A MINOR PROJECT REPORT

ON

ANDROID BASED SMART LOCK

SUBMITTED IN PARTIAL FULFILLMENT FOR THE AWARD OF DEGREE OF

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE ENGINEERING

Submitted By:

RAKSHIT GANGWAR (17103246 B6)

PUSHPAK AGARWAL (17103232 B6)

SHIVA KANSAL (17103221 B5)

Under the Guidance Of

DR. TRIBHUWAN KUMAR TEWARI

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA (U.P.)

CERTIFICATE

This is to certify that the minor project report entitled, "**ANDROID BASED SMART LOCK** " submitted by **RAKSHIT GANGWAR, PUSHPAK AGARWAL AND SHIVA KANSAL** in partial fulfillment of the requirements for the award of Bachelor of Technology Degree in **Computer Science Engineering** of the Jaypee Institute of Information Technology, Noida is an authentic work carried out by them under my supervision and guidance. The matter embodied in this report is original and has not been submitted for the award of any other degree.

Signature of Supervisor:
Name of the Supervisor: Dr. TRIBHUWAN KUMAR TEWARI
CSE Department,
JIIT, Sec-62,
Noida-201301

Dated:

DECLARATION

We hereby declare that this written submission represents our own ideas in our own words and where other ideas and words have been included have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified and idea/data/fact/source in our submission

Place:

Date:

Name: **RAKSHIT GANGWAR**

Enrollment: 17103246

Name: **PUSHPAK AGARWAL**

Enrollment: 17103232

Name: **SHIVA KANSAL**

Enrollment: 17103221

ABSTRACT

A Smart lock is a combination of an electronic and mechanical locking device. This device is purely wireless and requires no traditional key. Where everything around you are smart be it smartphones, smart cars etc. for your smart home you need a smart lock. These smart locks allow home owner to unlock their house doors wirelessly even if they are at home or not. These locks are an extension of home automation into home security, these smart locks are to pe considered as a part of internet of things.

In today's smart world having a smart phone is a common device to have. Everyone carries at least one smart device with them. Locks and keys are the basic requirement for a door but managing them is a cumbersome job. So the solution is the smart lock which uses a smart phone's Bluetooth or a SMS to unlock the door.

ACKNOWLEDGEMENTS

Success is an effort bounded activity that involves cooperation of all. We hereby express our profound sense of gratitude and are thankful to all those who have helped and encouraged us towards successful completion of the project. It has been a great learning experience. We would like to thank our mentor **Dr. Tribhuwan Kumar Tewari** for his valuable support and guidance, constant encouragement and the opportunity provided to us to complete the project under her supervision. We would also like to express our gratitude to great almighty, our parents and our friends for their concerned support and encouraging words without which this project would not have been the way it is.

Table of Contents

CHAPTER 1

INTRODUCTION

The android based smart lock is made to prevent unauthorized access and trespassing to your property. Places having high valuable things are at more risk for intrusion. The aim behind such activity is stealing the important wealthy things. Using physical keys to lock the door is the common way we do it but it has ita own flaws. The lock to your door can have copy of the keys and if you lost one of those keys, you always need to change the lock. If you rented a flat mostly of them have same locks installed and the previous tenants may have the keys of your flat which is sometimes, we have to worry about it. Furthermore, you have many keys which you need to carry around which increases the risk of losing one.

The main goal of the project to give a secure access control which can replace the traditional physical keys used for accessing the door. Even if you are outside your house and someone comes to your home but you are stuck in traffic and will reach after some time, you can allow to unlock the door using GSM module with just a click away using your smart phone. It also allows elderly people and disabled people to unlock the door through remote access.[1]

CHAPTER 2

METHODOLOGY

We have assumed that the users will be using Android Smart Phones having Bluetooth features and will run the application on it. It will connect with the Bluetooth HC-05 which is been installed in hardware part. The mobile application we developed is for mobile android phones and will not work on another platform. However, we can customize it for another platform also. It is also assumed that user will operate it within the Bluetooth device which is of 10m to 100m and for more range we can use WIFI module.

Initially you have to connect your mobile phone thorough Bluetooth to the Bluetooth module in the lock. Turn on the Bluetooth from the notification panel and connect to HC-05 by entering the password you have set. Open the app and click on the button "CONNECT TO BLUETOOTH MODULE". This button calls function BTinit() to initialize the Bluetooth module and on successfully initialization it calls the function BTconnect() to connect which creates socket to handle the outgoing connection and gets connected to Bluetooth module. Then it begin listening for data from the Arduino by function beginListenData() to get the state of the door and sends the number 3 to the Arduino asking it to send the current state of the door lock so the lock state icon can be updated accordingly. If Arduino sends 3 (In Arduino code it goes to case 3) then the door was in locked state previously and send 4 then it was in unlocked state.

Now the connection is all set and if a person comes at your doorsteps and rings the bell. You can lock/unlock the door using Bluetooth. On clicking the "LOCK/UNLOCK VIA BT" button it will check the Bluetooth connection first. If not connected a toast message will appear to establish connection first and if connected then it will send 1 to Arduino (It goes to case 1). In case 1 it reads whether the door was in locked or unlocked by checking EPROM. If EPROM.read(0)==1 then it means the door was locked and if EPROM.read(0)==2 then it means it was unlocked. It rotates the servo motor to opposite of the state given by EPROM and writes new value to EPROM. It also sends 3 for locked state or 4 for unlocked state accordingly to the mobile application.

If connection cannot be made through Bluetooth then we can click the “LOCK/UNLOCK VIA SMS” button. The button will send the message containing number 1 to the GSM module which has a sim with has the same 10-digit number we provided in the java code. On successfully sending it will toast a message “SMS sent” and if there is a problem it will toast a message “SMS failed, please try again”. This one will go to case 1 and it reads whether the door was in locked or unlocked by checking EPROM. If EPROM.read(0)==1 then it means the door was locked and if EPROM.read(0)==2 then it means it was unlocked. It rotates the servo motor to opposite of the state given by EPROM and writes new value to EPROM.

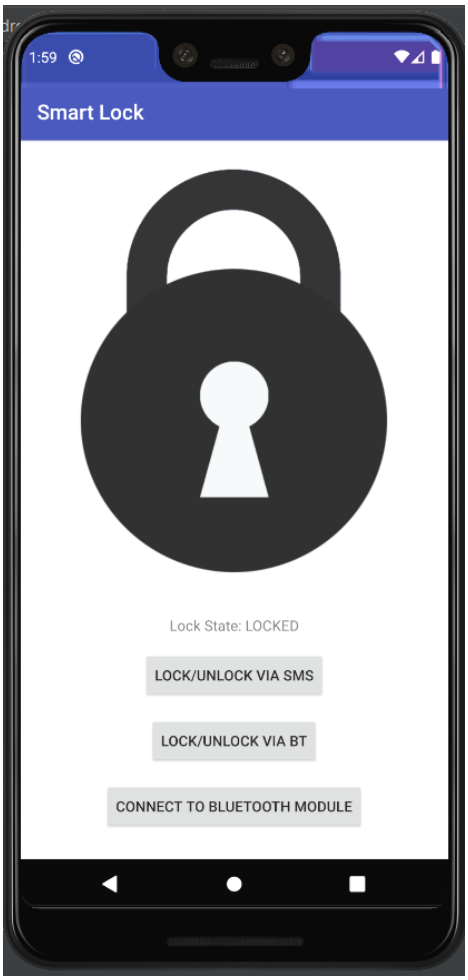
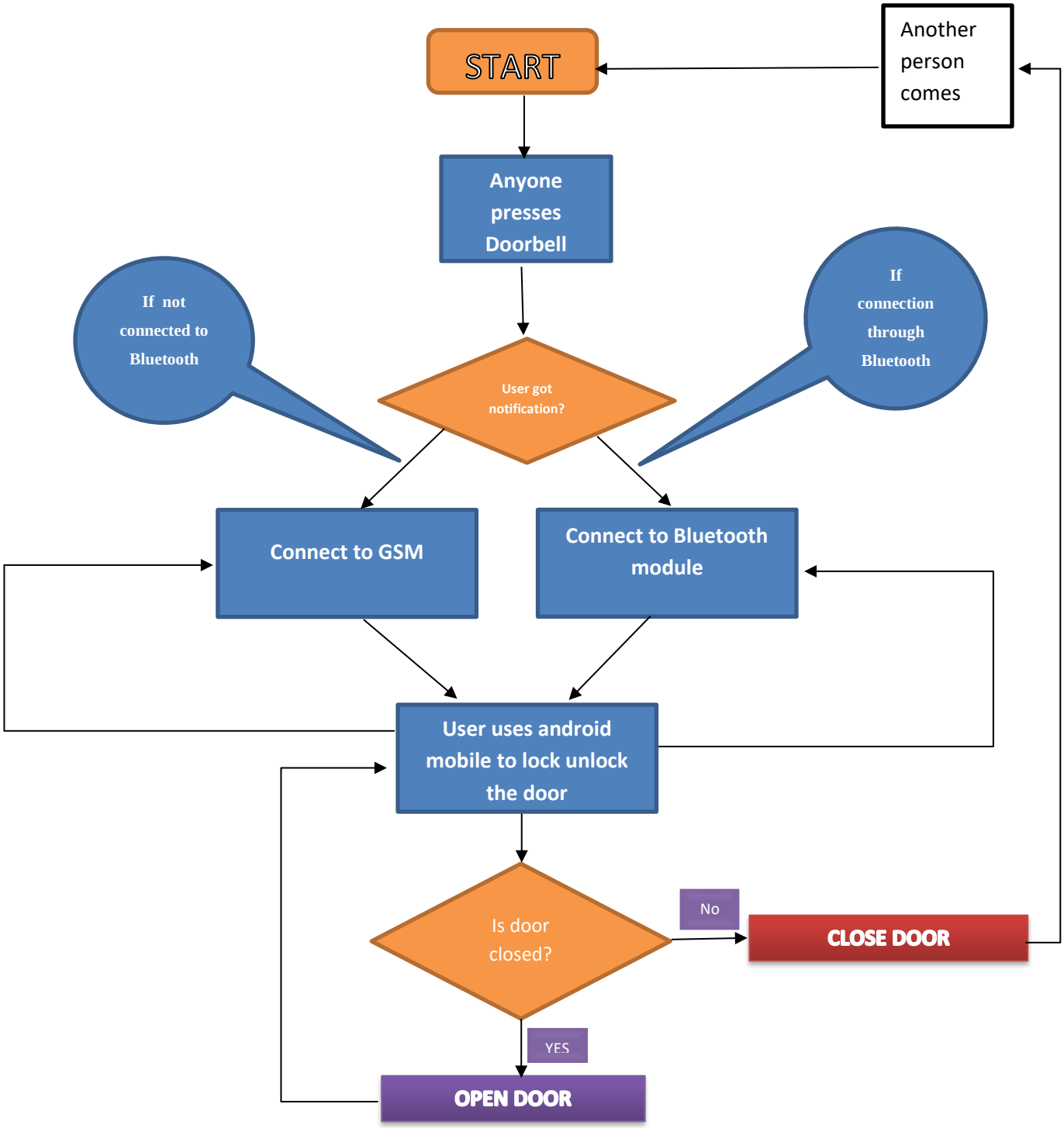


Fig 2.1: Application running on Android emulator.

To achieve our objective following methodology will be used:



CHAPTER 3

Review of Literature

Many places like commercial buildings and organization use automated advanced doors locking system has been developed. Radiofrequency identification (RFID) is the technology on which some automated doors locking system are based. These systems are very expensive, RFID card reader detects and checks the user accessibility. When the card is brought near the reader, it identifies the radio frequency of the card and thus verifies the key. Another system to open doors is by hand recognition. K. Lian, S. Hsiao and W. Sung, "Home safety handwriting pattern recognition system", the implementation was done using a hand writing pattern to open the doors.[2] Another advancement was done by M. Sahani, C. Nanda, A. K. Sahu and B. Pattnaik, "Web-based online embedded door access control and home security system based on face recognition," opening and closing of doors using facial structure of a person.[3] Over the years various control systems are designed to prevent unauthorized access. To provide security of our lives and property these locks are installed our home, school, office, and building. Lia Kamelia, Alfin Noorhassan S.R, MadaSanjaya and W.S., Edi Mulyana has implemented a "Door – Automation System Using Bluetooth", the implementation was on Android platform. By this the implementation cost is less and affordable by a common user. It is easy to install the system because of the use of wireless Bluetooth [4]. Arpita Mishra, Siddharth Sharma, Sachin Dubey, S. K. Dubey has implemented a "Password Based Security Lock Proposed Methodology system". It works by entering the password from keypad into the system. If entered password is correct then door is open by motor which is used to rotate the handle of the door lock. This system includes extra features like adding new users and changing old password etc. [5].In our findings we found that there are many smart lock systems but are very expensive. We identified these requirements and thought of making a system which is reliable, cost effective and the most important is to be user friendly. These features are the need of time and such functionalities will make the system more useful.

CHAPTER 4

PROBLEM STATEMENT

The main problem that this project attempts to solve is robbery. There must be an efficient and reliable system to help the user to lock the door. Even if a person is anywhere in the world he can look/unlock the door.

Most lockers are design with key lock commonly suffer from some possible flaws such as lost key, unauthorized key part, forgot to bring keys and the most serious problem is they are forgot to lock the door because their carelessness. The Android based smart lock system proposes proper system that is user friendly and easy to access. Gate can be unlocked by phone using Bluetooth if you are at home and if at any other location accordingly you can lock/unlock the door or report to police if necessary. This system also can adapt to home and office door. Besides, it has low cost and offer more security to user. Disabled people and elderly people who are unable to move can remote access the door without even moving. Pregnant women who are advised for bed rest can easily unlock/lock the door according to their convenience.

A person can easily unlock the door just by using the mobile irrespective of whatever place he is sitting in house or even outside the house. This will save a lot of time and disturbance. With this system you have no chance of losing your key to your lock.

CHAPTER 5

DEPENDENCIES

5.1. Libraries Imported

i. AppCompactActivity
ii. os.Bundle
iii. Bluetooth Adapter
iv. Bluetooth Device
v. Bluetooth Socket
vi. Intent
vii. MotionEvent
viii. View
ix. Context
x. Manifest
xi. smsmanager
xii. Button
xiii. ImageView
xiv. TextView
xv. Toast
xvi. IOException
xvii. InputStream
xviii. OutputStream
xix. Set
xx. UUID
xxi. Handler

5.2. Android Studio

Gradle – com.android.tools.build:gradle:2.2.0

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 24
    buildToolsVersion "24.0.2"
    defaultConfig {
        applicationId "com.example.philipgo.servodoorlock"
        minSdkVersion 15
        targetSdkVersion 24
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {
        exclude group: 'com.android.support', module: 'support-annotations'
    })
    compile 'com.android.support:appcompat-v7:24.2.1'
    testCompile 'junit:junit:4.12'
}
```

CHAPTER 6

HARDWARE USED

6.1. INTRODUCTION

Hardware refers to the physical components of a device. This is also sometime called the machine or the equipment of the computer. Examples of basic hardware in a computer are the keyboards, the monitor, the mouse and the central processing unit also known as CPU. However, most of a computer's hardware cannot be seen; in other words, it is not an external element of the computer, but rather an internal one, surrounded by the computer's casing.

A computer hardware also comprises of many different parts, but perhaps the most important of these is the motherboard. The motherboard is made up of many parts that control power and control the computer. In contrast to software's, hardware is a physical evidence. Hardware and software are interconnected, without software the hardware of a computer would have no function, so basically both work parallel. However, without the creation of hardware's to perform task directed by software via the central processing unit, software will be useless.

6.2. COMPONENTS USED

Arduino uno

Breadboard

Tower Pro Sg90 Servo Motor

Bluetooth Module Hc-05

Smart Phone

GSM Sim 800

Battery

Jumper Wire

Breadboard Wire

Cable

MAIN CIRCUIT WE ASSEMBLED:

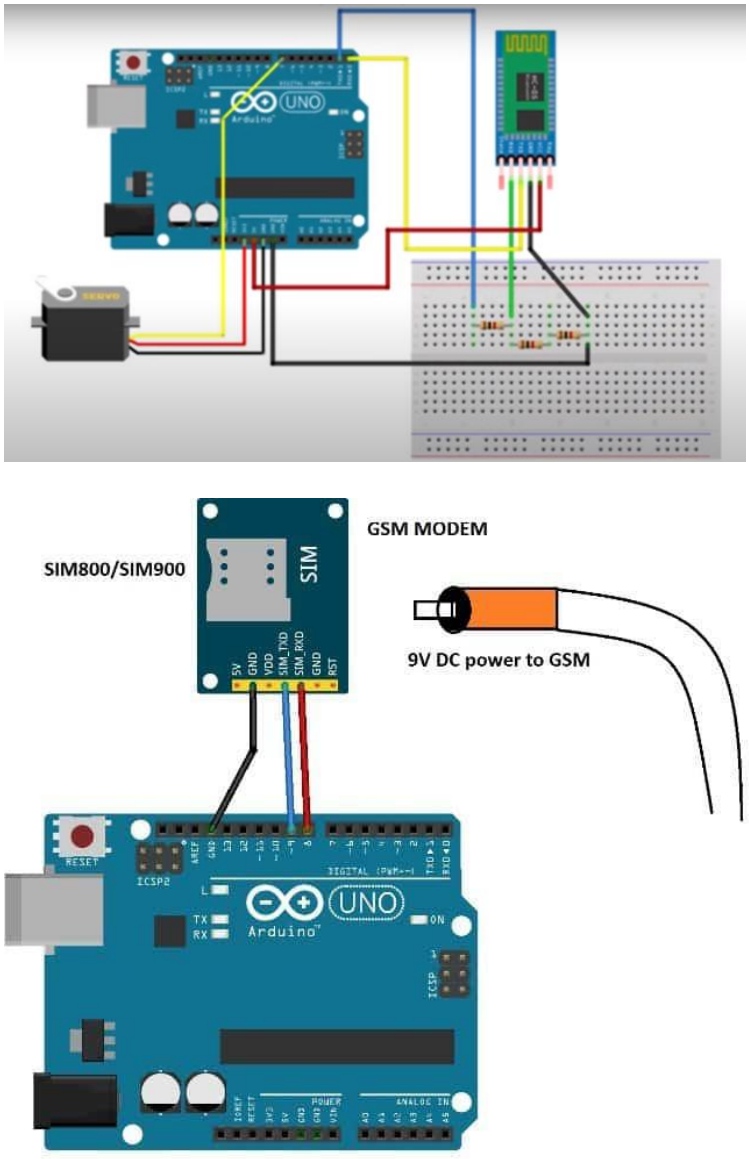


Fig 6.1: Circuit Diagram (Source: http://www.electronic-circuits-diagrams.com/)

6.3. ARDUINO UNO

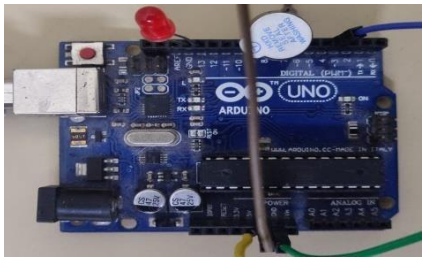


FIG 6.2: Arduino Uno

Arduino is an open-source electronic platform based on easy-to-use hardware and software. Arduino uno are able to read input - light on a sensor, a finger on button, or a Twitter message - and turn it into an output - activating a motor, turning on an LEDs or giving an output, publishing sometime online. All of it depends on you, can tell your device what to do by sending a set of instructions to the microcontrollers on the board. To do so we use the Arduino Programming Language (based on Wiring) which is the coding part of the Arduino, and the Arduino Software (IDE), based on Processing.

Over the years Arduino is the brain of thousands of projects like CPU for the computers, from everyday object to complex scientific instrument. A worldwide community of makers - students, hobbyist, artist, and professional also many companies - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible language that can be of great help to improve the functionality of Arduino.

Arduino was first made at the Ivrea Interaction Design Institute as an easy compilation for fast prototyping, which aim at students who have no a background in electronics and programming. As soon as it reached to more people wider, the Arduino board started changing to adapt to new needs, technology and challenges, differentiating its an offer from simple 8-bit boards to products for IoT applications, 3D printing, and embedded environments. All Arduino boards are completely open-source connection, empowering users to build them independently and eventually accept them to their particular needs.

6.4. BREADBOARD

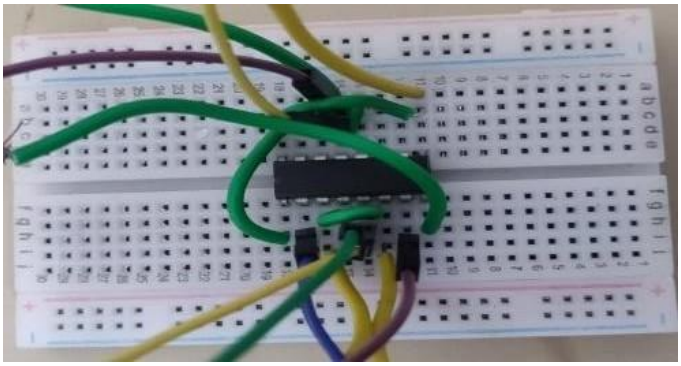


FIG 6.3: Breadboard

A breadboard is a construction based for case of electronics. It connects all the wire and circuits basically the hardware part of the program, Originally the word referred to a literal bread board, a polished piece of woodblock used for slicing bread in earlier century. In the 1970s the solder less breadboard become available and nowadays the term "breadboard" is commonly using to refer to these.

Because the solderless breadboard will not require soldering, it is reprocessed. This makes it easy to use again and again for creating temporary prototypes and experimenting with circuit design. For this reason, solderless breadboard are most common with students and in technological education. Older breadboard types did not have this property. A stripboard and similar prototyping printed circuit board, which are used to build semi-permanent soldered prototypes or one-offs, does not easily be reused. A variety of electronic systems may be prototyped by using breadboards, from small analog and digital circuits to complete central processing unit (CPUs).

6.5. TOWER PRO SG90 SERVO MOTOR

FIG 6.4: Tower PRO SG90 Servo Motor

It is made up of Carbon Fibre Gears which makes the servo motor much lighter than the same metal gear motor. For small load applications using the metal gear, servo motor implies on unnecessary weight, so we suggest using these lightweight plastic gear servo motors, because it is easy to get and use for multipurpose object. The Tower Pro SG90 9g Mini Servo is a 180° rotation servo. It is a Digital Servo Motor that receives and processes Pulse Width Modulation signal faster and good. It equips complicated internal circuitry that provides good torque, holding forces, and faster updates in response to external forces.

Wire Description

RED – Positive **Brown –** Negative **Orange –** Signal

Features:

1. High resolution
2. Accurate positioning
3. Fast control response
4. Constant torque throughout the servo travel range
5. Excellent holding power

6.6. BLUETOOTH MODULE HC-05

HC-05 is a Bluetooth module which is designed for wireless communication. This module will be used as a master or slave configuration. HC-05 has red LED which indicates connection account, whether the Bluetooth was connected or not. Before connecting to HC-05 module this red LED blinks continuously in a systematic manner. When it gets connected to any other Bluetooth device which is ready to pair, its blinking slows down to 2 seconds. This system works on 3.3 V. We can connect 5V supply voltage so we can get enough supply for the start of Bluetooth as well since the module has on board 5 to 3.3 V regulator. As HC-05 Bluetooth module has 3.3 V level.

It can detect 3.3 V level, so, no need to shift transmit level of HC-05 module. But we need to shift the supply voltage level from microcontroller to RX of HC-05 module.

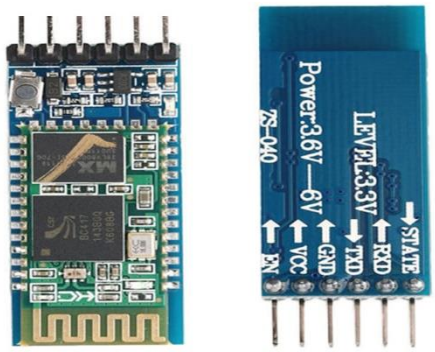


FIG 6.5: Bluetooth Module HC-05

PIN DESCRIPTION

Bluetooth serial modules allow all serial enabled devices to communicate with each other using Bluetooth.

It has 6 pins,

Key/EN: It is used to bring Bluetooth module in AT commands mode. If Key/EN pin is set to high or 1, then this module will work as command mode. Otherwise by default it

will work in data mode. The default baud rate of HC-05 at command mode is 38400bps and 9600bps in data mode.

HC-05 module has two modes,

1. **Data mode:** Exchange of data between devices.
2. **Command mode:** It uses AT commands which are used to change setting of HC-05.

To send these commands to module serial (USART) port is used.

1. **VCC:** Connect 5 V or 3.3 V to this Pin.

2. **GND:** Ground Pin of module.

3. **TXD:** Transmit Serial data (wirelessly received data by Bluetooth module transmitted out serially on TXD pin)

4. **RXD:** Receive data serially (received data will be transmitted wirelessly by Bluetooth module).

5. **State:** It tells whether module is connected or not.

BLUETOOTH COMMUNICATION BETWEEN DEVICES

E.g. Send data from Smartphone terminal to HC-05 Bluetooth module and see this data on PC serial terminal and vice versa.

To communicate smartphone with HC-05 Bluetooth module, smartphone requires Bluetooth terminal application for transmitting and receiving data. You can find Bluetooth terminal applications for android and windows in respective app. Store.

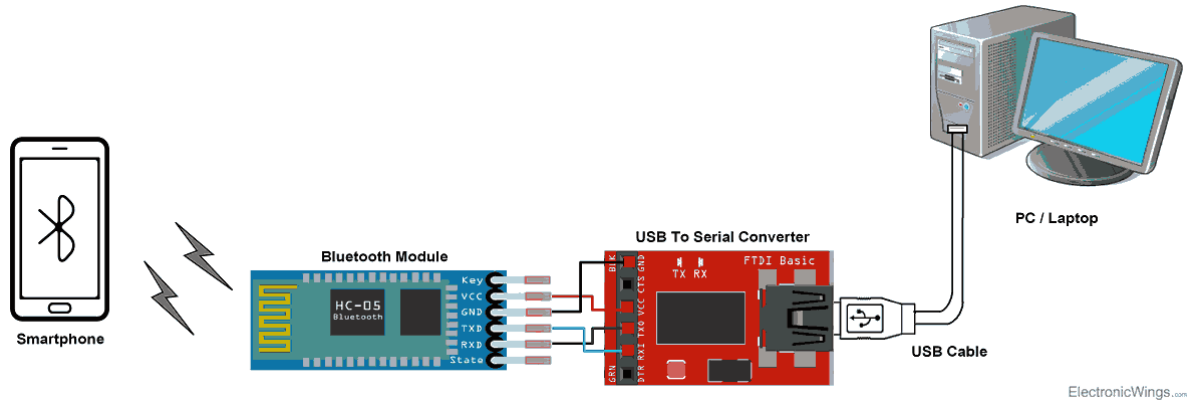


Fig 6.6 Bluetooth Communication (Source: www.ElectronicWings.com)

BLUETOOTH MODULE SERIAL INTERFACE

So, if we want to communicate through smartphone with HC-05 Bluetooth module, connect this HC-05 module to the PC using serial to USB converter.

Before establishing communication between two Bluetooth devices, first we need to pair HC-05 module to smartphone for communication.

6.7. SMART PHONE

Fig 6.7: Smart Phone

Smartphones is a class or generation of mobile phones and of multi-purpose mobile computing devices. These are distinguished from feature phones by their improved hardware capabilities and extensive mobile operating systems, which is facilitates wider software, and multimedia functionality (including music, video, cameras, and gaming), alongside core phone functions such as voiceover internet calls and text messaging. Smartphones typically numbers component of metal–oxide–semiconductor (MOS) integrated circuit (IC) chips, include various sensors that is longer), operating systems and valid for wireless communications protocols (such as Bluetooth, Wi-Fi, or satellite navigation).

6.8. GSM SIM 800

Fig 6.8: Smart Phone

SIM800 is a quad-band means 4 banded GSM/GPRS module that works on different frequencies like 850MHz GSM, 900MHz EGSM, 1800MHz, and 1900MHz PCS. It can feature GPRS multi-slot time 12/class 10 (optional), and supports CS-1, CS-2, CS-3, and CS-4 GPRS coding schemes. It has one UART port. It also has one USB port that can be used for updating firmware for protection and for debugging. Audio channels are also there, which include a microphone input and a receiver output signal. SIM800 has one SIM card interface. It integrates TCP/IP protocol. SIM 800 will be controlled/configured using simple AT commands. A host microcontroller will send some AT commands over the UART interface and control the SIM800.SIM800 works good on a supply in the ranges of 3.4 to 4.4V. It can be used for sending/receiving messages, making calls, sending/receiving data over the internet, etc. This makes it useful for applications such as home automation, agriculture automation, etc.

6.9. BATTERY

Fig 6.9: 9 Watt Battery

You can get a galvanic cell(Common battery) by combining two different electrodes together. However, you cannot use all the galvanic cells as practical cells or batteries. Usually, we use the term battery for a combination of a few cells that are similar in nature. A practical battery must have the following mentioned characteristics: It must be light in weight and compact in size. The cells or a battery must be able to give a constant voltage. Moreover, the voltage of the battery or the cell must not vary during the use.

6.10. JUMPER WIRE

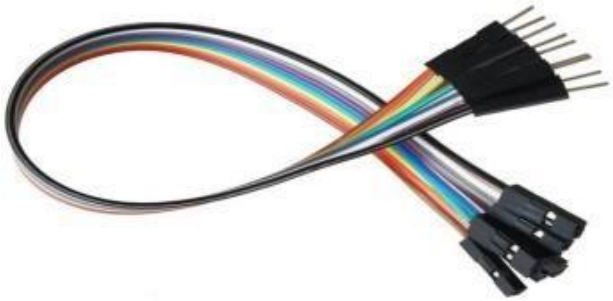


FIG 6.10: Jumper Wire

Jumper wires are simply wiring that have connecting pins at each ends of the connection, allowing them to be used to connect two different points to each other without doing soldering. Jumper wires are typically used with breadboards wireless and other prototyping tools in order to make it easy to change a circuit as needed in a very useful way. Fairly simple. In fact, it doesn't get much more basic than jumper wires. Jumper wires typically come in three versions: male-to-male means connector to each side, male-to-female one side connector and female-to-female receiving side. The difference between each is in the end point of the wire. Male ends have a pin protruding and can insert into things, while female ends do not and are used to plug things into. Male-to-male jumper wire is the most common and what you likely would be used most often. When connecting two ports on a breadboard, a male-to-male.

6.11. CABLE

FIG 6.11: USB cable

It is the most wire we use it to connect Arduino Uno, Arduino Mega 2560, Arduino 101 or any board with the USB port of our computer. USB cable type A/B Standard USB 2.0 cable.

6.12. CIRCUIT DIAGRAM

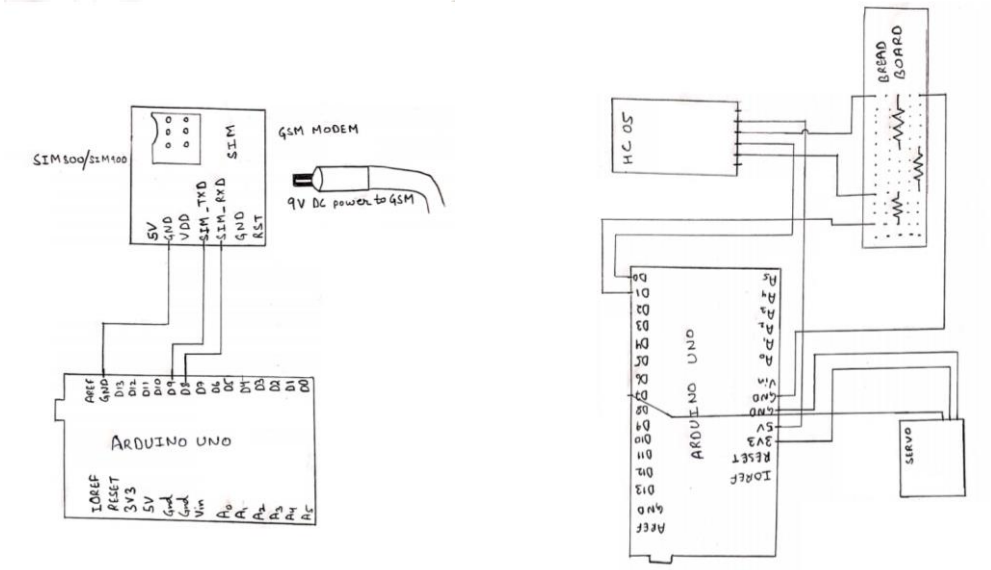


Fig 6.12: Handmade circuit Diagram [6]

CHAPTER 7

IMPLEMENTATION

7.1 Android Studio

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.smartlock.MainActivity">

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:src="@drawable/locked_icon"
        android:layout_marginBottom="250dp"
        android:adjustViewBounds="false"
        android:id="@+id/lock_state_img" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Lock State: LOCKED"
        android:layout_marginBottom="200dp"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:id="@+id/lock_state_text" />

    <Button
        android:text="Connect to Bluetooth Module"
        android:layout_width="wrap_content"
```

```
    android:layout_height="wrap_content"
    android:id="@+id/bluetoothbtn"
    android:layout_alignParentBottom="true"
    android:layout_centerHorizontal="true"
    android:layout_marginBottom="11dp" />

  <Button
    android:text="Lock/Unlock via BT"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_above="@+id/bluetoothbtn"
    android:layout_centerHorizontal="true"
    android:layout_marginBottom="15dp"
    android:id="@+id/lockbtn" />

  <Button
    android:text="Lock/Unlock via SMS"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_above="@+id/lockbtn"
    android:layout_marginBottom="15dp"
    android:id="@+id/locksms" />


</RelativeLayout>
```

MainActivity.java

```
package com.example.smartlock;

import android.Manifest;
import android.content.pm.PackageManager;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Intent;
import android.telephony.SmsManager;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.Set;
import java.util.UUID;
import android.os.Handler;


public class MainActivity extends AppCompatActivity {

    private final String DEVICE_ADDRESS = "98:D3:31:90:82:9A";
    private final UUID PORT_UUID = UUID.fromString("00001101-0000-1000-8000-
00805f9b34fb");
    private final int MY_PERMISSIONS_REQUEST_SEND_SMS=0;
    private BluetoothDevice device;
    private BluetoothSocket socket;

    private OutputStream outputStream;
    private InputStream inputStream;

    Thread thread;
    byte buffer[];

    boolean stopThread;
    boolean connected = false;
    String command;
```

```
String phoneNo;
Button lockbtn, bluetoothbtn, locksms;

TextView lock_state_text;

ImageView lock_state_img;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    lockbtn = (Button) findViewById(R.id.lockbtn);
    bluetoothbtn = (Button) findViewById(R.id.bluetoothbtn);
    locksms = (Button) findViewById(R.id.locksms);
    lock_state_text = (TextView) findViewById(R.id.lock_state_text);

    lock_state_img = (ImageView) findViewById(R.id.lock_state_img);

    bluetoothbtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {

            if (BTinit()) {
                BTconnect();
                beginListenForData();

                command = "3";

                try {
                    outputStream.write(command.getBytes());
                } catch (IOException e) {
                    e.printStackTrace();
                }

            }
        }
    });

    lockbtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {

            if (connected == false) {
                Toast.makeText(getApplicationContext(), "Please establish a connection with the
bluetooth smart lock first", Toast.LENGTH_SHORT).show();
```

```
            } else {
                command = "1";

                try {
                    outputStream.write(command.getBytes());
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }
    });

    locksms.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            send();
        }
    });
  }
  public void send()
    {
            phoneNo = "9999999999";
    if (ContextCompat.checkSelfPermission(this,
            Manifest.permission.SEND_SMS)
            != PackageManager.PERMISSION_GRANTED) {
        if (ActivityCompat.shouldShowRequestPermissionRationale(this,
                Manifest.permission.SEND_SMS)) {
        } else {
            ActivityCompat.requestPermissions(this,


                    new String[]{Manifest.permission.SEND_SMS},
                    MY_PERMISSIONS_REQUEST_SEND_SMS);
        }
    }
  }

  @Override
  public void onRequestPermissionsResult(int requestCode, String permissions[], int[]
grantResults) {
      switch (requestCode) {
        case MY_PERMISSIONS_REQUEST_SEND_SMS: {
            if (grantResults.length > 0
                    && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
                SmsManager smsManager = SmsManager.getDefault();
                smsManager.sendTextMessage(phoneNo, null, "3", null, null);
```

```
                Toast.makeText(getApplicationContext(), "SMS sent.",
                        Toast.LENGTH_LONG).show();
            } else {
                Toast.makeText(getApplicationContext(),
                        "SMS faild, please try again.", Toast.LENGTH_LONG).show();
                return;
            }
        }
    }

}
void beginListenForData()    {
    final Handler handler = new Handler();
    stopThread = false;
    buffer = new byte[1024];

    Thread thread = new Thread(new Runnable() {
        public void run() {
            while (!Thread.currentThread().isInterrupted() && !stopThread) {
                try {
                    int byteCount = inputStream.available();

                    if (byteCount > 0) {
                        byte[] rawBytes = new byte[byteCount];
                        inputStream.read(rawBytes);
                        final String string = new String(rawBytes, "UTF-8");

                        handler.post(new Runnable() {
                            public void run() {
                                if (string.equals("3")) {
                                    lock_state_text.setText("Lock State: LOCKED");
lock_state_img.setImageResource(R.drawable.locked_icon);
                                } else if (string.equals("4")) {
                                    lock_state_text.setText("Lock State: UNLOCKED");
                                    lock_state_img.setImageResource(R.drawable.unlocked_icon);
                                }
                            }
                        });
                    }
                } catch (IOException ex) {
                    stopThread = true;
                }
            }
        }
    });
```

```
        thread.start();
    }

    public boolean BTinit() {
        boolean found = false;

        BluetoothAdapter bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();

        if (bluetoothAdapter == null)       {
            Toast.makeText(getApplicationContext(), "Device doesn't support bluetooth",
Toast.LENGTH_SHORT).show();
        }

        if (!bluetoothAdapter.isEnabled())      {
            Intent enableAdapter = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
            startActivityForResult(enableAdapter, 0);

            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }

        Set<BluetoothDevice> bondedDevices = bluetoothAdapter.getBondedDevices();

        if (bondedDevices.isEmpty())       {
            Toast.makeText(getApplicationContext(), "Please pair the device first",
Toast.LENGTH_SHORT).show();
        } else {
            for (BluetoothDevice iterator : bondedDevices) {
                if (iterator.getAddress().equals(DEVICE_ADDRESS)) {
                    device = iterator;
                    found = true;
                    break;
                }
            }
        }

        return found;
    }

    public boolean BTconnect() {

        try {
            socket = device.createRfcommSocketToServiceRecord(PORT_UUID);
```

```
socket.connect();

            Toast.makeText(getApplicationContext(),
                    "Connection to bluetooth device successful", Toast.LENGTH_LONG).show();
            connected = true;
        } catch (IOException e) {
            e.printStackTrace();
            connected = false;
        }

        if (connected) {
            try {
                outputStream = socket.getOutputStream();            } catch (IOException e) {
                e.printStackTrace();
            }

            try {
                inputStream = socket.getInputStream();            } catch (IOException e) {
                e.printStackTrace();
            }
        }

        return connected;
    }

    @Override
    protected void onStart() {
        super.onStart();
    }
}
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.smartlock">

    <uses-permission android:name = "android.permission.BLUETOOTH" />
    <uses-permission android:name = "android.permission.SEND_SMS"/>
    <application
        android:allowBackup="true"
        android:icon="@mipmap/logo_launch"
        android:label="Smart Lock"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity" android:screenOrientation = "portrait">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

7.2. Arduino IDE code:

```
#include <Servo.h>
#include <EEPROM.h>
#include <SoftwareSerial.h>
SoftwareSerial gsm(9,8);
Servo servo;
char state;
void setup() {
 servo.attach(7);
  if(EEPROM.read(0) == 1)
  {
    servo.write(70);
    delay(200);
  }
  else if(EEPROM.read(0) == 2)
  {
    servo.write(120);
    delay(200);
  }
  Serial.begin(9600);
```

```
}
void loop() {
  if(Serial.available() > 0)
  {
    char data;
    data = Serial.read();
    switch(data)
    {
      case '1':
        if(EEPROM.read(0) == 1)
        {
          EEPROM.write(0, 2);
          Serial.print("3");
          for(int a = 70; a <= 120; a++) {
            servo.write(a);
            delay(15);
            Serial.println(servo.read());
          }
        }
        else if(EEPROM.read(0) == 2)
        {
```

```
        EEPROM.write(0, 1);
        Serial.print("4");
        for(int x = 120; x >= 70; x--) {
          servo.write(x);
          delay(15);
        }
      }
      break;
    case '3':
      if(EEPROM.read(0) == 1)
      {
        Serial.print("4");
      }
      else if(EEPROM.read(0) == 2)
      {
        Serial.print("3");
      }
      break;
  }
 }
}
```

CHAPTER 8

Conclusion

This “Android Based Smart Lock” is a modern use of the door locking system which allows remote access to lock and unlock the door. It is very reliable and easy to install. We can also attach a rechargeable battery with it which to can power backup upto 3-4 hrs. It can also be implemented using cloud computing and use of camera for surveillance. For more security purpose we can also add fingerprint scanner, face recognition etc.

Instead of Bluetooth we can use Wi-Fi for better connectivity and speed. Currently the app is on android platform but we can further make application for iOS or windows so that it can be more user friendly and be on multi platforms. We can also add voice communication to check the person standing outside. This system requires minimum hardware and support. It requires minimum power and unlocking of door happens in 4 seconds.

REFERENCES

[1] R. A. Ramlee, D. H. Z. Tang, and M. M. Ismail, "Smart home system for Disabled People via Wireless Bluetooth," in *2012 International Conference on System Engineering and Technology (ICSET)*, 2012, pp. 1–4, doi: 10.1109/ICSEngT.2012.6339347.

[2] K. Lian, S. Hsiao, and W. Sung, "Home safety handwriting pattern recognition system," in *2012 IEEE 11th International Conference on Cognitive Informatics and Cognitive Computing*, 2012, pp. 477–483.

[3] M. Sahani, C. Nanda, A. K. Sahu, and B. Pattnaik, "Web-based online embedded door access control and home security system based on face recognition," in *2015 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2015]*, 2015, pp. 1–6.

[4] K. S. G. B. Lia and W. S. M. S. G. B. Sanjaya, "Door-automation system using bluetooth-based android for mobile phone DOOR-AUTOMATION SYSTEM USING BLUETOOTH-BASED ANDROID FOR MOBILE PHONE," no. January 2014, 2016.

[5] A. Mishra, S. Sharma, S. Dubey, and S. K. Dubey, "Password Based Security Lock System," no. 02, pp. 100–103, 2014.

[6] V. B. Kitovski, "Electronic devices and circuit theory, 6th edition, R. Boylestad and L. Nashelsky, Prentice Hall International Inc., 1996, 950 pp. A4 (paperback)," *Microelectronics J.*, 1998, doi: 10.1016/s0026-2692(98)90008-8.

Names of Websites referred

https://www.hackerearth.com/blog/developers/arduino-programming-for-beginners/

https://maker.pro/pic/tutorial/introduction-to-python-serial-ports

http://www.learn-android.com/