

Summer Project Report

On

Fingerprint Based Attendance System



Advisor

Prof. Amey Karkare

Submitted by-

Rajat Chaudhary

Priyaranjan

Deepak Kumar

Durgesh Deep

(Group 02)

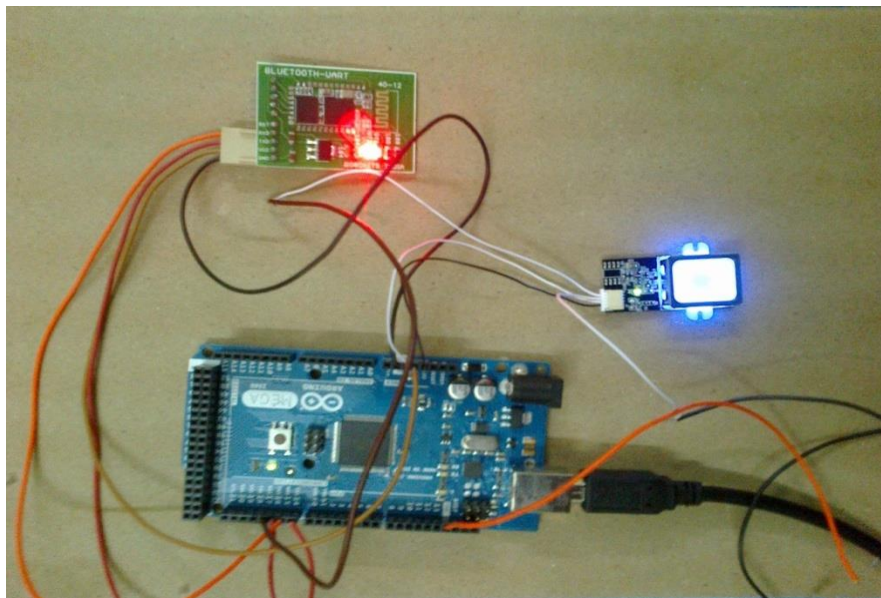
Introduction

The most common means of tracking student attendance in the classroom is by asking the students to manually sign the attendance sheet, which is normally passed around the classroom while the professor is conducting the lecture. The previous approach in which manually taking and maintains the attendance records was very inconvenient task. After having these issues in mind, we developed a [wireless automatic attendance system](#) which automates the whole process of taking attendance and maintaining it.

Fingerprint identification is one of the most well known and common biometric identification system. Because of their uniqueness & consistency over time, fingerprints have been used for identification over many years, more recently becoming automated due to advancement in computing capabilities. So, here the fingerprint identification technique was used for maintaining the attendance record. The record of the fingerprints of various students was maintained in a database. The communication between the PC and Module was done wirelessly over Bluetooth.

- For controlling both these modules the microcontroller board, Arduino Mega 2560 was used.
- For Implementing GUI Python's Tkinter library was used.
- The database was maintained over MySQL.

Students are supposed to enroll their fingerprint at the beginning of the semester for a particular course. During the class the fingerprint module would be passed among the students to mark their attendance.



About Hardware

Fingerprint Module GT-511C3

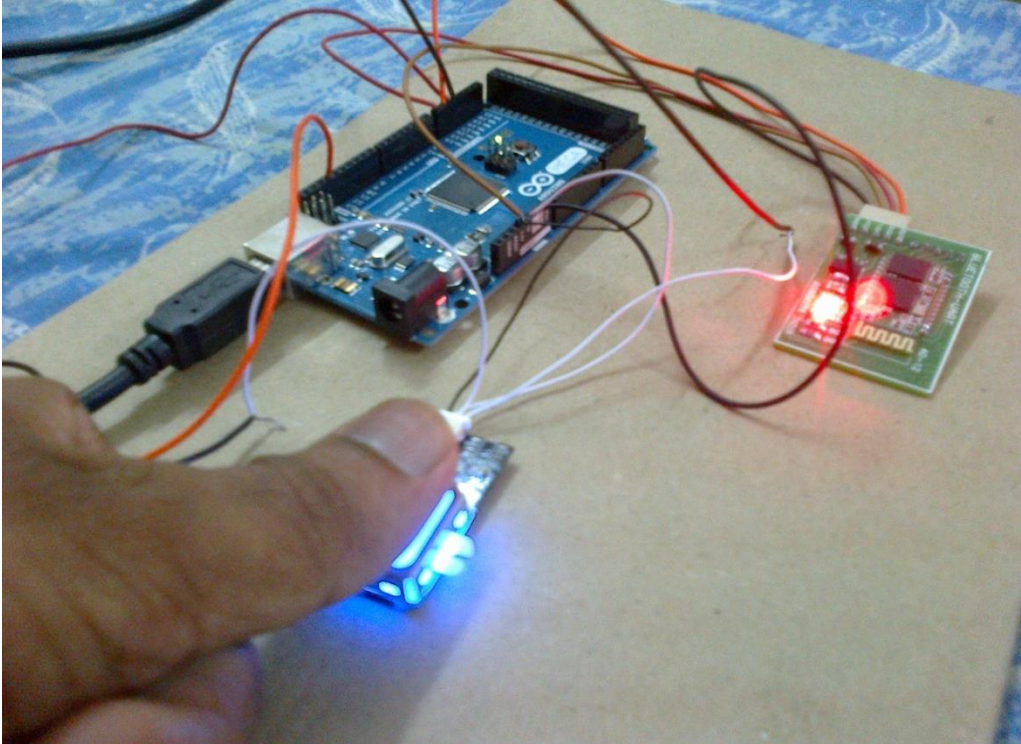
The module does all the heavy work of reading, identifying, and storing the fingerprint data. It can be issued several commands for all the functionalities. The module can store up to 200 different fingerprints and is capable of 360° recognition. For working the fingerprint must be registered by sending appropriate commands. On successful execution of the command it sends acknowledgement for success and Error code otherwise. The database of the prints can even be downloaded from the unit and distributed to other modules. The raw images of the fingerprints can also be retrieved from the module.

Enrolling: To enroll the fingerprint the finger is to be pressed to the module thrice. All the three times it creates a template for the finger that was put on the optical sensor, added to that, the third time it also merges the three templates to create the final template. On successful enrolling the device sends a unique ID pertaining to the finger enrolled. This ID can be saved and later used for verification of the finger.

- **Enrolling Procedure:**

1. EnrollStart(ID); // Issue command to start enrolling over the passed ID as parameter.
2. CaptureFinger ; // Take snapshot of the finger
3. Enroll1; // Create template of the 1st Image
4. Remove and press finger again
5. CaptureFinger;
6. Enroll2; // Create template of the 2nd Image
7. Remove and press finger again
8. CaptureFinger;
9. Enroll3 // Create template of the 3rd image and merge all 3 templates.

(In case of any error in the above procedure, the device sends an error code.)*



Enrolling a new Fingerprint

- **Verifying Procedure:**
 1. CaptureFinger
 2. Identify1_N
 3. If(ID < 200) → Verified ID
 4. Else Invalid Finger

NOTE:

- During enrolling, we have to press the fingerprint module thrice.
- During marking attendance, we have to press it only once.

COMMUNICATION PROTOCOL

Command Packet (Command)

OFFSET	ITEM	TYPE	DESCRIPTION
0	0x55	BYTE	Command start code1
1	0xAA	BYTE	Command start code2
2	Device ID	WORD	Device ID: default- 0x0001, fixed always
4	Parameter	DWORD	Input parameter
8	Command	WORD	Command Code
10	Check Sum	WORD	Check Sum (byte addition) OFFSET[0]+...+OFFSET[9] = Check Sum

Response Packet (Acknowledge)

OFFSET	ITEM	TYPE	DESCRIPTION
0	0x55	BYTE	Response start code1
1	0xAA	BYTE	Response start code2
2	Device ID	WORD	Device ID: default- 0x0001, fixed always
4	Parameter	DWORD	Response == 0x30 (ACK) Output Parameter Response == 0x31 (NACK) Error Code
8	Response	WORD	0x31 : Acknowledge (ACK) 0x31 : Non-Acknowledge(NACK)
10	Check Sum	WORD	Check Sum (byte addition) OFFSET[0]+...+OFFSET[9] = Check Sum

Data Packet (Data)

OFFSET	ITEM	TYPE	DESCRIPTION
0	0x5A	BYTE	Data start code1
1	0xA5	BYTE	Data start code2
2	Device ID	WORD	Device ID: default- 0x0001, fixed always
4	Data	N BYTES	N Bytes Data The size is pre-defined per protocol stage
4+N	Check Sum	WORD	Check Sum (byte addition) OFFSET[0]+...+OFFSET[4+N-1] = Check Sum

For Commands HEX code and Different Error Codes: Refer the DataSheet.

Connections: The fingerprint module has JST-SH connector with 4 pins. It needs a power supply of 3.3V-6V. But the Rx – Tx Communication voltage is 3.3V. So a potential divider is required to preventing the fingerprint module from the 5V Rx-Tx of the Arduino. The connections are as follows:

1. FPS Tx -----> Arduino Rx (Serial 2)
2. FPS Rx -----> Arduino Tx(Stepped down to 3.3V using potential divider of 560 and 1000 ohm resistors)
3. FPS Gnd ----->Gnd
4. FPS VCC -----> 3.3V

The serial communication is set at 9600 bps on both the devices(Arduino and Fingerprint module) for correct synchronization. The fingerprint module has a default baudrate of 9600 bps.

Various Commands that were useful and used in the code are as follows:

1. Open
2. Close
3. CmosLed
4. GetEnrollCount
5. CheckEnrolled
6. EnrollStart
7. Enroll1, Enroll2, Enroll3
8. IsPressFinger
9. DeleteID
10. DeleteAll
11. Identify
12. CaptureFinger

Arduino Mega 2560

Arduino microcontroller acts as the link between the fingerprint module and the Bluetooth module. It converts the data received from the FingerPrintSensor(FPS) to a string that can be sent over the Bluetooth. It also parses the data received from the PC and sends appropriate commands to the FPS. It was used since it has multiple serial ports available on the board. This makes it easy to communicate with both Bluetooth module and FPS.

Connections:

1. Serial Port 2 --→ Fingerprint Sensor
2. Serial Port 3 --→ Bluetooth Module
3. Serial Port 1 (Optional) --→ PC //Only for verifying if circuit is working

PC sends a 4 digit number as a command and parameter. The thousands digit is for command as described in the code. The rest of the 3 digits are used for passing the parameter. These 3 digits are the ID which we want to delete using the delete command.

Bluetooth Module

It supports simple serial communication. It works at a baud rate of 9600 bps and hence is always in sync with the PC and the Arduino board. Its PIN is 0000. It was first paired with the PC. After that the connection with the Bluetooth module can be done with the serial library of Python. There is a difference in the connection here. Manufacturing fault:

Arduino Tx -→ Module Tx

Arduino Rx → Module Rx

Power: 5V DC

About Software

Software tools used:

- MySQL(For Database Management)
- Python's GUI (Graphical User Interface) package- TkINTER

Why MySQL:

- ***It's easy to use:*** While a basic knowledge of SQL is required—and most relational databases require the same knowledge—MySQL is very easy to use. With only a few simple SQL statements, you can build and interact with MySQL.
- ***It's secure:*** MySQL includes solid data security layers that protect sensitive data from intruders. Rights can be set to allow some or all privileges to individuals. Passwords are encrypted.
- ***It's scalable:*** MySQL can handle almost any amount of data, up to as much as 50 million rows or more. The default file size limit is about 4 GB. However, you can increase this number to a theoretical limit of 8 TB of data.
- ***It runs on many operating systems:*** MySQL runs on many operating systems, including Novell NetWare, Windows, Linux, many varieties of UNIX, FreeBSD and others.

Why TkINTER:

- As the 'standard' toolkit it is included in the base Python distribution and as such Tkinter code should run on any system with Python installed.
- Since the Tk toolkit has been around for a very long time (since 1991), it is more portable than any other GUI toolkit.
- Many of the alternatives available don't even support all of X11, Windows, and Macintosh, let alone other, more obscure window systems.

List of all Databases used and their Tables :

<u>Databases</u>	<u>Tables</u>
➤ python	<ul style="list-style-type: none"> • LOGIN
➤ COURSES	<ul style="list-style-type: none"> • name_of_courses
<ul style="list-style-type: none"> ➤ Course_no_1 	<ul style="list-style-type: none"> • STUDENTS_LIST • DATE_AND_DAY • Roll_no_1 • • • Roll_no_n • D1 • • • Dn
<ul style="list-style-type: none"> • Course_no_2 	<ul style="list-style-type: none"> • STUDENTS_LIST • DATE_AND_DAY • Roll_no_1 • • • Roll_no_n • D1 • • • Dn
.....

ABOUT DATABASES:

1. python :

It contains a table viz. "LOGIN" which stores username and password. Whenever user changes password or username, entries of table LOGIN are updated automatically through our python script.

2. COURSES :

It contains the table "name_of_courses" which keeps the record of all the courses. The table has two columns, Course_no and Course_name of all the courses in progress.

3. Course No Databases :

This database is created each time a new course is registered using the interface. The name of the database is same as its course number(ex: CS220). This database contains following tables :

```
+-----+
| Tables_in_cur102 |
+-----+
| D1                |
| D2                |
| D3                |
| D4                |
| DATE_AND_DAY     |
| STUDENTS_LIST    |
| Y12228           |
| Y12250           |
| Y12519           |
| Y12543           |
+-----+
```

- a.) **STUDENTS_LIST:** This table contains the complete information about students who are registered in that particular course. It contains their Name, Roll no, Section, Department and their unique Finger Id.

```
mysql> select * from STUDENTS_LIST;
+-----+-----+-----+-----+-----+
| STUDENT          | ROLL  | SECTION | DEPARTMENT | FINGER_ID |
+-----+-----+-----+-----+-----+
| RAJAT_CHAUDHARY | 12543 | B11     | CSE        | 1         |
| PRIYARANJAN     | 12519 | B11     | CSE        | 2         |
| DURGESH_DEEP    | 12250 | B12     | CSE        | 3         |
| DEEPAK_KUMAR    | 12228 | B9      | CSE        | 4         |
+-----+-----+-----+-----+-----+
```

- b.) **DATE_AND_DAY:** It has two columns which keeps record of all the dates on which a particular class was held along with its day number. It is updated each time a new class is taken.

```
mysql> select * from DATE_AND_DAY;
+-----+-----+
| DATES      | DAY  |
+-----+-----+
| 2012-12-01 | 1    |
| 2012-12-02 | 2    |
| 1245-12-09 | 3    |
| 2014-07-02 | 4    |
+-----+-----+
```

- c.) **Roll no tables:** There are n tables where n is the number of students registered in that particular course. These tables contain the information about the dates on which the class was held and their attendance on that particular day.

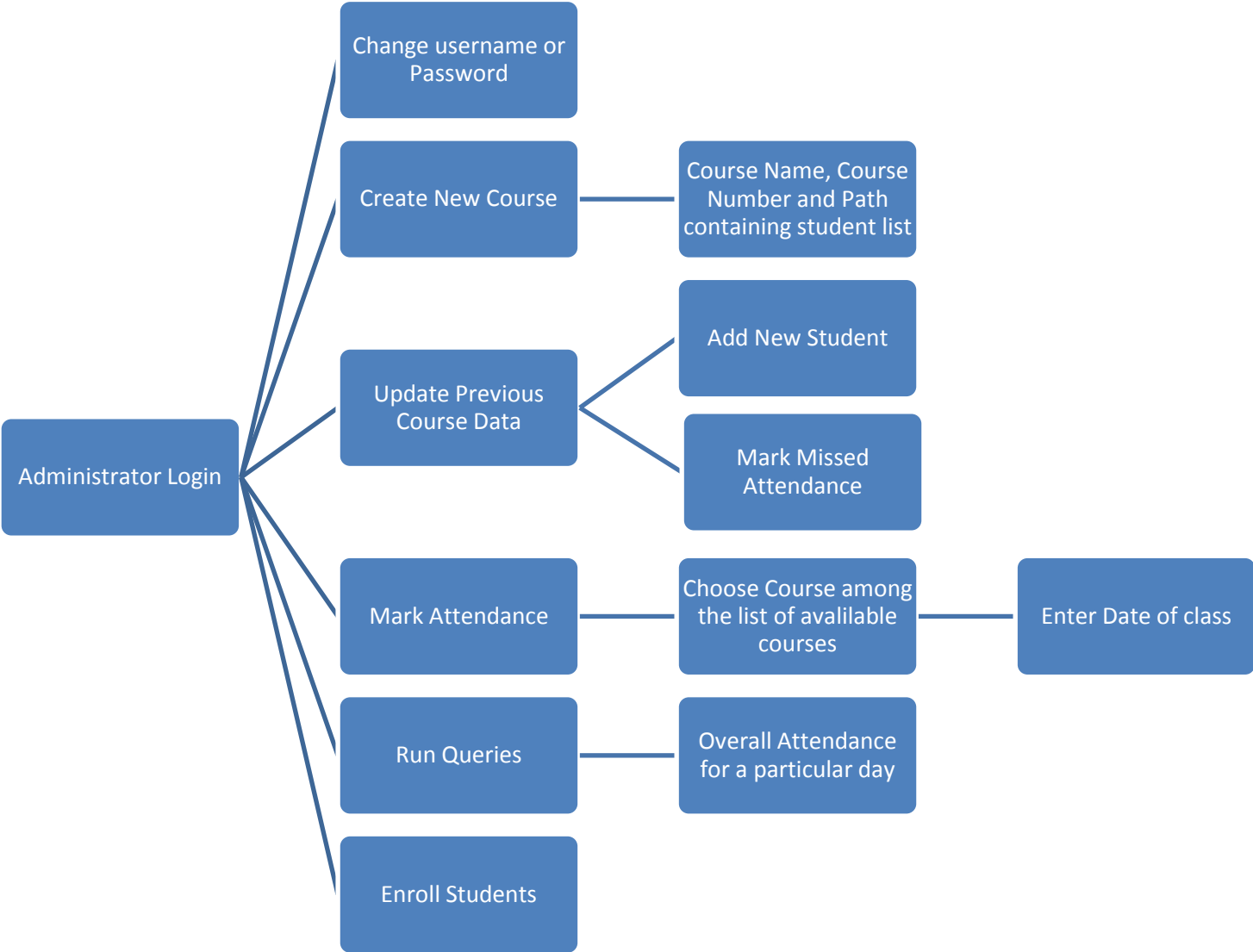
```
mysql> select * from Y12519;
+-----+-----+-----+
| DATES      | DAY  | PRESENTorABSENT |
+-----+-----+-----+
| 2012-12-01 | D1   | 1                |
| 2012-12-02 | D2   | 1                |
| 1245-12-09 | D3   | 0                |
| 2014-07-02 | D4   | 0                |
+-----+-----+-----+
```

- d.) **Day Tables:** There are n tables where n is the number of days on which classes were held. These tables contain the information about the students with their roll no and attendance*.

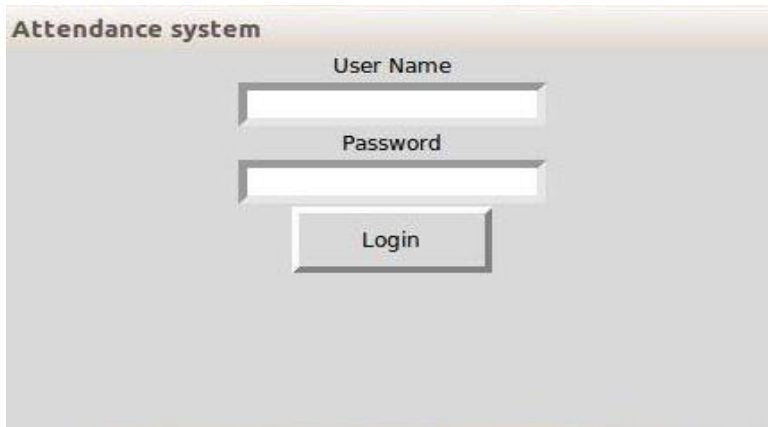
```
mysql> select * from D1;
+-----+-----+
| ROLL_NO | PRESENTorABSENT |
+-----+-----+
| 12543   | 0                |
| 12519   | 1                |
| 12250   | 0                |
| 12228   | 0                |
+-----+-----+
```

(*If student is present then attendance is marked as 1 otherwise 0.)

Flow Chart for the entire procedure



Some Screenshots



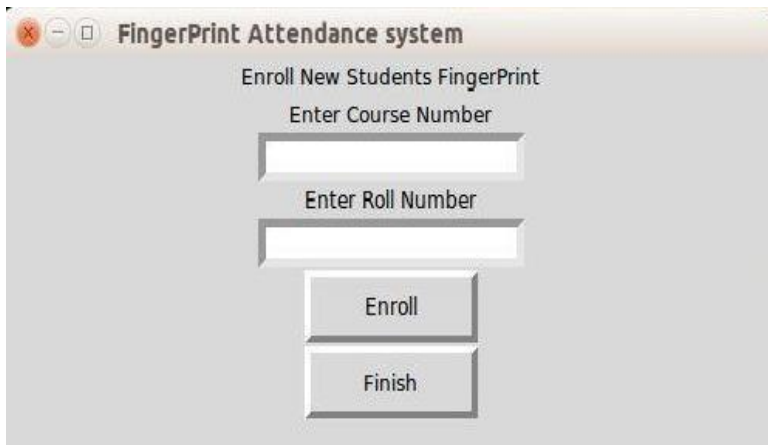
Attendance system

User Name

Password

Login

Administrator Login



FingerPrint Attendance system

Enroll New Students FingerPrint

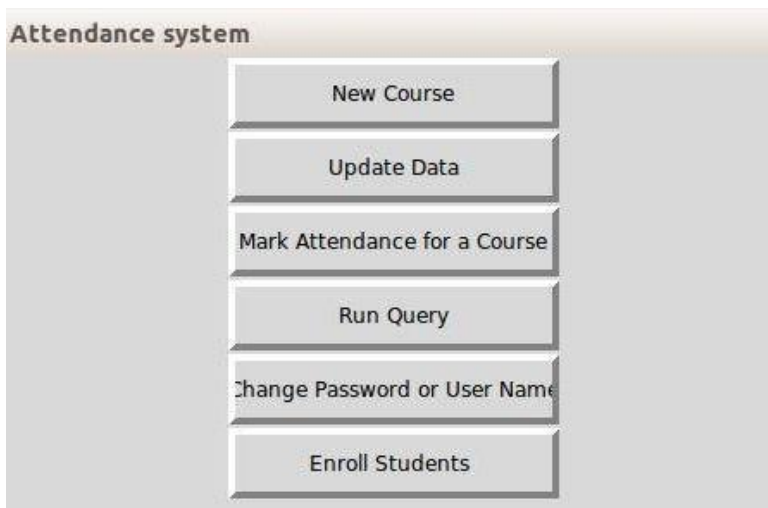
Enter Course Number

Enter Roll Number

Enroll

Finish

For Enrolling the Students



Attendance system

New Course

Update Data

Mark Attendance for a Course

Run Query

Change Password or User Name

Enroll Students

Main Menu after Login

Conclusion

This Wireless fingerprint attendance system is elegant and efficient way to track the presence of students in the class over an entire semester for various courses. It also gives easy interface to get detailed information of relevant queries. Using this attendance system, Professor can get the attendance of a particular student throughout whole semester, attendance of whole class for a particular day and attendance of whole class throughout the semester in a tabular form within few seconds .

Future Work & Expectations

- To make GUI more user friendly, so that it saves Professors' time in extracting data from updated databases.
- To use another LED in Arduino Mega which will indicate that the attendance has been marked in the local database.
- This attendance system can be extended for marking attendance from 200 students to 400 students or more and can be extended for using in large lecture hall.
- A small LCD screen can be attached which will show short messages like “put your finger”, “Your attendance has been marked” etc.
- The template data obtained from the Fingerprint module can be streamed to the PC for local record maintainance.

References

- [1] Website : http://www.tutorialspoint.com/python/python_gui_programming.htm
- [2] Website : <http://stackoverflow.com>
- [3] Website : <https://www.sparkfun.com/products/11792>
- [4] Website : <http://arduino.cc>
- [5] MySQL (4th edition), by *Paul DuBois*