SMART PASS

Report submitted to

Abhishek Sharma

LNM Institute of Information Technology

for the award of the degree

of

Bachelor of Technology
in Electronic Engineering & Communication
By

Dhruvraj Singh Rawat

LNMIIT
The LNM Institute of
Information Technology

DEPARTMENT OF ELECTRONIC ENGINEERING
LNM INSTITUTE OF INFORMATION TECHNOLOGY, JAIPUR
May 2018

DECLARATION

I hereby certify that,

a) the work contained in this report is original and has been done by me under the guidance of my supervisor(s).

b) the work has not been submitted to any other Institute for any degree or diploma.

c) I have followed the guidelines provided by the Institute in preparing the report.

d) I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

e) Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the report and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

Signature of the Student
Dhruvraj Singh Rawat (15UEC022)

Date: 11-May-2018

CERTIFICATE

This is to certify that the Iot Project Report entitled, "Smart Pass" submitted by Mr. "Dhruvraj Singh Rawat" to LNM Institute of Information Technology, is a record of bonafide Project work carried out by him under my and guidance and is worthy of consideration for the award of the degree of Bachelor of Technology in Electronics and Communication Engineering of the Institute.

Mr. Abhishek Sharma	Mr. Kumar Padmanabh
(Signature)	(Signature)

Department of Electronics and Communication Engineering
LNM Institute of Information Technology
Jaipur

Date: 11-May-2018

ACKNOWLEDGMENT

This document is prepared by the inspiration received from Mr. Kumar Padmanabh, Department of CSE and Mr. Abhishek Sharma, Department of ECE. Many colleagues at LNMIIT have carefully read and improved the document; their contributions are gratefully acknowledged.

Dhruvraj Singh Rawat (15UEC022)

TABLE OF CONTENTS

Abstract

In this project, I to build an **RFID Powered Smart Pass Access Control System using Raspberry Pi microprocessor**. We have already seen this kind of services implemented in big hospitals or companies and even in different colleges across the world: National Ilan University, we might have seen how they used RFID based locks to restrict access to certain areas. In hospitals, only hospital staff with approved RFID tags can access Intensive Care Unit and such restricted areas. The same system can be used in other systems like RFID based door lock or RFID based access management system and even in Smart Card payment systems.

In LNMIIT we should always carry our **Institute ID** with us thus if we need to board a college bus from the main gate we have to show our bus pass and well as our college ID. We can make our bus pass redundant if we can make smart use of our college ID

Introduction

Proposal

At present, there are generally two methods for access control: recording on the paper or managing with technology. However, managing with paper-based technology has a critical drawback, they can get lost, can get easily destroyed with external conditions and cost an environment damage with cutting down of tree for making pulp to produce paper.

According to the requirement of solving the dynamics of managing the large amount of crowd access to a restricted resource a radio frequency identification (RFID) technology is proposed in this project. RFID technology is stronger in its ability to withstand harsh environment, fast access time are its key component. It can automatically complete the identification and management tasks without stopping and getting tired. When it come to the access of large amount of people RFID technology will be more powerful. This system uses RFID technology to automatically capture the detail of the person as it passes by and dynamically sending the information to the remote server for validation. Thus, speeding up the entire process of from ID capturing and its validation.

This project is clearly useful in the scenario of LNMIIT, where there is a huge rush outside the gate whenever the college bus arrives from the city to get their bus pass punched. Apart from this we also have to produce our college ID for validation that this bus pass in fact belongs to us thus further increasing the time to process the student. Our college ID's are embedded with a RFID identification chip inside with a unique code allotted to all the students thus if we combine with server containing the details to process students faster.

LNMIIT

The LNM Institute of Information Technology

UG Student

Valid Upto
30/06/2019

DHRUVRAJ SINGH RAWAT

15UEC022

D.O.B. : 14/07/1996

Figure 1: RFID Chip embedded inside our college ID

Objective

Goals:

- To implement a RFID based access control system powered by raspberry pi with server implementation that can securely verify each passing student in timely and fast fashion.

Functions:

- Get the RFID information from the user via sensor.
- Transmit the information to the server via raspberry pi
- On server, Validate the information weather the RFID bearer is authorized to enter or not.
- Return the output back to the Raspberry pi.

Literature Survey

Shelf Management
Check in/out Service
Self Check in/out
Book Drop
Tagging
Anti-Theft Detection

The paper from Robert Waszkowski and his colleges in the paper “Evaluating the Impact of Testing Document Management System with RFID Tags Software on the Level of its Reliability” attempts to describe a method of using formal evaluating the impact of testing of software on the level of its reliability. Presented method has been developed on the basis of formal model of software testing process that has been constructed in the paper for some assumed testing scheme. The method under the investigation has been proposed for the mean of planning the software testing process of document management system with RFID tags.

Furthermore, paper from Mehak Tanveer from University of Gujrat showed the use of RFID based technology on smart traffic management system using the Internet of Things and a decentralized approach to optimize traffic on the roads and intelligent algorithms to manage all traffic situations more accurately. This proposed system is overcoming the flaws of previous traffic management systems. The system takes traffic density as input from RFID sensors which is then abstracted using different technique and sensors data, resultantly giving output as signals management.

Proposed System

Problem Statement

Build a Realtime system RFID powered smart pass access control system using EM-18 RFID Module and Raspberry-pi. Also, to implement a server in FLASK framework with dedicated SQL database maintaining all the user's information. Apart from that to establish the interfacing between the Raspberry pi and the Remote Server.

Solution Control Flow

- Student taps its RFID card in the RFID scanner.
- The Information (unique rfid_no) is transmitted to the Raspberry pi via serial port.
- The Raspberry pi encodes the information (rfid_no) into the Json format and transmit it to the Server
- The Server parses the information and check the statement balance regarding that rfid_no in SQL database and return a Boolean response signifying whether to grant access or not.
- The response is transmitted back to the Raspberry pi via WIFI and it parses the information and takes the appropriate step.

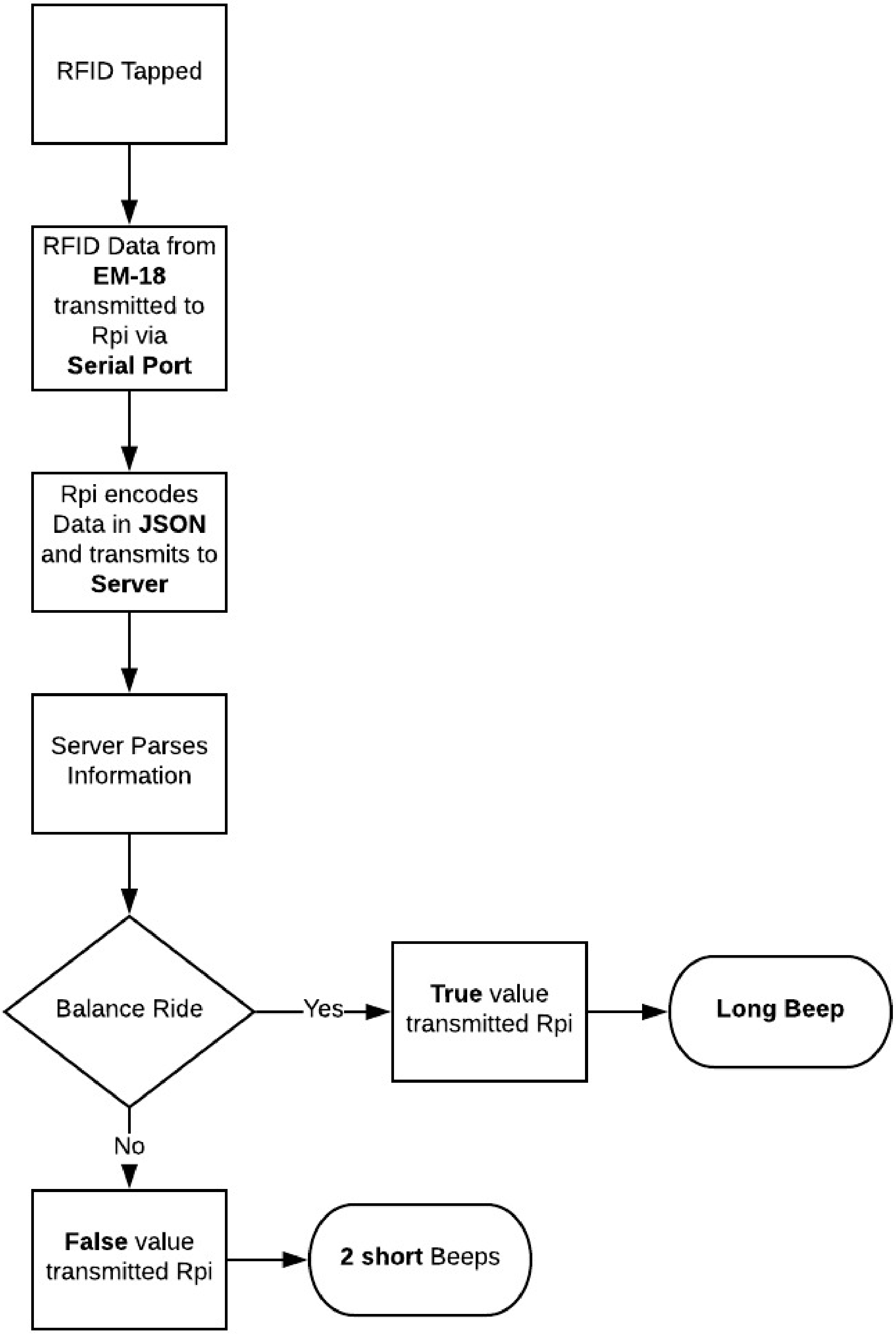

RFID Tapped
RFID Data from **EM-18** transmitted to Rpi via **Serial Port**
Rpi encodes Data in **JSON** and transmits to **Server**
Server Parses Information
Balance Ride
Yes
True value transmitted Rpi
Long Beep
No
False value transmitted Rpi
2 short Beeps

Figure 2: Control Flow chart of the system

Components

- **Raspberry Pi v3:**

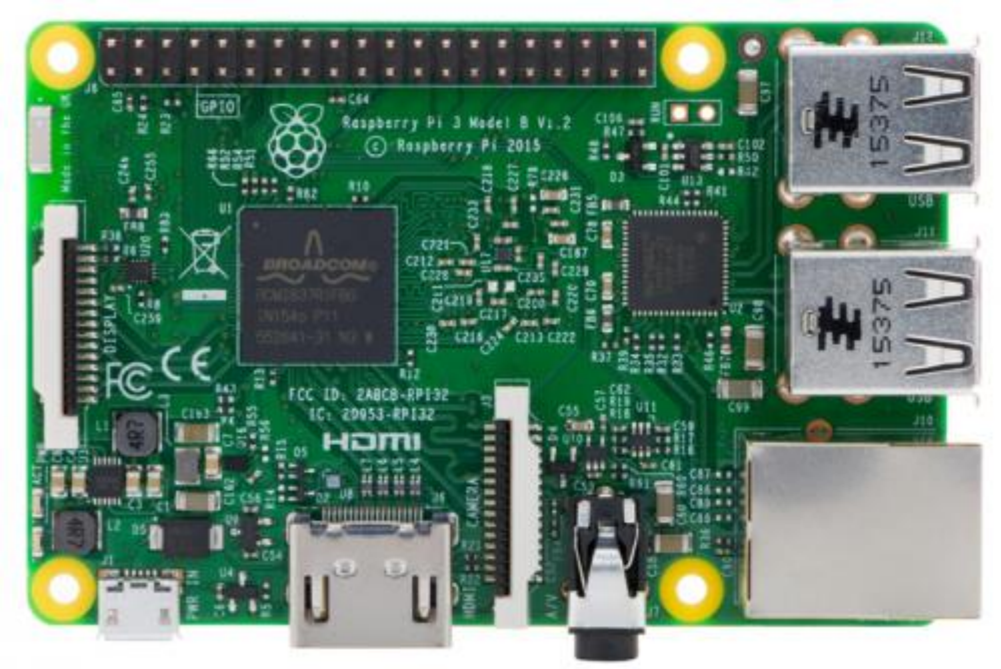


Figure 3: Raspberry Pi 3

The Raspberry Pi 3 Model B+ is the latest product in the Raspberry Pi 3 range, boasting a 64-bit quad core processor running at 1.4GHz, dual-band 2.4GHz and 5GHz wireless LAN, Bluetooth 4.2/BLE, faster Ethernet, and PoE capability via a separate PoE HAT

The dual-band wireless LAN comes with modular compliance certification, allowing the board to be designed into end products with significantly reduced wireless LAN compliance testing, improving both cost and time to market.

The Raspberry Pi 3 Model B+ maintains the same mechanical footprint as both the Raspberry Pi 2 Model B and the Raspberry Pi 3 Model B.

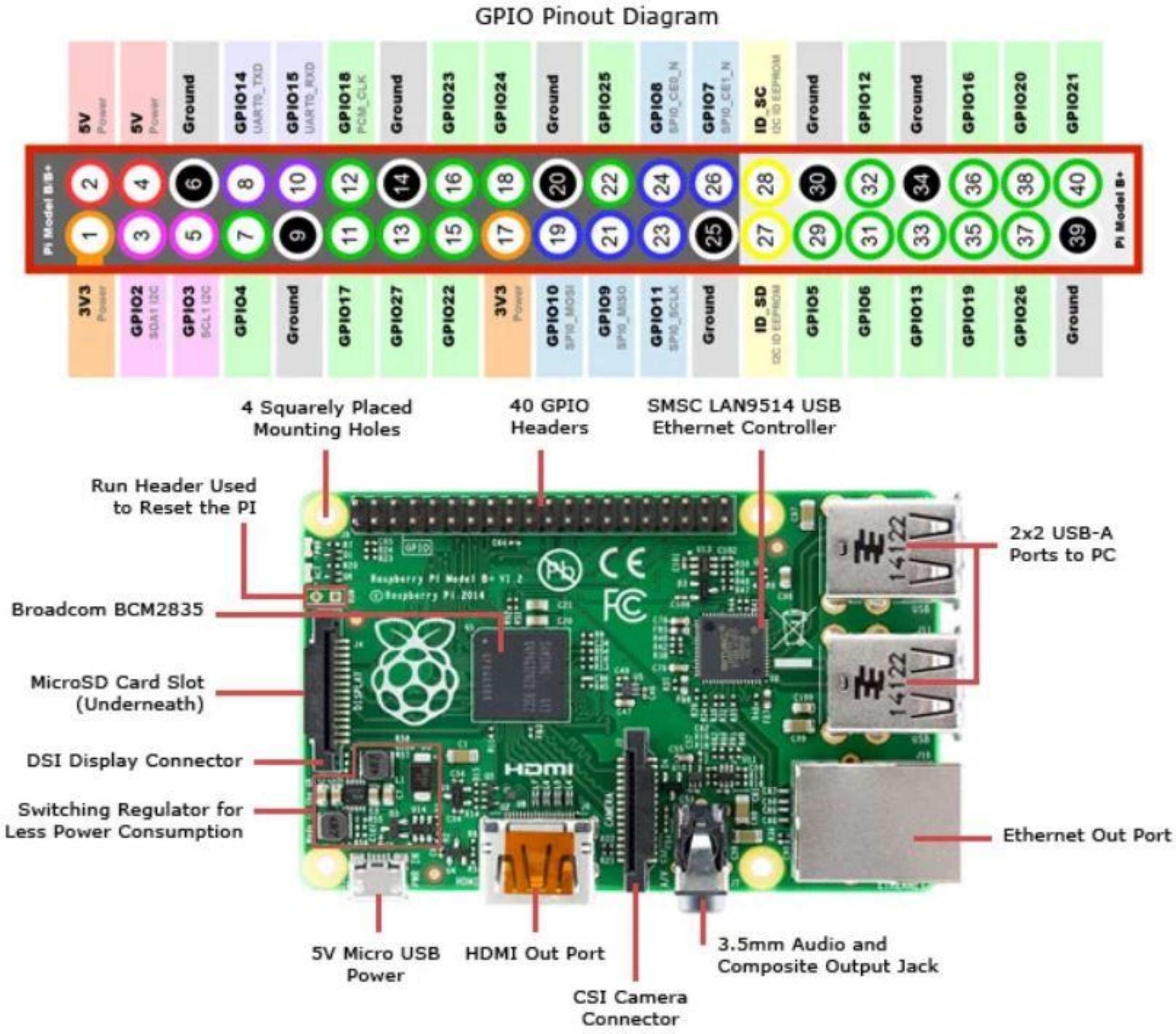


Figure 4: GPiO Layout of Raspberry Pi 3

- **EM - 18**

EM-18 RFID reader is one of the commonly used RFID reader to read 125KHz tags. It features low cost, low power consumption, small form factor and easy to use. It provides both UART and Wiegand26 output formats. It can be directly interfaced with microcontrollers using UART and with PC using an RS232 converter.

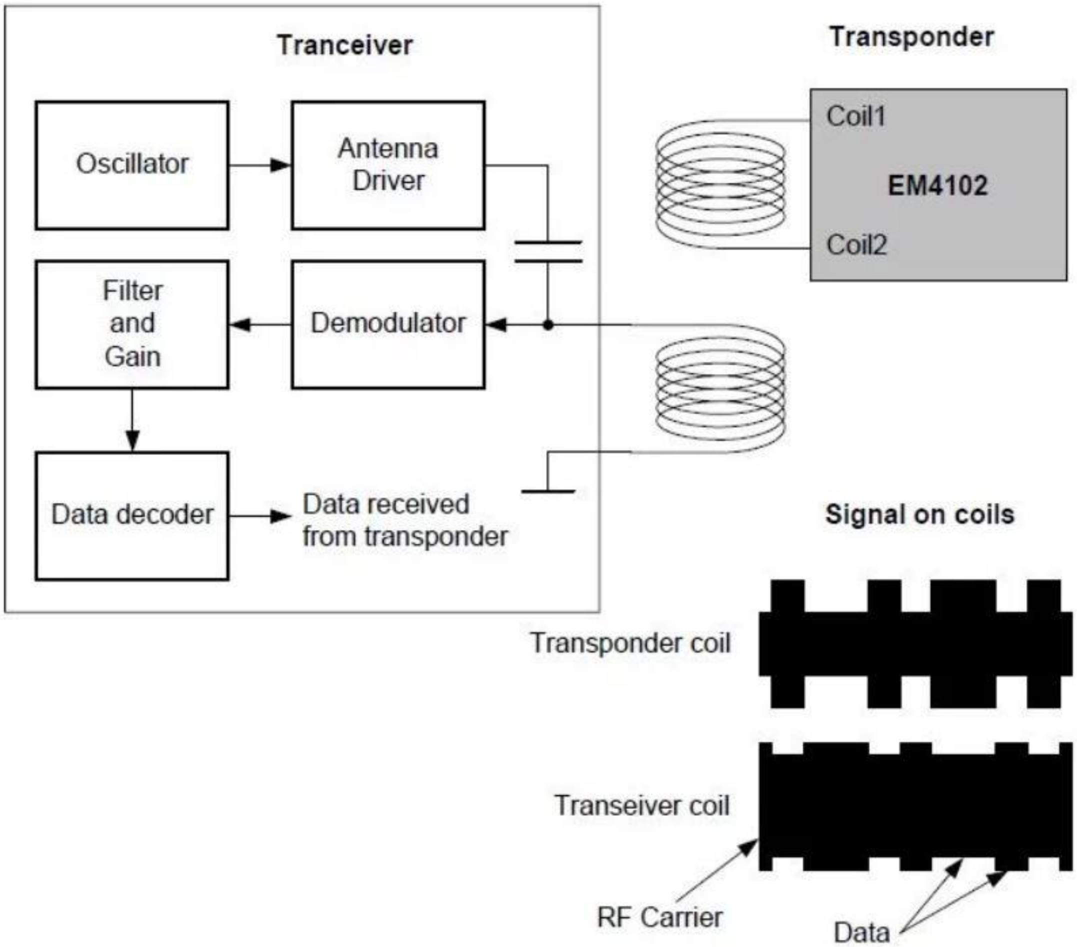


Figure 5: Internal Layout EM – 18

Circuit Diagram

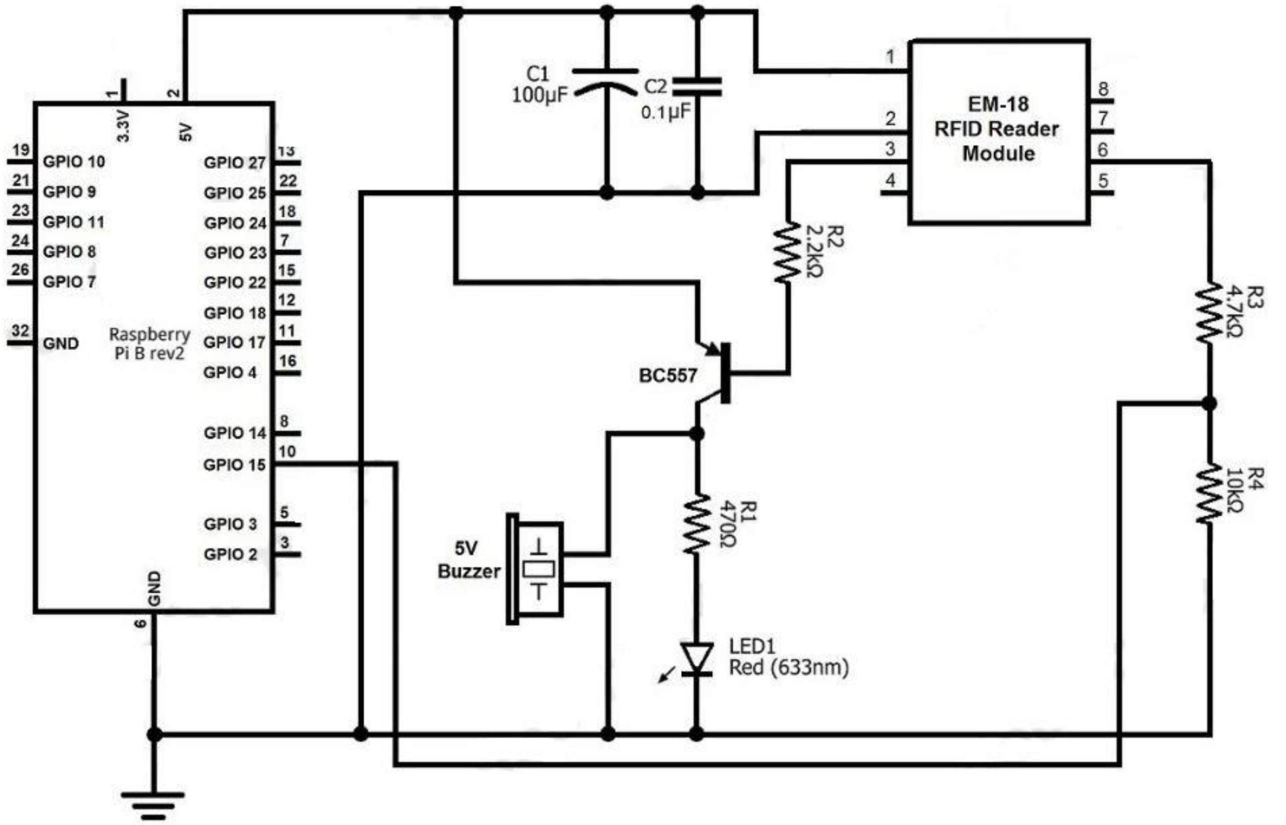


Figure 6: RFID Scanner Circuit Diagram

Technologies Used

Client Side

- Python 2.7
- Request: Http Library for Python
- Python Json: Encoder and Decoder for parsing Json requests & response

Server Side

- Python 2.7
- Flask Framework: Python micro web framework
- SQLAlchemy: Python SQL database framework
- Bootstrap Framework: WebUI creation

Description of Important Technologies Used

- **Python**

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- **Flask Framework**

Flask is a micro web framework written in Python and based on the Werkzeug toolkit and Jinja2 template engine. It is BSD licensed.

Flask is called a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more regularly than the core Flask program.

- **SQLAlchemy**

SQLAlchemy (source code) is a well-regarded database toolkit and object-relational mapper (ORM) implementation written in Python. SQLAlchemy provides a generalized interface for creating and executing database-agnostic code without needing to write SQL statements.

Implementation and Module Creation

- Database Creation using SQLAlchemy

```
class user(db.Model):

    pin = db.Column(db.String(10))
    rollno = db.Column(db.String(10))
    name = db.Column(db.String(250), nullable=False)
    email = db.Column(db.String(250), nullable=False)
    typeCard = db.Column(db.String(5))
    rfidno  = db.Column(db.String(50))

          id = db.Column(db.Integer, primary_key = True)

      class history(db.Model):

          id = db.Column(db.Integer, primary_key = True)
          name = db.Column(db.String(250), nullable=False)
          rfidno  = db.Column(db.String(50))
          day = db.Column(db.String(50))
          time = db.Column(db.String(50))

      class statement(db.Model):

          id = db.Column(db.Integer, primary_key = True)
          balance_trip = db.Column(db.String(8))
          rfidno  = db.Column(db.String(50))
          name = db.Column(db.String(250), nullable=False)
```

- Server API Ends for Communication with Client Side

```
@app.route("/api_responce", methods = ['GET', 'POST'])
def api_responce():
    if request.method=='POST':

        rfidno=request.args.get('rfid_no')

        user_balance = db.session.query(statement).filter_by(rfidno=rfidno).one()
        user_info=db.session.query(user).filter_by(rfidno=rfidno).one()

              balance_available=int(user_balance.balance_trip)


              if balance_available>0:
                  balance_available=balance_available-1
                  user_balance.balance_trip=balance_available
                  db.session.commit()

                  day=datetime.datetime.now().strftime("%d/%m/%y")
                  timee=time.strftime("%H:%M:%S")

                  trip_update=history(name=user_balance.name,rfidno=rfidno,day=day,time=timee)
                  db.session.add(trip_update)
                  db.session.commit()

                  mailer_server(user_info.email,day,timee,balance_available)

                  output={
                      'status':1,

                  }
                  return json.dumps(output)
              else:
                  output={
                      'status':0,
                  }
                  return json.dumps(output)

          elif request.method=='GET':

              response=make_response(json.dumps('GET is working',200))

              response.headers['Content-Type']='application/json'
              return response
```

- Mailing System

```
def mailer_server(reciever_email,datee,timee,leftee):

    # Python code to illustrate Sending mail from
    # your Gmail account
    import smtplib

    # creates SMTP session
    s = smtplib.SMTP('smtp.gmail.com', 587)

          # start TLS for security
          s.starttls()

          # Authentication
          s.login("feedbacklnmiit@gmail.com", "helloworld12345")

          # message to be sent
          SUBJECT="Alert: BusPass Used"
          message_body = "Your Bus was used on "+str(datee)+" at "+str(timee)+" ! Remaining Trip Left "+str(leftee)

          message = 'Subject: {}\n\n{}'.format(SUBJECT, message_body)
          # sending the mail
          s.sendmail("digitalcampuslnmiit@gmail.com", reciever_email, message)

          # terminating the session
          s.quit()
```

Results

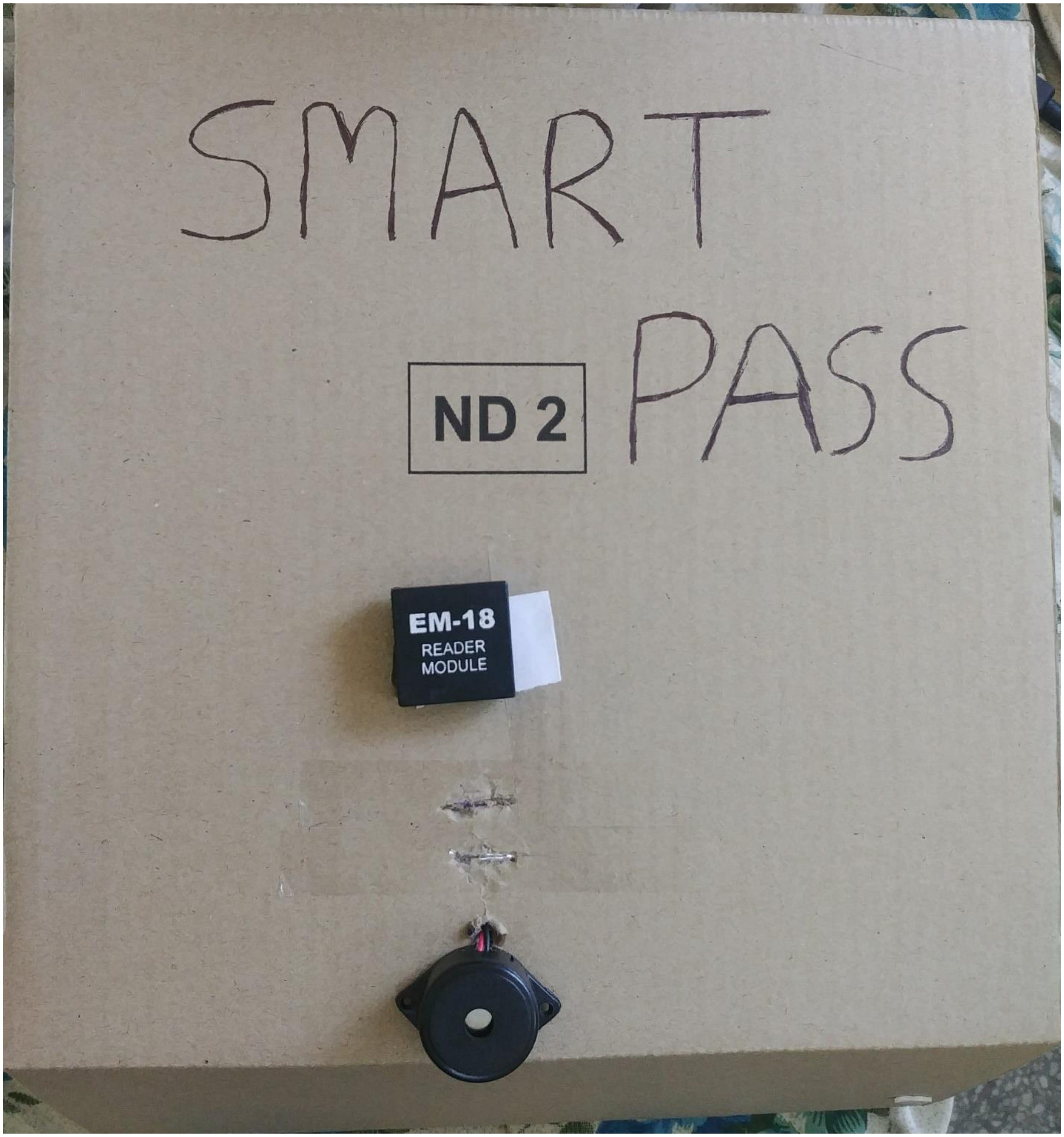


Figure 7: Front Panel of the System

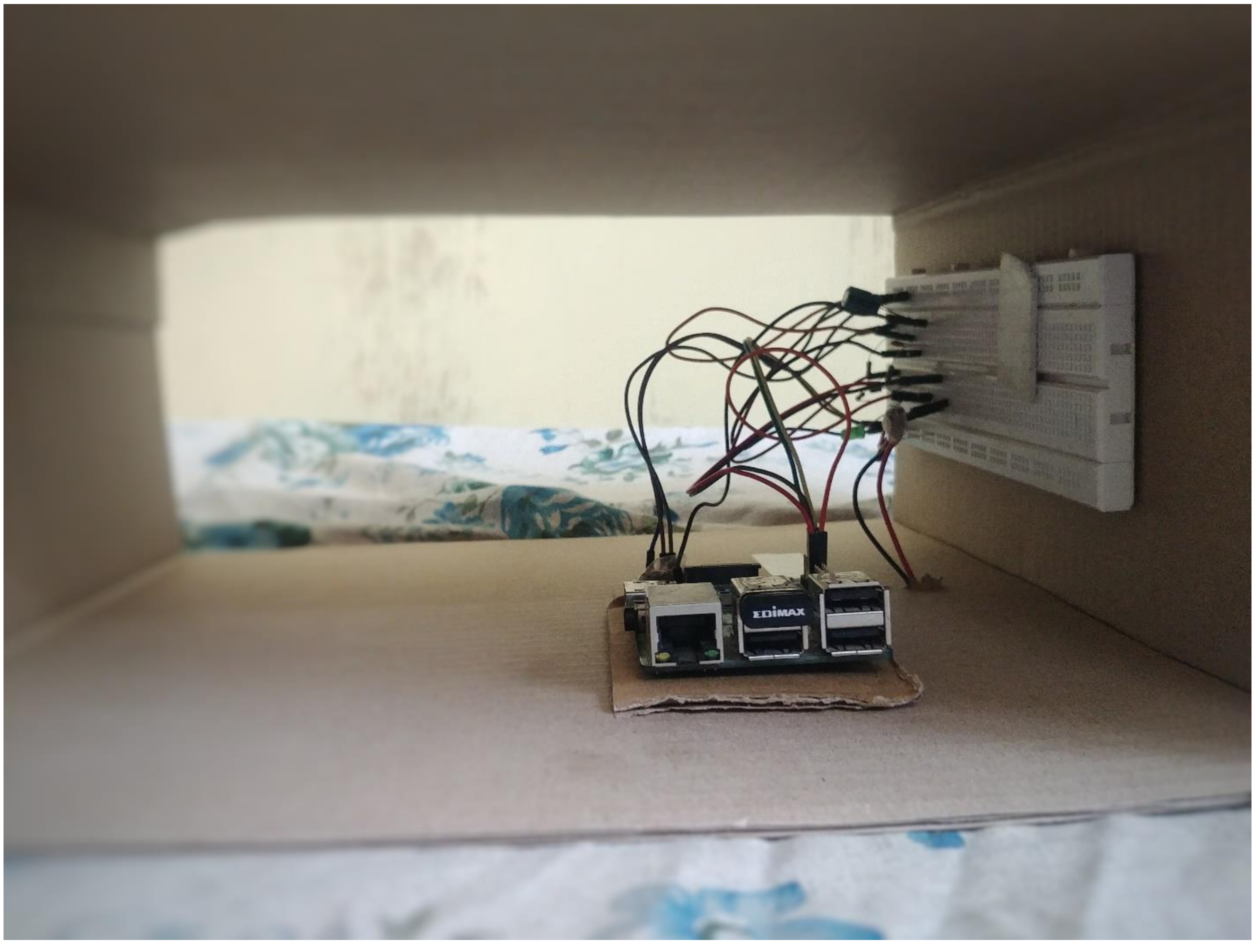
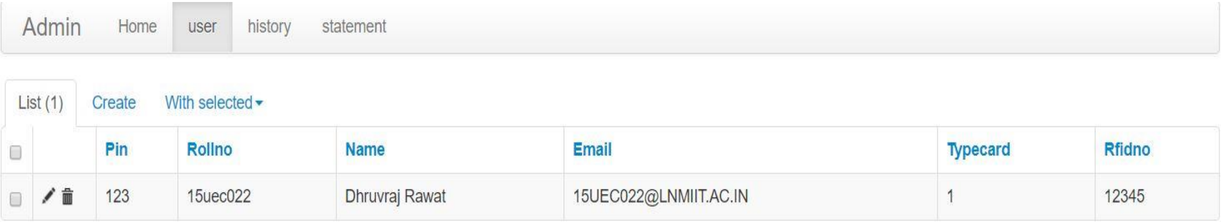
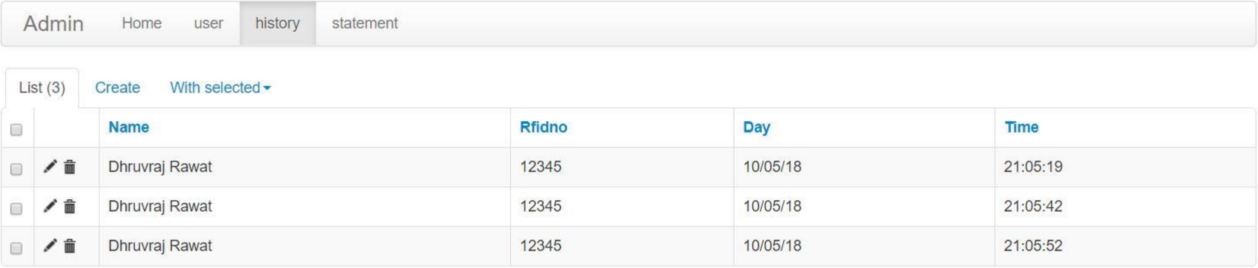


Figure 8: Lateral View of the Proposed System



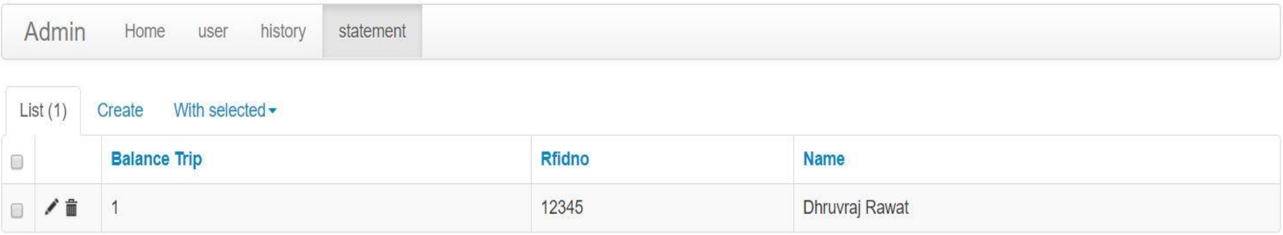
□		Pin	Rollno	Name	Email	Typecard	Rfidno
□	✎ 🗑	123	15uec022	Dhruvraj Rawat	15UEC022@LNMIIT.AC.IN	1	12345

Figure 9: In admin panel showing User-Table



□		Name	Rfidno	Day	Time
□	✎ 🗑	Dhruvraj Rawat	12345	10/05/18	21:05:19
□	✎ 🗑	Dhruvraj Rawat	12345	10/05/18	21:05:42
□	✎ 🗑	Dhruvraj Rawat	12345	10/05/18	21:05:52

Figure 10: In admin panel showing History Table



□		Balance Trip	Rfidno	Name
□	✎ 🗑	1	12345	Dhruvraj Rawat

Figure 11: In admin panel showing Statement Table

Conclusion

An equipment-based access management system based on RFID technology is proposed in this project. This kind of RFID technology is sweeping the globe. In this project raspberry pi based base station powered by EM-18 (RFID Reader) and a Flask based remote project is proposed. The Data of the entire users are saved securely on the SQL base database with only admin having access to it. The mailing functionality is also implemented to notify user when they use their pass showing how much trip are remaining and at what data and time it was used.

Future Work

A custom hardware-based implementation of the circuit can be done, by first prototyping it in FPGA then making an ASIC implementation. The server-side code can also be improved to detect the fraud by using a Machine learning model. Future, more the RFID reading range can be increased by using the powerful reader by supply power via external power source.

Bibliography

- **An intelligent management system based on RFID technology**
  https://ieeexplore.ieee.org/document/6011807/

- **Interfacing EM-18 *RFID* reader with Raspberry Pi**
  https://electrosome.com/em-18-rfid-reader-raspberry-pi/

- **Smart traffic management system using Internet of Things**
  https://ieeexplore.ieee.org/document/8323770/

- **Flask framework Documentation**
  http://flask.pocoo.org/docs/1.0/

- **SQLAlchemy Documentation for Database Creation**
  http://docs.sqlalchemy.org/en/latest/