CSE 3009

INTERNET OF THINGS

PROJECT REPORT ON

SMART LOCKER

Completed under the guidance of

Dr. Krishnamoorthy A

Submitted by

Lade Vivek	**20BCE0432**
Saratchandra Hemanth T	**20BCE0510**
Srikanth S	**20BCE2343**
Kagana Karthikeya	**20BCE0490**

Table of Contents

1. Introduction **2**
1.1 Purpose 2
1.2 Document Conventions 2
1.3 Intended Audience and Reading Suggestions 2
1.4 Product Scope 2
1.5 References 2
2. Overall Description **3**
2.1 Product Perspective 3
2.2 Product Functions 3
2.3 User Classes and Characteristics 3
2.4 Operating Environment 3
2.5 Design and Implementation Constraints 3
2.6 User Documentation 3
2.7 Assumptions and Dependencies 3
3. External Interface Requirements **4**
3.1 User Interfaces 4
3.2 Hardware Interfaces 4
3.3 Software Interfaces 4
4. System Features **5**
4.1 System Feature 1 5
4.2 System Feature 2 6
5. Other Nonfunctional Requirements **7**
5.1 Performance Requirements 7
5.2 Safety Requirements 7
5.3 Security Requirements 7
Appendix A: Glossary **11**
Appendix B: Analysis Models **12**

1. Introduction

1.1 Purpose

In our everyday lives, we use lockers to keep our valuables safe. Lockers with a lock and key are sometimes problematic, as humans, we tend to become lethargic to search for our key. Incidents occur where we lose our keys when need can't find them, we tend to panic. The agenda of our project is to design a smart lock which generates an OTP to the user when triggered and unlocks the lock only if the OTP is provided is the provided OTP is wrong a red LED starts glowing up for every wrong attempt and the user has three attempts to enter the correct OTP is the user fails to enter the correct OTP th buzzer starts buzzing and alerts the surroundings .

1.2 Document Conventions

This document uses IEEE format, and all the acronyms and technical terms were well defined in the glossary section

1.3 Intended Audience and Reading Suggestions

This document is intended for everyone using the Smart locker. All the users can start reading the document from introduction section and then reading the overall description section and then the system features and the developers can read the same along with external interface requirements and system features and other non-functional requirements...etc.

1.4 Product Scope

Smart locker is the upgraded version of the normal lock it uses an OTP to unlock the door. Smart lockers generates an OTP when triggered by the user and the user should enter the OTP which is received to them through SMS to their registered mobile number then the lock takes the OTP as input through push buttons and opens the lock if the OTP is correct and if the OTP entered is wrong the RED led glow and the user has 3 attempts to enter the correct OTP and if the user fails to enter the correct OTP the three LEDs glows and the buzzer starts buzzing.

1.5 References

- Arduino Coding Basics - JavaTpoint
- https://create.arduino.cc/projecthub/shafeekashraf44/otp-based-locker-e6bc8c
- https://www.electronicsforu.com/electronics-projects/otp-based-smart-wireless-locking-system
- How to Connect an LCD Display to Your Arduino | Arduino | Maker Pro
- Interface GSM Module to Arduino - Send and Receive SMS (circuitstoday.com)
- https://www.wellpcb.com/arduino-breadboard.html
- Send SMS using a GSM module connected to Arduino (tutorialspoint.com)
- How to Send Message from GSM Module using Arduino - (duino4projects.com)

2. Overall Description

2.1 Product Perspective

Generally, when it comes to locks people usually use a mechanical lock which can be easily accessed by a duplicate key and usually the locks won't last more than 6 months and in general people won't use a smart lock even if they want to, they are expensive and bit complicated for normal users. So we decided to design a smart lock which is easy to use and has simple interface and added security through OTP generated and with minimal cost.

2.2 Product Functions

The product function is to generate an OTP when triggered by the user and another function of the product is to unlock the door if the entered OTP is correct and another function of the product is to glow an red led every time when a wrong OTP is entered and the user has three attempts to enter the correct OTP if the user fails to do so then the buzzer starts buzzing this are the main functions of the product.

2.3 User Classes and Characteristics

The product has only one user, he is the one who triggers the OTP and unlocks the lock by entering the correct OTP.

2.4 Operating Environment

The product operates through Arduino UNO and GSM module and the user enters the OTP through push buttons provided.

2.5 Design and Implementation Constraints

The project should be completed within the time given and there are no expertise in the given area in our team and we have to make the product with the minimum cost possible in order to make the product cost effective.

2.6 User Documentation

The user will be provided with a small user manual on how to use the product and what push buttons are to be pressed to generate the OTP and unlock the lock.

2.7 Assumptions and Dependencies

The user having proper network connection in there are so that they can get the OTP with no time delay and they are able to unlock the door without any difficulties.

3. External Interface Requirements

3.1 User Interfaces

The user interface is very simple the user needs to give his mobile number and needs to enter the OTP received through the push buttons.

3.2 Hardware Interfaces

We are using Arduino UNO and GSM module to generate a OTP to the user mobile number and a servo motor and an buzzer and four pushbuttons and an LCD to display the number of attempts remaining for the user to enter the correct OTP and breadboard for connections and four LEDs to tell the user if the entered OTP is correct or not and a battery to power the GSM module.

3.3 Software Interfaces

We are using Arduino IDE for coding the Arduino UNO and GSM module.

4. System Features

4.1 OTP Generation

4.1.1 Description and Priority

When ever the user needs to unlock the door, the user has to trigger the OTP generation by pressing the fourth ush button and then the Arduino code starts generating an OTP and with help of the GSM module the OTP is sent to the user mobile number through SMS and the priority is High.

4.1.2 Stimulus/Response Sequences

If the user presses the fourth pushbutton the OTP should be generated and to be sent to the user's mobile number.

4.1.3 Functional Requirements

User's mobile number, Arduino UNO and GSM module are needed foe the generation of the OTP to the user's mobile number.

REQ 1: Arduino UNO

It is an open-source microcontroller board based on the Microchip ATmega328P microcontroller. As we can see in Figure 2 the board is equipped with sets of digital and analog input / output (I/O) pins that may be interfaced to various expansion boards such as Motor Driver Shield L293D as we used it in this project and other circuits. The board has fourteen digital I/O pins (six of them are capable of PWM output), six analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable. It can be powered by the type B USB cable or by an external 9-volt battery, though it accepts voltages ranging from 7 volts to a maximum of 20 volts. Each of the 14 digital pins and 6 analog pins on the board operate at 5 volts and each pin can provide or receive 20mA as the recommended operating condition and has an internal pull-up resistor of 20 – 50K ohms. A maximum of 40mA must not be exceeded on any pin to avoid permanent damage to the microcontroller

REQ 2: GSM module

Global system for mobile communication (GSM) is a globally accepted standard for digital cellular communication. GSM is the name of a standardization group establishedin1982tocreate a common European mobile telephone standard that would formulate specifications for a pan-European mobile cellular radio system operating at 900MHz. A GSM modem is a wireless modem that works with a GSM wireless network. A wireless modem behaves like a dial-up modem. The main difference between them is that a dial-up modem sends and receives data through a fixed telephone line while a wireless modem sends and receives data through radio waves. The working of GSM modem is based on commands, the commands always start with AT (which means ATtention) and finish with a character. For example, the dialling commands ATD; ATD3314629080; here the dialling command ends with semicolon. The AT commands are given to the GSM modem with the help of PC or controller. The GSM modem is serially interfaced with the controller with the help of MAX 232

REQ 3: Mobile number

REQ 4: Push buttons

4.2 OTP validation

4.2.1 Description and Priority

After the OTP is generated and received by the user now the user has to enter the OTP through the push buttons and after giving the input the Arduino code now must check the OTP provided by the user is correct or not if the OTP provided is correct it has turn the servo motor by ninety degrees otherwise a red LED has to glow and the LCD should show how many attempts are remaining for the user to enter the correct OTP and it has to trigger the buzzer if the user has exceeded their limit of wrong attempts and the priority is High.

4.2.2 Stimulus/response Sequences

The product has to check if the OTP provided by the user is correct or not and has to turn the servo motor by certain angle if the given OTP is correct and if the OTP is wrong it has glow red LED and it has to display how many attempts are remaining in the LCD and has to start buzzing the buzzer if the user exceeds their limit of wrong attempts.

4.2.3 Functional Requirements

User's mobile number, Arduino UNO and GSM module and a servo motor and LCD and four LEDs and four push buttons and a buzzer.

REQ 1: Arduino UNO

REQ 2: GSM module

REQ 3: Mobile number

REQ 4: Push buttons

REQ 5: LCD

REQ 6: LEDs

REQ 7: Breadboard

REQ 8: Wires

REQ 9: Buzzer

REQ 10: Servo motor

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The push buttons, LCD and LED should be working properly without any errors like the pushbuttons should respond after clicking them and the LCD should be bright enough and should show the correct information and LED should be bright enough and the servo motor needs to be working and the buzzer needs to be loud enough and the GSM module should send the OTP to the user and the Arduino uno should validate and open the lock without any error.

5.2 Safety Requirements

All the parts of the product should be intact and working properly and the Arduino uno and GSM module and remaining parts are needed to be covered with a card box or with some protection and only the pushbuttons are to be accessed by the user.

5.3 Security Requirements

The OTP generation and OTP validation need to be working properly. The GSM module needs to work without any malfunctioning the OTP should get delivered to the user within no time and correct OTP needs to be delivered to the user and not the wrong one. And the OTP validation should be done properly without any error and should open the lock only when the user enters the correct OTP, and the OTP should be only sent to the user and not any other person other than the user.

SOURCE CODE:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <SoftwareSerial.h>
#include<Servo.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
SoftwareSerial mySerial(9, 10);
int pB[4]={2,3,4,5};
int pass[4];
int q[4]={0,0,0,0};
int outPin[4]={11,8,7,6};
char gsmPass[4];
int epass[4];
const int buzzer = 13;
int servoPin=12;
Servo servo1;
int t;
float d;
int a,ch;
int co=0;
char c;
int a2=7;
int a3=6;
int ma;
int lo=0;
```

```
float aw=256;
//LiquidCrystal lcd(12,13,11,10,9,8);
void setup()
{
  int i=0;
  lcd.begin();
  lcd.backlight();
  lcd.clear();
  lcd.print("to open press ");
  lcd.setCursor(0,1);
  lcd.print(4);
  for (i=0;i<4;i++)
  {
    pinMode(pB[i],INPUT);
    pinMode(outPin[i],OUTPUT);
  }
  pinMode(buzzer,OUTPUT);
  servo1.attach(servoPin);
  //pinMode(led, OUTPUT);
  mySerial.begin(9600);
  Serial.begin(9600);

}

void loop()
{
  int i=0;
  a=0;
  while(lo==0)
  {
    while(q[i]<4)
    {
      q[i]++;
      if (q[i]==4)
        q[i]=0;
      i++;
      if (i==4)
      i=0;
      if(digitalRead(pB[3]))
      {
        a++;
        Serial.println("...");
         break;
      }
      delay(50);
      Serial.print(i);
      Serial.println(q[i]);
    }
    if(a)
    {
      lo++;
      break;
    }
  }
```

```
Serial.print("pass is : ");
for(t=0;t<4;t++)
{
  pass[t]=q[t];
  if (q[t]==0)
  {
    pass[t]=4;
  }
  gsmPass[t]=pass[t]+'0';
  Serial.print(pass[t]);
}
Serial.println(gsmPass);
mySerial.println("AT+CMGF=1");    //Sets the GSM Module in Text Mode
delay(1000);  // Delay of 1 second
 mySerial.println("AT+CMGS=\"+917659898992\"\r"); // Replace x with mobile number
   delay(1000);
   mySerial.println(gsmPass);// The SMS text you want to send
   delay(100);
   mySerial.println((char)26);// ASCII code of CTRL+Z for saying the end of sms to  the module
    delay(1000);

mySerial.println(gsmPass);// The SMS text you want to send
mySerial.println("is your passcode");
delay(100);
mySerial.println((char)26);// ASCII code of CTRL+Z for saying the end of sms to  the module
delay(1000);
Serial.println(".");
int j=0;
ma=0;
a=0;
t=check();
delay(1000);
lcd.setCursor(0,0);
lcd.print("enter passcode");
while (co==0)
{
  for (i=0;i<4;i++)
  {
    t=check();
    epass[i]=t;
    Serial.print(epass[i]);
    Serial.print("==");
    Serial.println(pass[i]);
    delay(450);
    c=t;
    lcd.setCursor(i,1);
    lcd.print(t);
    delay(200);
    lcd.setCursor(i,1);
    lcd.print("*");

  }
  for (i=0;i<4;i++)
  {
   if (epass[i]==pass[i])
   {
     j++;
     Serial.println(j);
   }
  }
  if (j==4)
       {
         Serial.println("correct password");
         lcd.clear();
         lcd.setCursor(0,0);
        lcd.print("Correct Password");
       co=co+1;
       digitalWrite(outPin[0],HIGH);
       digitalWrite(outPin[1],LOW);
       digitalWrite(outPin[2],LOW);
       digitalWrite(outPin[3],LOW);
       servo1.write(90);
       break;

  }
 else
 {
   Serial.println("incorrect password");
   lcd.clear();
   lcd.setCursor(0,0);
   lcd.print("incorrect Password");
   ma++;
   if (ma<3)
   {
     lcd.setCursor(0,1);
     lcd.print(3-ma);
     lcd.print(" attempts remaining");
     delay(1000);
     lcd.clear();
     lcd.setCursor(0,0);
     lcd.print("enter password:");
   }
   else if (ma==3)
   {
     lcd.setCursor(0,1);
     lcd.print("max limit exceeded");
     tone(buzzer,1000);
     delay(1000);
     noTone(buzzer);
     co=2;
   }
   /*if (ma==1)
   {
     digitalWrite(a3,LOW);
   }
   else if (ma==2)
   {
     digitalWrite(a2,LOW);
```

```
        digitalWrite(a3,HIGH);
      }
      else if (ma==3)
      {
        digitalWrite(a2,HIGH);
        digitalWrite(a3,LOW);
        break;
      }*/
      for(i=1;i<=ma;i++)
      {
        digitalWrite(outPin[i],HIGH);
        delay(50);
      }
      j=0;
    }
  }
}

int check()
{
  int o=0;
  int i,j;
  while (o==0)
  {
    for (i=0;i<4;i++)
    {
      j=digitalRead(pB[i]);
      if (j==HIGH)
      {
        o=i+1;
        return o;
        break;
      }
    }
  }
}
```

Appendix A: Glossary

GSM: Global System for Mobile Communication

LED: Light-Emitting Diode

OTP: One-Time Password

LCD: Liquid-Crystal Display

Appendix B: Analysis Models

Flow Chart:

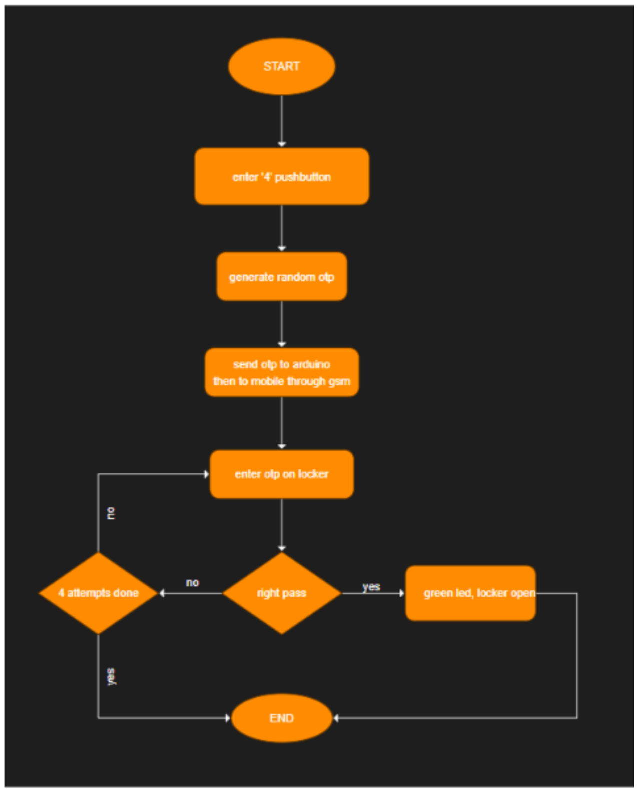

Fig-(i)

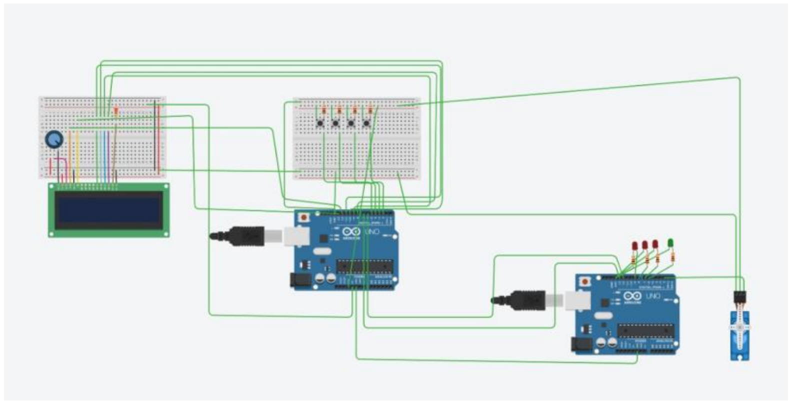

Fig-(ii)