



معاونت پژوهشی و فناوری

## طراحی و پیاده سازی سامانه احراز هویت با رزبری و Rfid

شماره طرح:

مجری: اسما محمدی

(96405096)

ماه بهمن سال 1400



معاونت پژوهشی و فناوری

## **Raspberry Pi RFID Attendance System**

Project Number: ....

Project Manager:

Asma Mohammadi

February of 2022

عنوان: طراحی و پیاده سازی سیستم احراز هویت با رزبری و Rfid

دانشگاه: شهید باهنر کرمان

مجری: اسما محمدی (96405069)

تاریخ شروع: بهمن 1399

تاریخ پایان: بهمن 1400

## سیستم احراز هویت با رزبری:

دفاتر ثبت حضور و غیاب در اداره ها و شرکت ها، علاوه بر اینکه نگه داری داده ها فضا فیزیکی را اشغال میکند، برای رسیدگی به داده ها و یا پردازش داده ها، نیاز به دقت و صرف زمان توسط یک فرد است.

این سامانه دیجیتال علاوه بر حذف دفاتر ضخیم، پردازش اطلاعات را سریعتر میکند و از کارهای تکراری مثل نوشتن دائم اسم افراد یا تاریخ ها جلوگیری میکند.

این سیستم با برد رزبری و ماژول Rfid کارت ها را شناسایی میکند و داده ها را در mysql ثبت و ذخیره میکند.

استفاده از رزبری در این سامانه، به ما قدرت اضافه کردن و تکمیل سامانه با توجه به نیاز های جدید را میدهد.

داده ها در جداول ذخیره میشوند و در صورت پردازش میتوانند به آسانی مورد استفاده قرار گیرند.

## Contents

3	شناسه طرح
4	چکیده
6	مقدمه
7	وسایل موردنیاز
8	راه اندازی رزبری
11	راه اندازی نمایشگر
15	راه اندازی Rfid
18	اتصال پایگاه داده
23	اتصال پایگاه داده
26	ثبت حضور
28	جداول پایگاه داده
30	پیشنهادهات
31	منابع



سامانه احراز هویت با کارت، یک سامانه دیجیتال و کاربردی است که جایگزین دفاتر بزرگ و قطور "ثبت ورود و خروج" افراد است که هم کاغذ و قلم زیادی را هدر میدهد و هم رسیدگی به اطلاعات به صورت دستی، زمان بیشتری را میگیرد.

دهه هاست که با ورود ابزار دیجیتال، پایگاه داده های دیجیتال جایگزین پایگاه داده های فیزیکی جاگیر را گرفته است. علاوه بر این، خارج کردن اطلاعات مفید از تاریخچه ها و داده ها را در صورت نیاز آسان تر و سریع تر کرده است.

سامانه به صورتی ست که هرکسی داری یک کارت با شماره مشخصی ست که با آن ورود پیدا میکند و زمان و تاریخ آن، در جداول پایگاه داده ثبت می شود.

## وسایل مورد نیاز

- 1- رزبری wh : مدلی از رزبری که ارزان تر است ولی با قدرت پردازشی کمتر. مثلاً برای پردازش تصویر که به قدرت بالایی نیاز دارد مناسب نیست.
  - 2- RFID-RC522: که شامل کارت و کارت خوان است.
  - 3- صفحه نمایشگر 2\*16
  - 4- سیم های جامپر
  - 5- پتانسیومتر (10k)
  - 6- کارت SD
  - 7- نمایشگر 5 اینچ
  - 8- کابل HDMI
  - 9- مبدل mini HDMI به HDMI
  - 10- مبدل usb به mini usb
- توضیح: دلیل استفاده از مبدل ها، کوچکی رابط های رزبری wh است





ابتدا فایل iso سیستم عامل Raspbian را از سایت رسمی رزبری پای دانلود می کنیم سپس آن را از حالت فشرده خارج می کنیم

در مرحله ی بعدی SD card را فرمت می کنیم و فایل سیستم عامل Raspbian پسوند img. را بر روی آن رایت می کنیم.

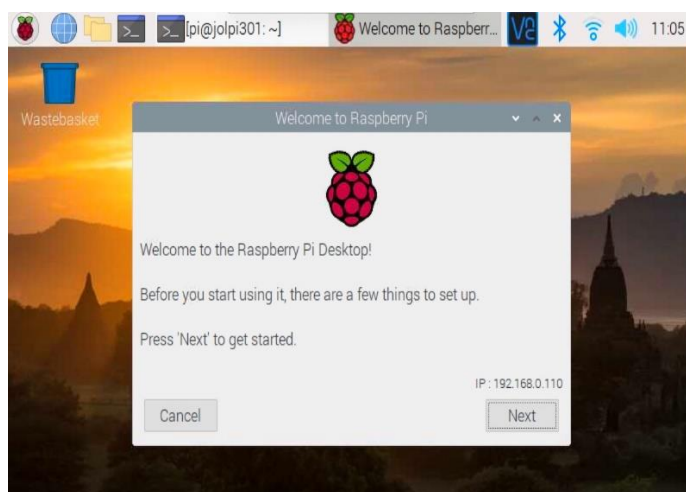
کابل HDMI را از رزبری پای به نمایشگر 5 اینچی به صورت زیر متصل می کنیم.

برای اتصال پین های رزبری به نمایشگر باید پین هدر های رزبری را به پین های مادگی نمایشگر به صورت که پین های 5 ولت آن درون هم قرار میگیرند وصل کنیم تا ارتباط برقرار شود. و بعد اتصال موس به رزبری با استفاده از یک مبدل. (موس بلوتوثی ست که فاقد سیم است)



و در آخر یک عدد آداپتور 5 ولت با حداقل جریان خروجی 700 میلی آمپر به رزبری پای متصل می کنیم. اکنون چراغ پاور بر روی رزبری روشن میشود و رزبری شروع به بوت شدن میکند

اولین بوت مقداری زمان می برد چون زمان بیشتری برای ساخت دایرکتوری ها برای اولین بار نیاز دارد تا سیستم عامل کاملا آماده شود. بعد از ورود، به یک وای فای متصل می کنیم تا بعدا بتوانیم از طریق ssh به آن متصل شویم.



## اتصال به رزبری پای از طریق SSH و VNC:

برای کار با رزبری میتوانیم بجای استفاده از صفحه نمایشگر، از طریق ارتباط SSH آن را روی لپ تاب بالا بیاوریم.

نرم افزارهای مورد نیاز:

1- IP Scanner

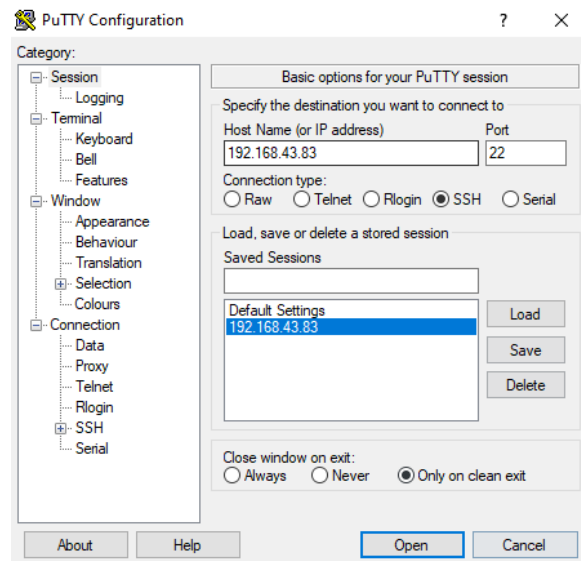
2- Putty

3- Vnc viewer

بعد از آماده سازی کارت حافظه SD که شامل سیستم عامل Raspbian است با استفاده از یک Ram Reader ، وارد کارت حافظه متصل به رزبری می شویم و یک فایل به نام SSH که باید پسوند txt را از انتهای آن حذف کنیم، می سازیم. کارت حافظه را به رزبری متصل می کنیم و به برق متصل میکنیم. صبر می کنیم تا سیستم عامل رزبری بالا بیاید.

لپ تاپ خود را از طریق وای فای به مودمی که رزبری را به آن وصل کردیم ، وصل می کنیم و با استفاده از نرم افزار IP Scanner آبیی آدرس دستگاه های متصل به مودمی که لپ تاب متصل است را می یابیم.

برای ادامه کار باید نرم افزار Putty را اجرا میکنیم. داخل برنامه Putty در قسمت آدرس، آدرس رزبری که در قسمت قبل پیدا کردیم را وارد می کنیم. در قسمت Connection Type ارتباط SSH را انتخاب میکنیم و open را میزنیم.



در ادامه پنجره سیاه باز میشود که باید نام کاربری و رمز عبور رزبری را وارد کنیم. در ادامه در همان پنجره، متن vncserver را وارد کنیم.

هنگامی که Enter را فشار بدهیم در همان پنجره سیاه رنگ یک آدرسی به ما می دهد که این همان آدرس و پورت ارتباطی با رزبری است.

در مرحله بعدی باید نرم افزار VNC Veiwer را اجرا کنیم و آدرس و پورت ارتباطی را در آن وارد کنیم.

و در انتها پنجره ای باز میشود که باید نام کاربری و رمز عبور رزبری پای را وارد کنیم.

پس از وارد کردن نام کاربری و رمز عبور، رزبری پای روی صفحه نمایش بالا می آید.

### بسته های مورد نیاز:

build-essential, git,python3-dev, python3-pip, python3-smbus

برای نصبشان، ابتدا ترمینال را باز میکنیم و دستور های زیر را وارد تا رزبری را به روز سازی کنیم:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

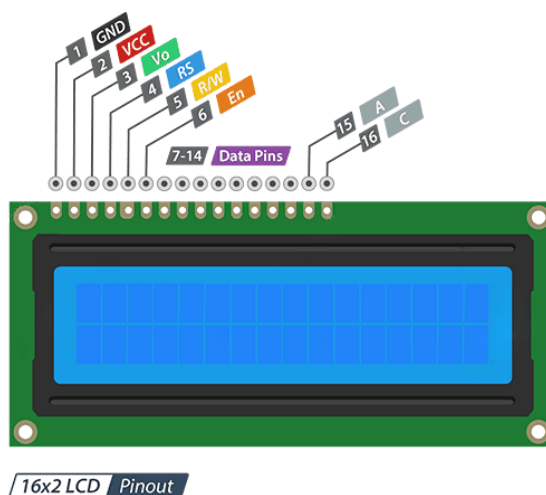
و بعد کد زیر برای نصب بسته ها در ترمینال وارد میکنیم:

```
sudo apt-get install build-essential git python3-dev python3-pip python3-smbus
```

```
sudo pip3 install RPi.GPIO
```

## اتصال سخت افزار:

این نمایشگر شامل 16 محل اتصال به صورت روبه روست:



برای کنترل نمایشگر از کتابخانه Adafruit استفاده میکنیم.

توضیح پین ها به اختصار:

Vo: پین سوم جهت کنترل کنتراست نور نمایشگر است که با اتصال به یک پتانسیومتر قابل کنترل است.

RS: تمایز بین فرمان و داده. 0 = فرمان و 1 = داده.

R/W: تمایز بین نوشتن و خواندن. چون در این پروژه به خواندن نیازی نداریم آن را به 1 وصل میکنیم و همیشه در حالت "نوشتن" باشد.

E: فعالسازی نمایشگر.

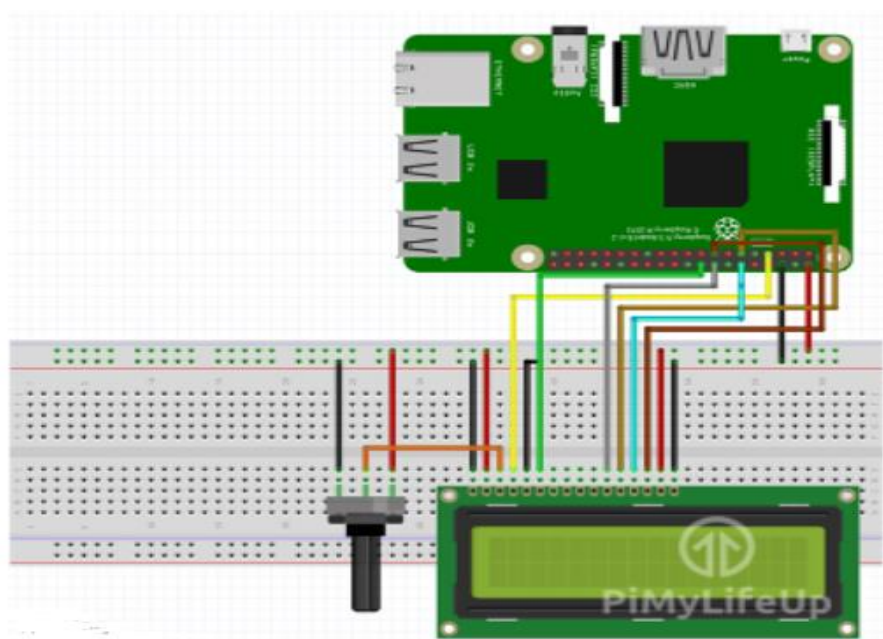
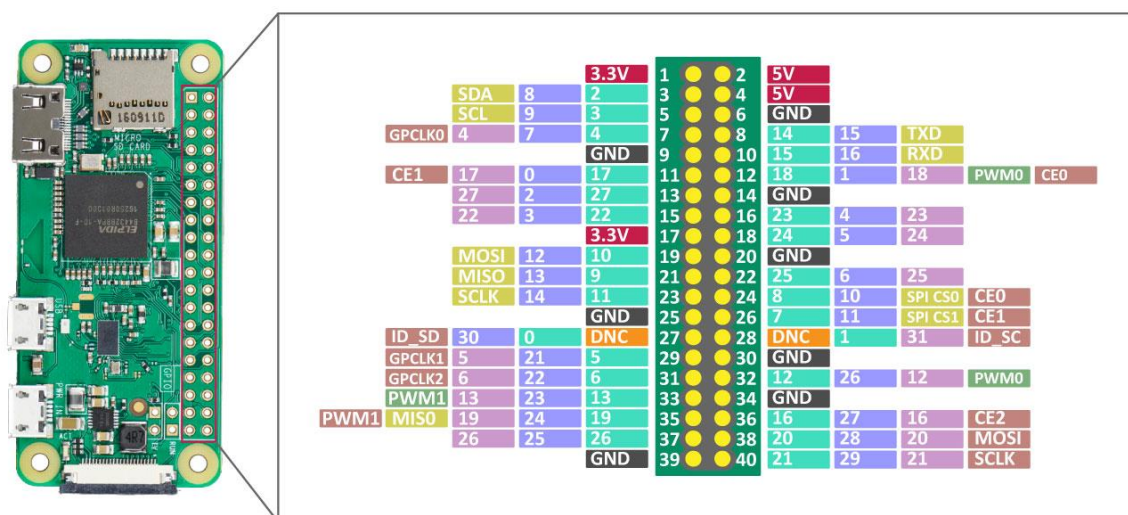
D0-D7: پین های داده. که در اینجا برای صرفه جویی در پین ها فقط پین D4-D7 استفاده شده و از روش 4تایی (در مقابل 8تایی) برای ارسال داده استفاده کرده ایم.

A-K: استفاده از این پین ها ضروری نیست و برای روشن کردن نور پس زمینه استفاده میشود.

## نحوه اتصال پین ها:

توجه کنید که انتخاب پین ها در کد، به طور کلی به دو صورت است. شیوه BOURN که انتخاب شماره ها در کنار پین هاست و شیوه BCM که شماره های برجسب های کنار پین ها در شکل روبه رو استفاده میشود.

در این پروژه از شیوه BCM استفاده شده است.



## قسمت نرم افزار:

برای راه اندازی نمایشگر 2\*16 و تست آن، کتابخانه Adafruit را نصب میکنیم.  
به این صورت که آدرس گیت هاب زیر را clone میکنیم و بعد به درون پوشه میرویم و فایل setup.py را اجرا میکنیم.

```
git clone https://github.com/pimylifeup/Adafruit_Python_CharLCD.git
cd ./Adafruit_Python_CharLCD
sudo python3 setup.py install
```

## تست نمایشگر:

این فایل یک پوشه "example" دارد که دارای نمونه است برای تست کردن. با برنامه nano مثال زیر را باز میکنیم و نمایشگر را با آن تست میکنیم.

```
nano ~/Adafruit_Python_CharLCD/examples/char_lcd.py
```

کد روبه رو نیازی به تغییر ندارد مگر در قسمت پین ها:

```
#!/usr/bin/python
# Example using a character LCD connected to a Raspberry Pi or BeagleBone Black.
import time
import Adafruit_CharLCD as LCD
# Raspberry Pi pin configuration:
lcd_rs = 4 # Note this might need to be changed to 21 for older revision
lcd_en = 24
lcd_d4 = 23
lcd_d5 = 17
lcd_d6 = 18
lcd_d7 = 22
lcd_backlight = 4
```

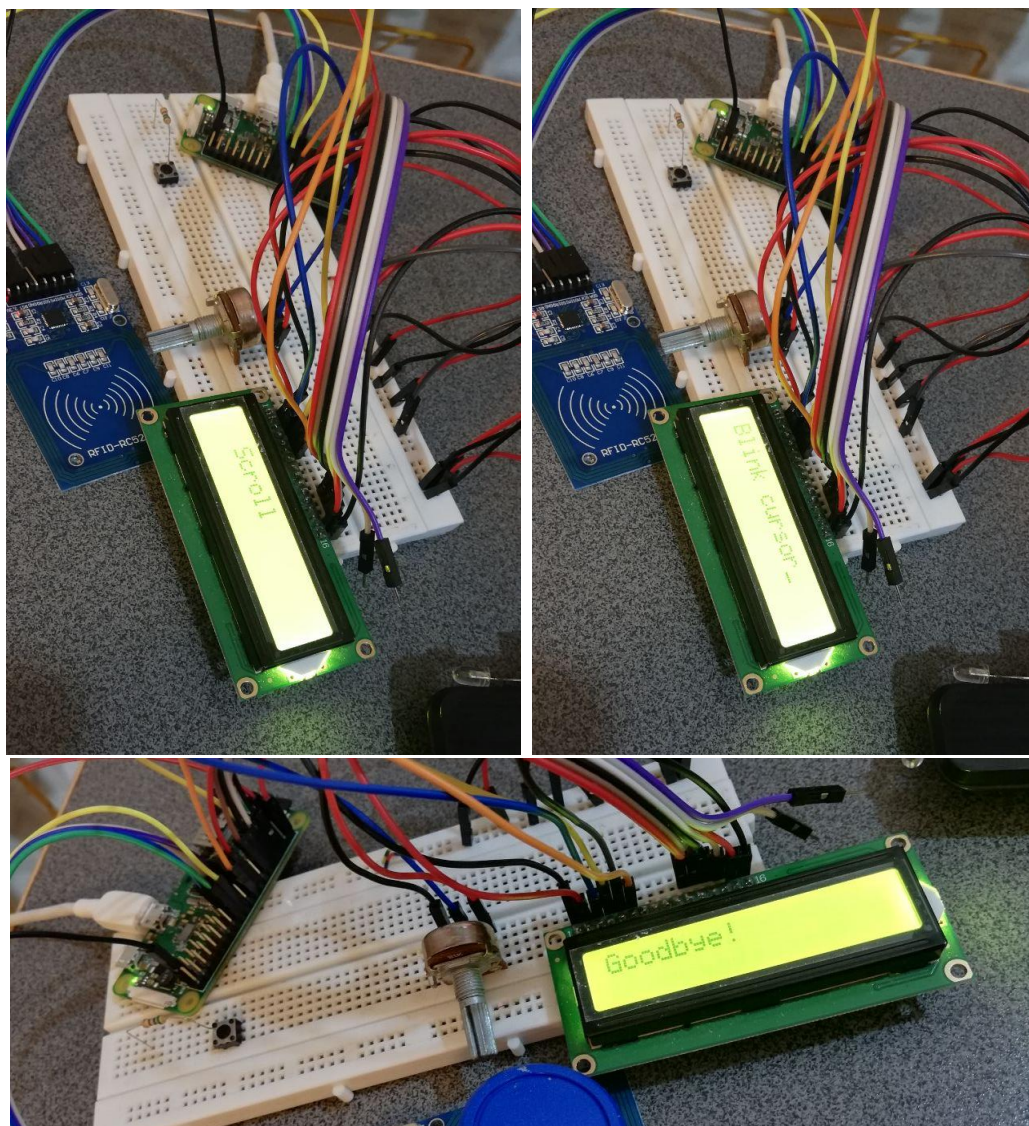


بعد با زدن Ctrl + X و Y و زدن Enter فایل را ذخیره میکنیم.

و اجرای کد:

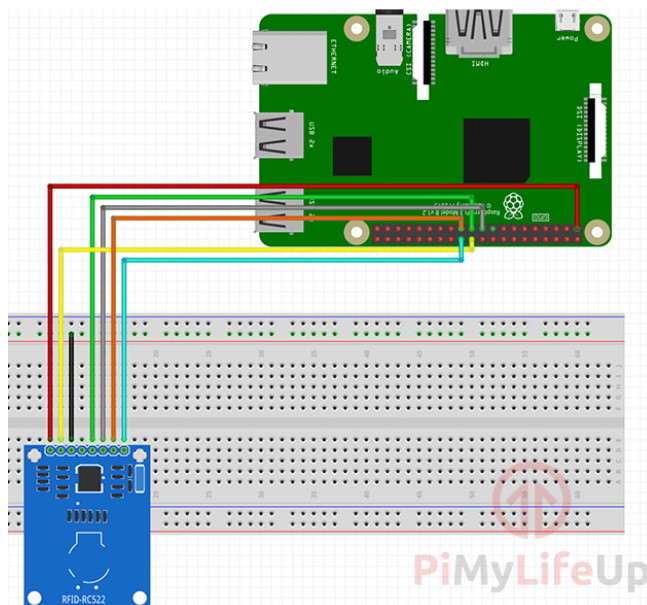
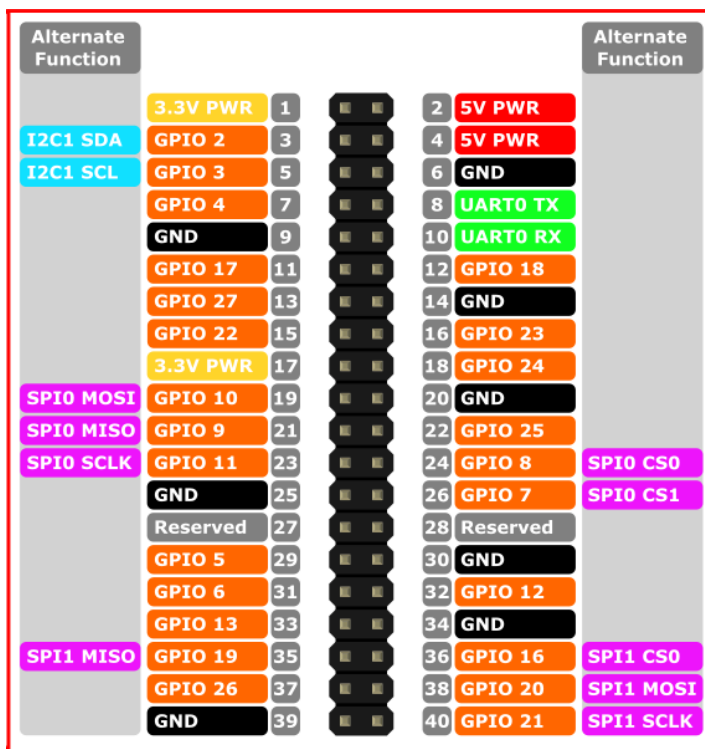
`python3 ~/Adafruit_Python_CharLCD/examples/char_lcd.py`

نمایش نوشته های زیر در LCD، نشان از درست بودن اتصال نمایشگر است:



Rfid مدل RC522 را تحت پروتکل SPI راه اندازی خواهیم کرد. بنابراین از پین هایی از رزبری که دارای این پروتکل هستند استفاده میکنیم.(نشانه گذاری با رنگ بنفش)

پایه Rfid	پایه رزبری
SDA	GPIO 8
SCK	GPIO 11
MOSI	GPIO 10
MISO	GPIO 9
RST	GPIO 25
GND	GND
3.3 v	3.3 v



مطابق جدول پایه ها را بهم وصل میکنیم.



## قسمت نرم افزار:

برای فعال کردن SPI دستور زیر را میزنیم تا به بخش پیکر بندی برود:

```
sudo raspi-config
```

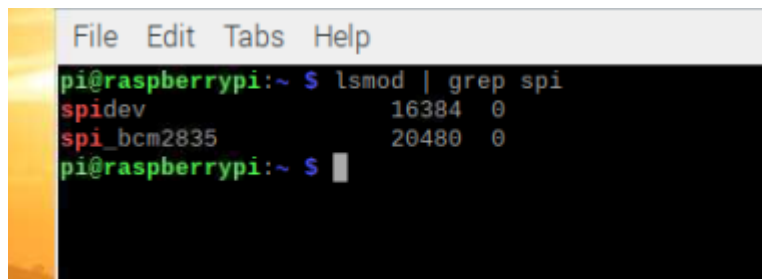
در قسمت Interfacing Options گزینه SPI را فعال میکنیم. و بعد رزبری را ری بوت میکنیم تا تغییرات اعمال شود.

برای بررسی و اطمینان از فعال شدن SPI دستور زیر را در ترمینال وارد میکنیم:

این دستور ماژول های کرنلی مربوط به spi که فعال شدند را نشان میدهد.

```
lsmod | grep spi
```

با دیدن عبارت "spi\_bcm2835" میتوانیم مدار را ببندیم.



```
File Edit Tabs Help
pi@raspberrypi:~ $ lsmod | grep spi
spidev                16384  0
spi_bcm2835           20480  0
pi@raspberrypi:~ $
```

به نصب دو کتابخانه دیگر هم نیاز داریم: spidev که واسط ارتباط با rfid reader (کارت خوان) را فراهم میکند و کتابخانه MFRC522 که کارهای اولیه و تکراری مربوط به rfid را انجام میدهد:

```
sudo pip3 install spidev
```

```
sudo pip3 install mfrc522
```

## تست Rfid:

برای بررسی درستی کارکرد Rfid، یک پوشه میسازیم و یک فایل جدید read.py، که با ویرایشگر nano آن را باز میکنیم:

```
mkdir ~/pi-rfid
```

```
nano ~/pi-rfid/read.py
```

و کد زیر را به آن اضافه میکنیم:

```
#!/usr/bin/env python

import RPi.GPIO as GPIO
from mfrc522 import SimpleMFRC522

reader = SimpleMFRC522() #make an object for rfid reader
try:
    id, text = reader.read()
    print(id)
    print(text)
finally:
    GPIO.cleanup() #clean gpio status for next time
```

آخرین کد مربوط به پاک کردن وضعیت پین هاست تا پین آزاد شوند در مرحله بعد یا استفاده بعدی بخاطر درگیر بودن پین ها به مشکل بر نخوریم.

با Ctrl + X و وارد کردن Y و بعد Enter فایل را ذخیره میکنیم. و حالا فایل را اجرا میکنیم:

```
python3 ~/pi-rfid/read.py
```

با زدن کارت، شناسه کارت را چاپ میکند که نشان از درست بودن قسمت Rfid دارد.

```
pi@raspberrypi:~ $ python3 ~/pi-rfid/read.py
932165081559

pi@raspberrypi:~ $
```

## اتصال پایگاه داده

دستور زیر را برای نصب پایگاه داده mySql در ترمینال وارد میکنیم:

```
sudo apt install mariadb-server
```

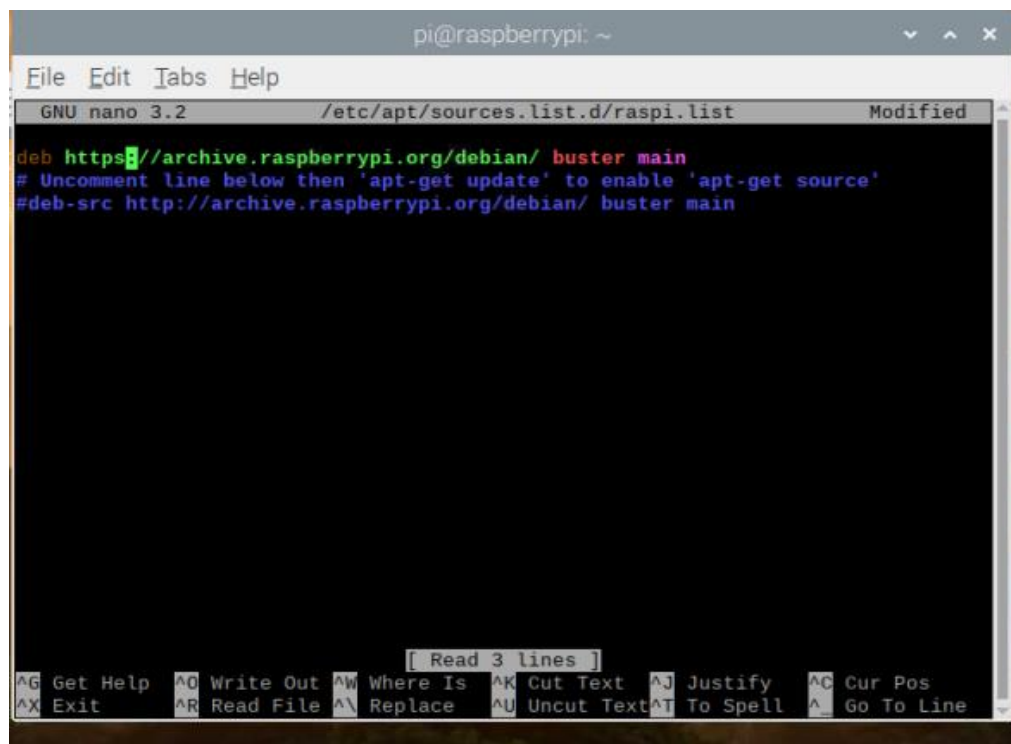
در اجرای دستور به مشکل "Failed to fetch ..." برخوردیم. این مشکل برای دانلود بعضی پکیج ها پیش می آید. مشکل از آدرسی ست که رزبری برای دانلود پکیج ها به آن رجوع میکند.

برای حل مشکل کد زیر را در ترمینال وارد میکنیم تا یک فایل باز شود:

```
sudo nano /etc/apt/sources.list.d/raspi.list
```

به قسمت http آدرس ارجاع روبه رو، یک s اضافه کردیم:

<http://archive.raspberrypi.org/debian/>



```
pi@raspberrypi: ~
File Edit Tabs Help
GNU nano 3.2 /etc/apt/sources.list.d/raspi.list Modified
deb https://archive.raspberrypi.org/debian/ buster main
# Uncomment line below then 'apt-get update' to enable 'apt-get source'
#deb-src http://archive.raspberrypi.org/debian/ buster main
[ Read 3 lines ]
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line
```

خارج میشویم. رزبری را ریboot میکنیم و در Ctrl+X فایل را ذخیره و با Enter و زدن Ctrl+O بعد با ترمینال عبارت زیر را وارد میکنیم تا تمامی پکیج را به روز کند:

```
sudo apt update
```

```
sudo apt full-upgrade
```

مشکل نصب mySql بعد از این سری حل شد.

نصب یک کتابخانه برای امنیت بیشتر پایگاه داده که جلوی دسترسی های ناامن را می گیرد:

```
sudo mysql_secure_installation
```

بعد از نصب، پیشنهاد تنظیم یک رمز برای پایگاه داده مان میدهد که با انتخاب Y و تعیین رمز ادامه میدهیم. در ادامه سوالات دیگری را میپرسد تا دسترسی ها را کم کند که همه را Y زدیم.

```
Remove anonymous users? [Y/n]
... Success!

Normally, root should only be allowed to connect from
localhost. This ensures that someone cannot guess at the root password
from the network.

Disallow root login remotely? [Y/n]
... Success!

By default, MariaDB comes with a database named 'test'
that has all permissions. This is intended only for testing, and should
be removed before moving into a production environment.

Remove test database and access to it? [Y/n]
- Dropping test database...
... Success!
- Removing privileges on test database...
... Success!

Reloading the privilege tables will ensure that all changes
will take effect immediately.

Reload privilege tables now? [Y/n]
... Success!

Cleaning up...

All done! If you've completed all of the above steps,
your installation should now be secure.

Thanks for using MariaDB!
pi@raspberrypi:~$
```

## ورود به پایگاه داده و ساخت پایگاه داده:

برای ورود به عنوان کاربر اصلی کد زیر را وارد میکنیم:

```
sudo mysql -u root -p
```

بعد یک پایگاه داده برای سامانه احراز هویتمان به نام attendancesystem میسازیم:

```
CREATE DATABASE attendancesystem;
```

و ساخت یک کاربر اصلی با دسترسی تمام و دادن یک رمز در قسمت password:

```
CREATE USER 'attendanceadmin'@'localhost' IDENTIFIED BY 'password';
```

```
GRANT ALL PRIVILEGES ON attendancesystem.* TO 'attendanceadmin'@'localhost';
```

و انتخاب پایگاه داده ای که ساختیم برای ساخت جداول:

```
use attendancesystem;
```

```
MariaDB [(none)]> CREATE DATABASE attendancesystem;
Query OK, 1 row affected (0.004 sec)

MariaDB [(none)]> CREATE USER 'attendanceadmin'@'localhost'
;
Query OK, 0 rows affected (0.028 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON attendancesystem
'@'localhost';
Query OK, 0 rows affected (0.007 sec)

MariaDB [(none)]> use attendancesystem;
Database changed
MariaDB [attendancesystem]> create table attendance(
-> id INT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE,
-> user_id INT UNSIGNED NOT NULL,
-> clock_in TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
-> PRIMARY KEY ( id )
-> );
Query OK, 0 rows affected (0.131 sec)

MariaDB [attendancesystem]> create table users(
-> id INT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE,
```

دو جدول با فیلدهای زیر برای سامانه نیاز داریم:

جدول کاربران

شناسه	شناسه کارت	نام کاربر	زمان ساخت

جدول حضور

شناسه	شناسه کاربر	ساعت

برای ساخت جداول کد زیر را یک جا وارد میکنیم.

```
create table attendance(  
  id INT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE,  
  user_id INT UNSIGNED NOT NULL,  
  clock_in TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,  
  PRIMARY KEY ( id )  
);
```

توضیح کد بالا:

id اولین فیلد و کلید اصلی (primary key) این جدول است که دو شرط خالی نبودن (not null) و یکتا بودن (unique) را دارد. این فیلد به صورت خودکار و با اضافه شدن اعداد ساخته و اضافه میشود.

فیلد clock\_in هم به صورت خودکار زمان ساخته شدن یک داده را از رزبری میگیرد و به داده مربوط اختصاص میدهد.

```
create table users(  
  id INT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE,  
  rfid_uid VARCHAR(255) NOT NULL,  
  name VARCHAR(255) NOT NULL,  
  created TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,  
  PRIMARY KEY ( id )  
);
```

خروج از قسمت پایگاه داده با استفاده از عبارت exit:

```
MariaDB [(none)]> use attendancesystem;
Database changed
MariaDB [attendancesystem]> create table attendance(
-> id INT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE,
-> user_id INT UNSIGNED NOT NULL,
-> clock_in TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
-> PRIMARY KEY ( id )
-> );
Query OK, 0 rows affected (0.131 sec)

MariaDB [attendancesystem]> create table users(
-> id INT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE,
-> rfid_uid VARCHAR(255) NOT NULL,
-> name VARCHAR(255) NOT NULL,
-> created TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
-> PRIMARY KEY ( id )
-> );
Query OK, 0 rows affected (0.102 sec)

MariaDB [attendancesystem]> exit;
Bye
pi@raspberrypi:~ $
```

برای ارتباط با mySql در حین اجرا و ذخیره و بازیابی اطلاعات در کد، نیاز به نصب کتابخانه mysql connector داریم:

```
sudo pip3 install mysql-connector-python
```

### کد اصلی پروژه ذخیره کارت ها:

حالا یک فایل جدید به اسم save\_user میسازیم که آن را با nano باز میکنیم:

```
nano ~/attendancesystem/save_user.py
```

کد اصلی و توضیحاتی که به صورت کامنت جلو بعضی کد ها نوشته شده:

```
1  #!/usr/bin/env python #os knows that this file should execute using python
2  import time
3  import RPi.GPIO as GPIO #free pins end of the script for next time
4  from mfrc522 import SimpleMFRC522
5  import mysql.connector #to talk with DB
6  import Adafruit_CharLCD as LCD #talk to LCD
7
8  #make connection to mySql server with required info
9  #db is an object created by the connector.
10 db = mysql.connector.connect(
11     host="localhost",
12     user="attendanceadmin",
13     passwd="password",
14     database="attendancesystem"
15 )
16
17 cursor = db.cursor() #cursor object from DB connection
18 reader = SimpleMFRC522() #ready to use rfid reader
19 #required pins for LCD
20 lcd = LCD.Adafruit_CharLCD(4, 24, 23, 18, 15, 14, 16, 2, 4)
21 #CD.Adafruit_CharLCD(lcd_rs, lcd_en, lcd_d4, lcd_d5, lcd_d6, lcd_d7,
22 # lcd_columns, lcd_rows, lcd_backlight)
23
24 try:
```



```

24 ~ try:
25 ~ while True:|
26     lcd.clear()
27     lcd.message('Place Card to\nregister')
28     id, text = reader.read() #wait for user card
29     #check to this id was unique
30     cursor.execute("SELECT id FROM users WHERE rfid_uid="+str(id))
31     cursor.fetchone() #grab the data
32
33 ~ if cursor.rowcount >= 1: #check last Sql call
34     lcd.clear()
35     lcd.message("Overwrite\nexisting user?")
36     overwrite = input("Overwrite (Y/N)? ")
37
38 ~ if overwrite[0] == 'Y' or overwrite[0] == 'y':
39     lcd.clear()
40     lcd.message("Overwriting user.")
41     time.sleep(1)
42     sql_insert = "UPDATE users SET name = %s WHERE rfid_uid=%s"
43 ~ else:
44     continue;
45 ~ else:
46     sql insert = "INSERT INTO users (name, rfid uid) VALUES (%s, %s)"
47     #ready insert comment to pass. %s means save
48     lcd.clear()
49     lcd.message('Enter new name')
50     new_name = input("Name: ")
51
52     cursor.execute(sql_insert, (new_name, id))
53
54     db.commit() #to occur our insert or update
55
56     lcd.clear()
57     lcd.message("User " + new_name + "\nSaved")
58     time.sleep(2)
59 finally:
60     GPIO.cleanup()
61

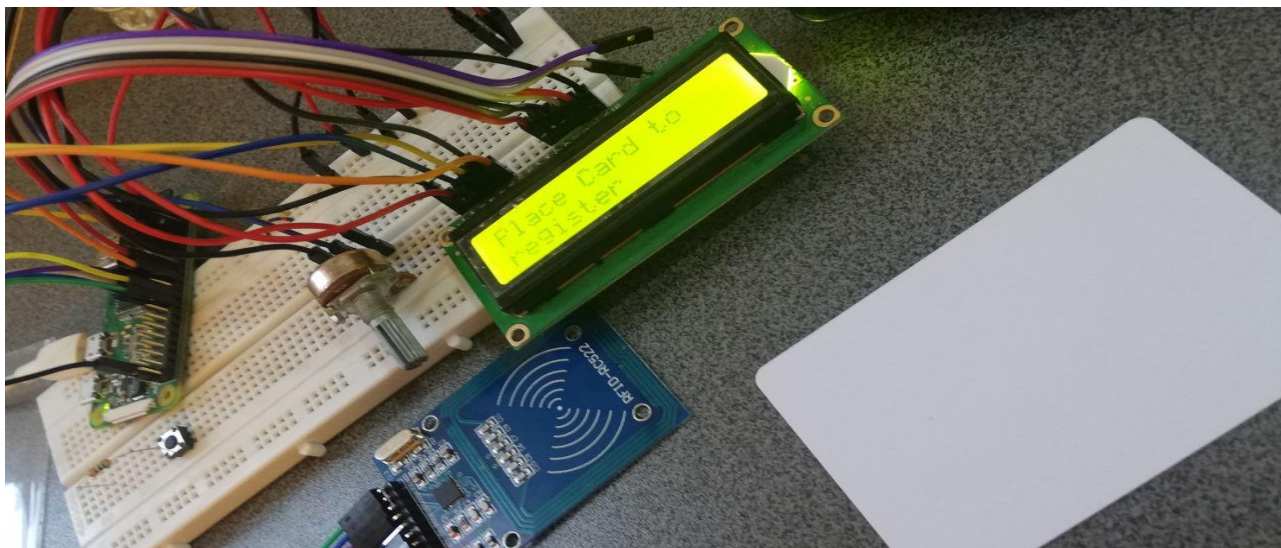
```

روی نمایشگر عبارت "کارت را برای ثبت نام قرار دهید" نوشته میشود و reader.read() منتظر قرار گرفتن کارت می ماند و بعد از قرار دادن کارت، با cursor.execute سعی میکند که شناسه خوانده شده را از پایگاه داده بازیابی میکند. Fetchone() یک خط از داده را برای مقایسه برمیدارد.

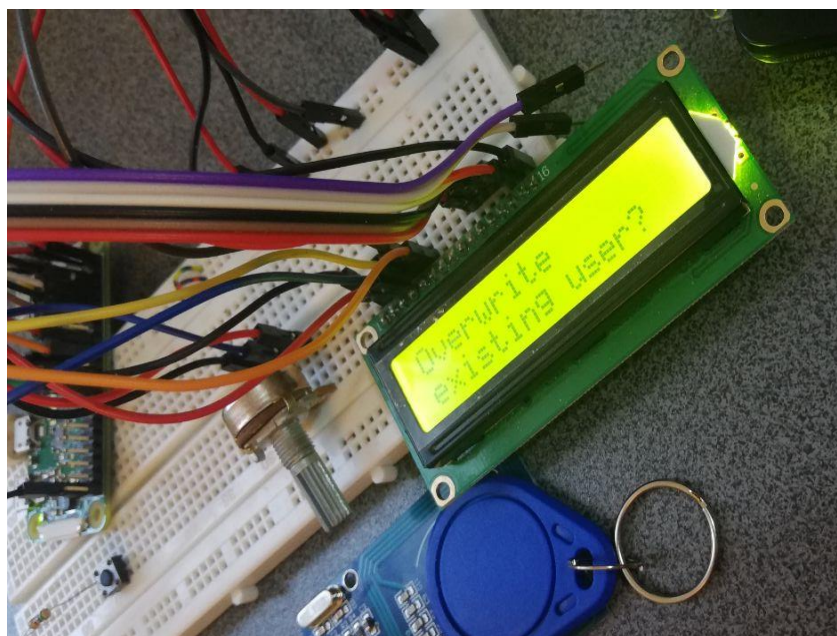
هر قسمت از کد که str() یا %s آمده است منظور فرمت رشته (string) است. دستور db.commit() را بعد از دستورات پایگاه داده ای زدیم تا تغییرات مربوط به پایگاه داده روی پایگاه داده اعمال شود.

اجرای کد:

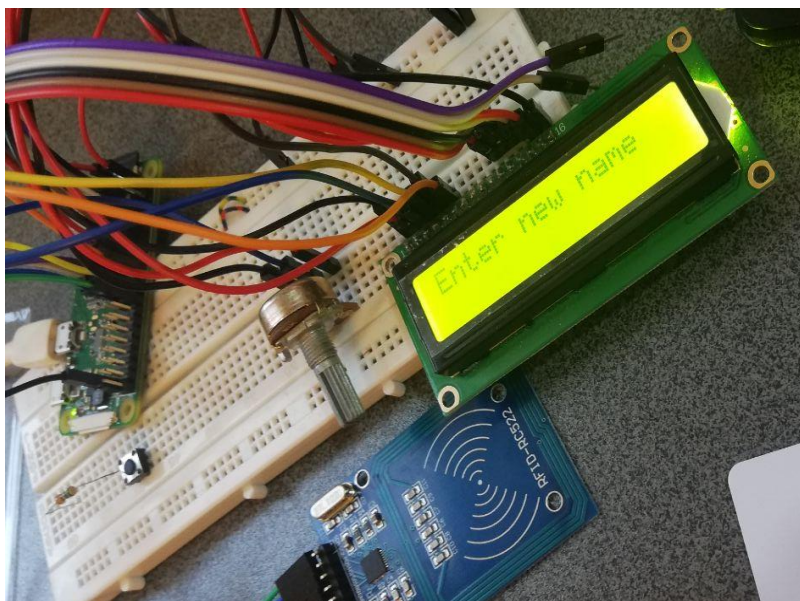
```
python3 ~/attendancesystem/save_user.py
```



بعد از خواندن کارت و جستجوی شناسه در پایگاه داده، اگر داده ای را برگرداند یعنی کاربر قبلاً ثبت شده و میپرسد که "آیا باز نویسی اش کنم؟" در صورت زدن  $\gamma$  باز نویسی میکند. بعد نام جدید را میپرسد تا به آن کارت اختصاص دهد.



اما اگر داده ای از پایگاه داده برگرداند یعنی کاربر جدید، که قبلاً ثبت نشده است. بعد از پرسیدن نام، آن کارت را به اسم فرد ثبت میکند.



```
[1]+ Stopped nano ~/attendancesystem/save_user
pi@raspberrypi:~ $ nano ~/attendancesystem/save_user.py
pi@raspberrypi:~ $ nano ~/attendancesystem/save_user.py
pi@raspberrypi:~ $ nano ~/attendancesystem/save_user.py
pi@raspberrypi:~ $ python3 ~/attendancesystem/save_user.py
Overwrite (Y/N)? Y
Name: asma
Name: sina
```

## ثبت حضور

## ثبت حضور:

یک فایل جدید به نام `check_attendance` ساخته و با ویرایشگر `nano` باز میکنیم:

```
nano ~/attendancesystem/check_attendance.py
```

و کد زیر را در آن قرار دادیم:

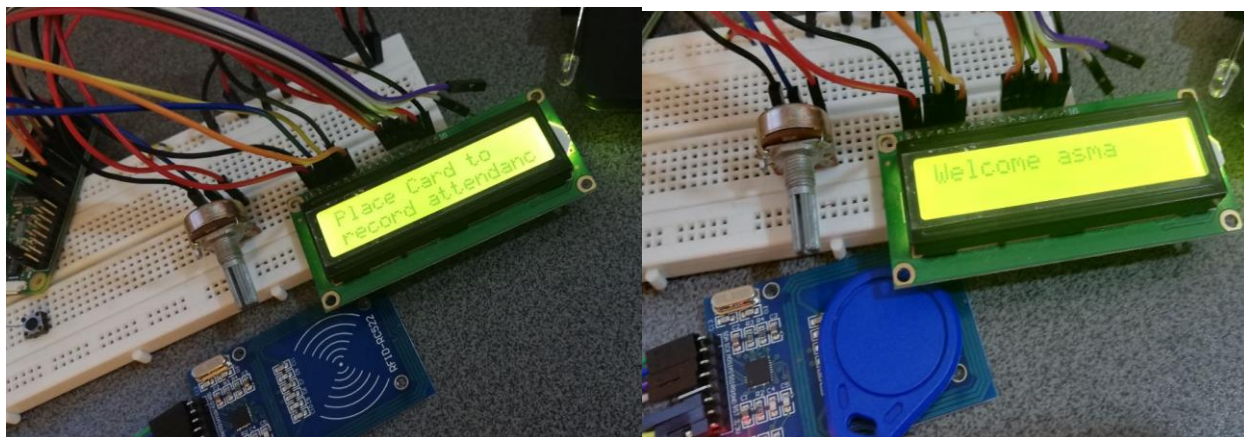
```

1  #!/usr/bin/env python
2  import time
3  import RPi.GPIO as GPIO
4  from mfrc522 import SimpleMFRC522
5  import mysql.connector
6  import Adafruit_CharLCD as LCD
7
8  db = mysql.connector.connect(
9      host="localhost",
10     user="attendanceadmin",
11     passwd="password",
12     database="attendancesystem"
13 )
14
15 cursor = db.cursor()
16 reader = SimpleMFRC522()
17
18 lcd = LCD.Adafruit_CharLCD(4, 24, 23, 17, 18, 22, 16, 2, 4);
19
20 try:
21     while True:
22         lcd.clear()
23         lcd.message('Place Card to\nrecord attendance')
24         id, text = reader.read()
25         #find id in user table:
26         cursor.execute("SELECT id, name FROM users WHERE rfid_uid="+str(id))
27         result = cursor.fetchone() #result[0]=id and result[1] =name form user talbe
28
29         lcd.clear()
30
31     while True:
32         lcd.clear()
33         lcd.message('Place Card to\nrecord attendance')
34         id, text = reader.read()
35
36         cursor.execute("Select id, name FROM users WHERE rfid_uid="+str(id))
37         result = cursor.fetchone()
38
39         lcd.clear()
40
41         if cursor.rowcount >= 1: #if there is this user
42             lcd.message("Welcome " + result[1]) #result[1] = user name
43             cursor.execute("INSERT INTO attendance (user_id) VALUES (%s)", (result[0]) #attendance save
44             db.commit()
45         else: #this card wasn't registered
46             lcd.message("User does not exist.")
47             time.sleep(2)
48     finally:
49         GPIO.cleanup()

```



در این کد بعد از خواندن کارت، شناسه کارت را در جدول "کاربران" پایگاه داده جستجو میکند. اگر شناسه ثبت شده باشد، حضور و ساعت آن را ثبت میکند. اما اگر کاربر ثبت نشده باشد پیام "User does not exist" را در نمایشگر چاپ میکند. بعد از 2 ثانیه تأخیر، نمایشگر را پاک میکند و منتظر کارت بعدی میماند.



## جداول پایگاه داده

### جداول پایگاه داده:

با کدهای زیر در ترمینال، به MySQL وصل می شویم و پایگاه داده مربوط به سامانه احراز هویت وصل می شویم و بعد جدول "کاربران" را انتخاب میکنیم برای دیدن:

```
sudo mysql -u root -p
use attendancesystem;
SELECT * FROM users;
SELECT * FROM attendance;
```

جدول کاربران:

```
MariaDB [(none)]> use attendancesystem;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with 'no_table_stats'

Database changed
MariaDB [attendancesystem]> SELECT * FROM users;
+-----+-----+-----+-----+
| id | rfid_uid | name | created |
+-----+-----+-----+-----+
| 1 | 932165081559 | asma | 2022-05-18 15:03:14 |
| 2 | 530833215699 | sina | 2022-05-18 15:10:36 |
+-----+-----+-----+-----+
2 rows in set (0.005 sec)

MariaDB [attendancesystem]>
```

جدول حضور:

```
MariaDB [attendancesystem]> SELECT * FROM attendance
-> ;
+-----+-----+-----+
| id | user_id | clock_in |
+-----+-----+-----+
| 1 | 2 | 2022-05-18 15:57:14 |
| 2 | 2 | 2022-05-18 15:57:17 |
| 3 | 2 | 2022-05-18 15:57:27 |
| 4 | 2 | 2022-05-18 15:57:30 |
| 5 | 1 | 2022-05-18 15:57:41 |
| 6 | 1 | 2022-05-18 15:57:43 |
| 7 | 2 | 2022-05-18 15:57:46 |
| 8 | 1 | 2022-05-18 15:57:53 |
| 9 | 1 | 2022-05-18 15:57:56 |
| 10 | 1 | 2022-05-18 15:57:59 |
```

## پیشنهادهات

ایرادی که به سامانه وارد است این است که در صورتی که یک کاربر اشتباها کارت را چندین بار (پشت سرهم کمتر از چند دقیقه) روی کارتخوان بگذارد، سامانه همه این داده های تکراری را ذخیره میکند و پایگاه حضور یک فرد در یک روز کاری را چندبار ذخیره میکند.

برای حل این مشکل، میتوان یک بازه زمانی تعریف کرد که در آن بازه فقط یک حضور ثبت شود. بدین گونه که به جدول "حضور"، یک ستون دیگر به نام isEnter اضافه کرد که داده bool بگیرد. اگر فردی کارت زد در صورتی که آن متغیر false بود حضور را ثبت کند و آن داده true شود. بعد از هر بازه مشخص مثلا 4 ساعت، همه مقادیر این متغیر را به false برگرداند تا برای ثبت حضور بعدی مشکلی نباشد و حضور ثبت شود.

راه پیشنهادی دیگر، اضافه کردن دو کلید برای تعیین "ورود" یا "خروج" به سامانه است که در صورت زدن کارت به همراه فشار کلید "ورود"، اجازه ثبت مجدد کارت به عنوان ورود را ندهد، مگر اینکه وضعیت فرد "خارج" باشد. برای این روش نیاز به اضافه کردن یک متغیر bool به جدول "حضور" داریم که ورود یا خروج فرد را ثبت کند.

این دو ایده، دو ایده اولیه هستند که بسته به داده های مورد نیاز محل استفاده سامانه، ایده ها میتوانند بهینه تر و تکمیل تر شوند.



<https://digispark.ir/getting-stated-with-python-rfid-and-raspberry-pi/>

<https://tuskanic.com/2022/01/11/-/کاراکتری-با-رزبری-پای-کد-یا-/cd-اراه-اندازی->

<https://pimylifeup.com/raspberry-pi-rfid-attendance-system/>

<https://pimylifeup.com/raspberry-pi-mysql/>

<https://www.easytechguides.com/raspberry-pi-os-update-problem/>