double theta = lon1 - lon2;

double dist = Math.sin(deg2rad(lat1))

\* Math.sin(deg2rad(lat2))

+ Math.cos(deg2rad(lat1))

\* Math.cos(deg2rad(lat2))

\* Math.cos(deg2rad(theta));

dist = Math.acos(dist);

dist = rad2deg(dist);

dist = dist \* 60 \* 1.1515;

return (dist);

}

private double deg2rad(double deg) {

return (deg \* Math.PI / 180.0);

}

private double rad2deg(double rad) {

return (rad \* 180.0 / Math.PI);

}

}

## 7.3 Dashboard Application Code

Code can be found at:

<https://github.com/rickyramnath97/gps/tree/master/CENG355/Code/parts-crib-front>