**WORK SUMMARY**

1. Understanding product management
2. Resetting Ubuntu root password
3. Arduino tutorials and introduction(hardware part as well software part)
4. Working with Arduinos (Arduino Uno)

* Programming with Arduinos using C programming language
* Basic connections of Arduino and other related components e.g. LEDs, jumpers, Resistors
* Controlling LED blinks using Arduino Uno

1. Introduced to ESP32

* Installing and understanding libraries that Arduino IDE uses in order to recognize ESP32 board
* Controlling LED blinks with ESP32

1. Introduction to Micro python(UpyCraft) python IDE
2. Introduced to RFID-RC522 module

* Connecting RFID-RC module to Arduino Uno
* Understating libraries that C programming language uses on Arduino IDE to establish a serial connection between the host and the RFID-RC522 module.
* Writing a code to scan RFID card information and display it on serial monitor with Arduino IDE

1. Working with ESP32 and RFID-RC522

* Connecting RFID-RC522 to ESP32
* Writing a code to scan the RFID card through the RFID-RC522 module and display the information on a serial monitor.
* Creating a database and able to add the information form the card (RFID card/tag)
* RFID identification with LEDs

1. Log into a computer with an RFID card

* This is possible using Arduino Leonardo or micro Arduino
* Working with Atmel flip to flush the firmware of Arduino Uno so that it should act as Arduino Leonardo. This kind of an Arduino has the ability to accommodate a library in C programming language known as Keyboard.h, This enables the Arduino to be recognized as an input device. This technique emphases on having the password on the card such that instead of printing it using the computer’s keyboard, you could just scan the card.
* The code was compiling properly as well as uploading. Although the keyboard library was recognized after flipping the Arduino Uno into such functionality like Arduino Leonardo and uploading the code. But the component after being plugged in on log in and scanned, the device did not perform its intended purpose.
* I migrated to ESP32 to try the same technique but this did not work as well.
* Given a door locking system code done on ESP32 board to try to understanding the basic authentication of RFID cards and deny access to an authorized card on the door locking system.

1. Introduction to Raspberry pi micro controller running on raspian OS

* Understanding the raspberry pi
* Research on how to connect the RFID-RC522 module to the raspberry pi
* Installing python IDE and python modules that will establish serial connecting between the RFD-RC522 module and the raspberry pi

**MODULES INSTALLED**

RPi.GPIO as GPIO

mfrc522

As well as enabling the SPI interface

* Developing a python code to scan for an RFID card and displays the information
* The coding apart was basically done on Thonny IDE so all the module were supposed to be installed in connection to the Thonny IDE and not any other IDE e.g. python3.7 or python 2.7 which I had previously installed.
* With the research I founded a code which could also be used to write data on the card by just scanning the card on the RFID-RC522 module, the information will be automatically be installed on the card.
* Developing an algorithm which would utilize the files of raspian OS to be able to log into the raspberry pi.
* This was done after a research of some days.
* Finally the algorithm was developed and I had to find a means on how I can implement this algorithm.
* I also worked with mysql and sqlite in a way that I was writing a python code which would insert the scanned information into a database which I had created.
* The tables within this database recorded the ID number, User\_Name, and the time (TIMESTAMP) when scanned.
* I decided to implement the algorithm through python programming language because the syntax is quite familiar to me.
* Did a pretest to see if the scanning of the RFID card given the unique id number would actually initiate any action on the computer. So it happened that this test worked as expected hence the proof of concept was there.
* Within the research I encounter the need to understand the /etc/profile file which I would utilize in my code execution.
* So I had to learn and find out more of this so called /etc/profile file
* Then I implemented the technique which successfully worked after a number of trials and configuration.
* Techniques to change shell mode by editing the raspberry pi SD on a windows machine by editing the cmdline.txt file, and add init=/bin/sh this normally helps to perform some of the commands that you are not allowed to when logged as a pi user or root sometimes. After the changes have been done on the raspberry pi, you need to go back to the cmdline.txt file and return its initial state by erasing the init=/bin/sh statement

1. Migrating to Ubuntu MATE

* Repeated most of the operations done on the Raspian OS because there structure is quite similar though not exactly the same.
* The mechanism worked as well on UbuntuMATE though with a slight change
* I proposed to develop and interface so that I should be having a fully working system
* Started working with different Softwares that could help me in coming up the interface. It would include integrating the background python code and the interface buttons to call for such actions.
* Crontab to change wallpaper when booting
* The wares that I slightly familiarized with : Visual Studio, PyQt and XOJO
* On the final presentations, the use of crotab was also proposed instead of /et/profile