

# SISTEMA DE AUTENTIFICACIÓN MEDIANTE NFC

## DESARROLLO DE APLICACIONES MULTIPLATAFORMA

Curso 2018/19



Fernando Durán Torres



## ANEXO I: CALIFICACIÓN

Reunidos, en la fecha indicada, el Tribunal Evaluador, compuesto por los Profesores abajo firmantes, y vista la defensa del Proyecto Integrado denominado:

---

Presentado por D/D<sup>a</sup>:

---

---

Se otorga la calificación de: \_\_\_\_\_

Y para que así conste, se firma en Coín a\_\_\_\_\_de\_\_\_\_\_de 20\_\_\_\_

El Tribunal Evaluador

D/D<sup>a</sup>\_\_\_\_\_D/D<sup>a</sup>\_\_\_\_\_D/D\_\_\_\_\_



## AGRADECIMIENTOS.

Me gustaría agradecer a todas las personas que hicieron posible este proyecto, ya que sin sus herramientas o conocimientos nunca lo hubiera sacado adelante. Gracias al profesorado Dña. Ana María Cabello, Dña. Silvia Cáceres, D. Rafael Rubio por estos años de enseñanza y conocimientos en programación, sistemas operativos, redes y base datos. A los conocimientos de electrónica de D. Rafael que me enseño la creación y diseño de una pequeña PCB.

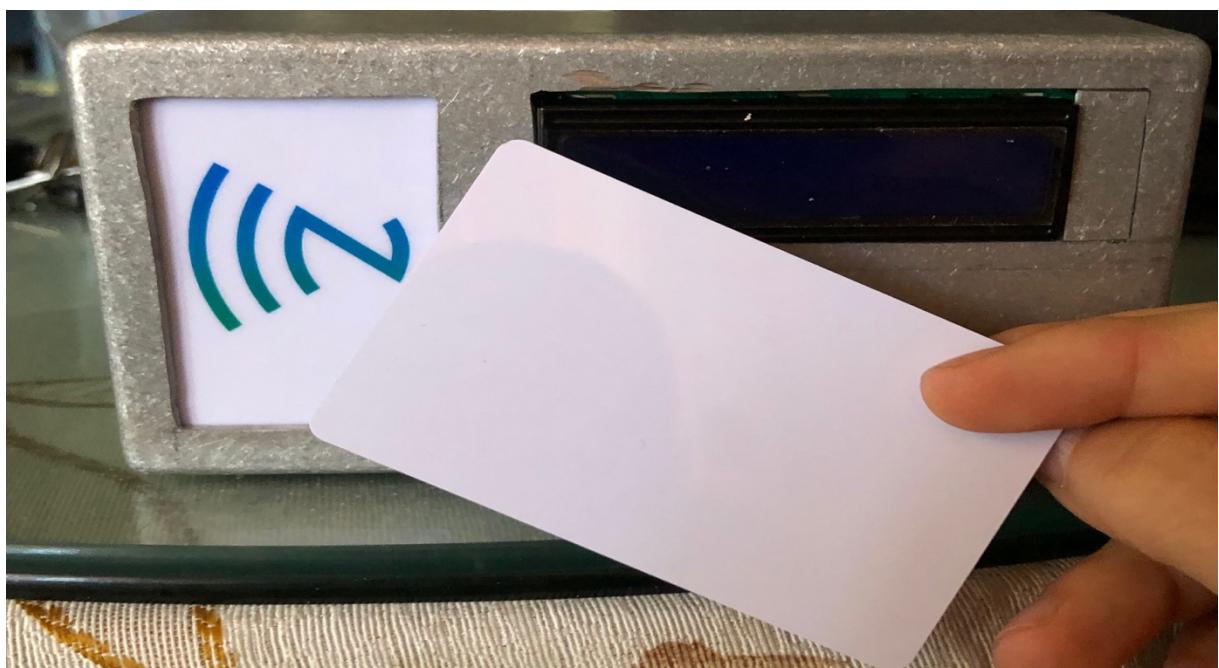
También me gustaría agradecer a amigos como Antonio López cuya amabilidad y conocimientos me ayudaron a realizar algunos cortes y pegados sobre la maqueta final. Por otro lado, esta también mi cuñado Pablo Carrasco cuyos conocimientos y herramientas me permitieron realizar los últimos retoques de la PCB. Por último, a mi familia por aportar también su pequeño granito de arena en el proyecto, aguantando el desorden y frustraciones que me llevaron confeccionarlo.

## Contenido

ANEXO I: CALIFICACIÓN.....	3
AGRADECIMIENTOS.....	5
Contenido.....	6
CAPÍTULO 1. INTRODUCCIÓN Y JUSTIFICACIÓN.....	7
CAPÍTULO 2. OBJETIVOS .....	9
CAPITULO 3. ANTECEDENTES .....	11
CAPITULO 4. FASES DE TRABAJO .....	11
CAPÍTULO 5. PRUEBAS DE FUNCIONAMIENTO.....	31
CAPÍTULO 6. PROBLEMAS ENCONTRADOS.....	39
CAPÍTULO 7. MEJORAS PROPUESTAS .....	41
CAPÍTULO 8. CONCLUSIONES.....	43
CAPÍTULO 9. REFERENCIA Y BIBLIOGRAFÍA.....	45

## CAPÍTULO 1. INTRODUCCIÓN Y JUSTIFICACIÓN

iLockPi comienza con la idea de realizar un sistema de autentificación para desbloquear una puerta con un cierto nivel de seguridad o con control de acceso elevado. El método favorito de autentificación será mediante una tarjeta NFC. Desde el primer momento se tenía claro los objetivos, y el equipo informático que se necesitaría. Este sistema funcionaría con una base de datos, en dicha base de datos estarán los usuarios que podrán abrir la puerta con su tarjeta de identificación.



Para la gestión del personal se realizará mediante una plataforma web llamada [ilockpanel.tk](http://ilockpanel.tk). En dicha plataforma se podrán meter nuevos usuarios, eliminar o modificar administradores del sistema y también se podrán gestionar el personal que podrá desbloquear las puertas. Dicha plataforma se apoya de una simple app en Android que mostrará información algo limitada sobre el sistema.

Todo esto esta gestionando con un mini-pc llamado Raspberry Pi en este caso en su versión 3. Este dispositivo cuenta con un sistema operativo Linux. Esto me permitirá hacer uso de muchas librerías y funciones de este sistema operativo. También tiene una fluidez y rapidez que necesito. Dicho sistema se apoya en una mini SD, esto le da aun mayor rapidez.

Hay muchas variantes de equipos de este tipo. Arduino es uno de ellos, este mini-pc cuenta con un firmware un poco simple. Es capaz de hacer funcionar el proyecto, pero su entorno de programación se hace un poco engorroso. No cuenta con un sistema operativo como tal, ya que usa uno sistema embebido. Para este proyecto era necesario tener un sistema capaz de tener un reloj actualizado mediante internet y también importante una tarjeta de red eficiente. Era necesario un sistema capaz de poderse gestionar de forma remota o incluso mediante conexión SSH para su mantenimiento.

La Raspberry Pi 3 cuenta con la tecnología, wifi muy importante en este proyecto, también cuenta con tecnología Bluetooth y un sistema operativo Linux. En este caso el sistema usado se llama Raspbian en su versión de kernel de Linux 4.4.



Cuenta con un procesador ARM-v8 de 1,2 Ghz de 64 bits, 1 GB de RAM, WIFI (802.11n) y Ethernet (100M), salida USB para periféricos o dispositivos de almacenamiento, salida HDMI y mini-jack 3.5. Este dispositivo es capaz de poder conectarle una pantalla táctil o una cámara de video o una multitud de sensores gracias a su puerto GPIO de 40 pines.

## CAPÍTULO 2. OBJETIVOS.

El problema principal a resolver esta en dotar con seguridad a una puerta o lugar de trabajo con cierta seguridad. Por ejemplo, en una oficina donde solo los trabajadores autorizados tienen acceso la entrada a una zona restringida. También es funcional para una clase, cuyos profesores tienen el poder de abrir la puerta del aula al inicio de un turno de clase. Para la autentificación se usará tarjetas llamadas con un chip NFC. Este chip usa un código único como un DNI.

Los objetivos a cumplir estaban claros de un principio, están enfocados a la seguridad. Pensando en una posible manipulación externa. Los objetivos son los siguientes:

1. Tarjetas de identificación.
2. Base datos con el personal autorizado.
3. Panel de control para visualizar todos los datos de las tarjetas basado en un sitio web basado en el framework de PHP llamado CodeIgniter.
4. Diseño responsive en la plataforma web.
5. Sistema físico y cableado para llevar a cabo la apertura.
6. Mecanismo magnético que desbloquea la puerta.
7. Software de autentificación que realiza las consultas a una base de datos.
8. Software de registro de nuevas tarjetas de autentificación.
9. App básica de funcionamiento en Android.
10. Servidor web para la apertura rápida desde una app de móvil.

Una de las limitaciones de este sistema es la dependencia de internet. Porque su capacidad principal recae en una base de datos externa al dispositivo. Puede estar ubicada en el mismo edificio o como en este caso de forma externa.

Algunas de las futuras mejoras a complementar en este proyecto debido a la complejidad y el poco tiempo que se dispone serian las siguientes:

1. Interfaz grafica en el software de autentificación, actualmente esta en modo terminal.
2. Interfaz grafica en el software de registro de nuevas tarjetas NFC.
3. Posibilidad de registro de nuevas tarjetas mediante la app de Android.
4. Registro de nuevas tarjetas mediante la plataforma web.
5. Mejoras de seguridad para evitar intrusos no deseados.
6. Posibilidad de agregar lector de huella como un método mas de autentificación.
7. Mejoras en la app móvil pudiendo modificar información del sistema. Actualmente solo permite visualización de datos. Si se quiere modificar el sistema es necesario ir a la web.  
Esta web tiene un formato responsive para móviles, tabletas o pc.
8. Mayor rapidez en los tiempos de apertura de la puerta.

Este proyecto tiene también como objetivo demostrar que con poco presupuesto e ingenio se puede lograr un sistema seguro y funcional. Todos los componentes son muy baratos y fácil de conseguir. Como objetivos de este proyecto enfocado al mundo laboral me gustaría destacar:

- Mantenimiento fácil y rápido.
- El Sistema es 100% modular.
- Gasto energético muy bajo.
- Fácil de encontrar repuestos físicos en caso de falla.
- Automatización en cuanto seguridad se refiere.
- Creación de zonas seguras.
- Incremento del valor de la oficina.

## CAPITULO 3. ANTECEDENTES

La idea de este sistema de seguridad es debido a una pequeña excursión realizada a una empresa del parque tecnológico. Esta empresa tenía una sala en la cual, solo un pequeño grupo autorizado por el gobierno podía entrar en ella. El método para abrir dicha puerta de seguridad era un sistema similar al mío. También nace en base a una visita al sitio web de blog.bricogeek.com donde tienen un post que hablan detenidamente de este sistema.

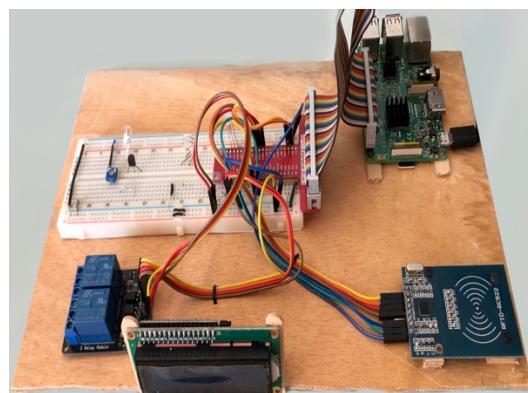


En el explican como llevo acabo un ejemplo similar al mío. Pero en este caso con una Raspberry pi 1 y algunos elementos más. Visitando otro blog, decidí añadir una pantalla de 16x2 en donde mostraría información del estado del sistema en tiempo real.

## CAPITULO 4. FASES DE TRABAJO

Para llevar comenzar con este proyecto decidí empezar por la parte del físico del proyecto. Me dispongo a reunir todos los elementos básicos para llevarlos a cabo. En un chapo de madera y con algo de pegamento posiciono los elementos que son:

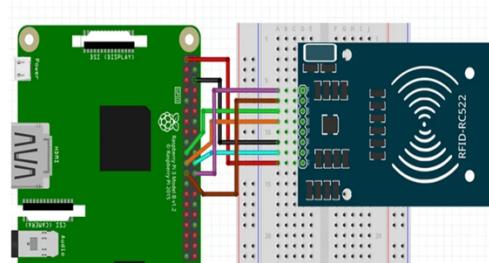
- Lector RFID-RC522
- Relé de 2 salidas y 4 pines.
- LCD con modulo Serial incorporado
- Cableado variado
- Led RGB
- Protoboard



El esquema de conexionado de los elementos a la protoboard y después de este a la Raspberry es el siguiente:

## 1. Modulo RFID-RC522

- SDA conectar a Pin 24 – GPIO 8
- SCK conectar a Pin 23 – GPIO 11
- MOSI conectar a Pin 19 – GPIO 10
- MISO conectar a Pin 21 - GPIO 9
- GND conectar a Pin 6
- RST conectar a Pin 22 – GPIO 25
- 3.3v conectar a Pin 1



## 2. RELE

- GND conectar al Pin 6
- IN1 conectar al Pin 38 – GPIO 20
- IN2 conectar al Pin 36 - GPIO 16
- VCC conectar al Pin 2

## 3. LCD

- GND conectar al Pin 6
- VCC conectar al Pin 2
- SDA conectar al Pin 3 – GPIO 2
- SCL conectar al Pin 5 – GPIO 3



## 4. LED

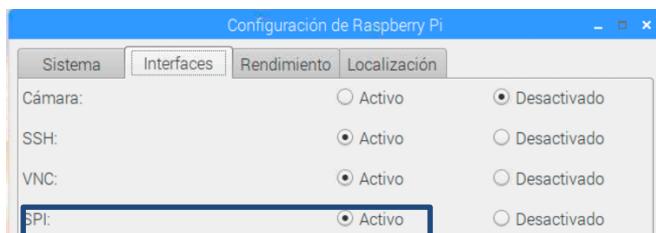
- ROJO conectar al Pin 33 - GPIO 13
- GND - GPIO 6
- VERDE conectar al Pin 35 - GPIO 19
- Azul conectar al Pin 37 - GPIO 26

Una vez todo conectado comencé a instalar las librerías de Python 3 que necesitaba, mediante pip.

- sudo pip3 install spidev
- sudo pip3 install mfrc522
- sudo pip3 install mysqlclient
- sudo pip3 install datetime

El siguiente paso que tome fui habilitar la interface SPI que por defecto viene desactivada.

Es necesario reiniciar después de esto.



Para comprobar que todo este correctamente funcionando podemos poner el comando **lsmod | grep spi**, este mostrara si el **spi\_bcm2835** esta funcionando correctamente.

A terminal window titled 'pi@raspberrypi3: ~'. The user runs the command 'lsmod | grep spi'. The output shows two modules: 'spidev' and 'spi\_bcm2835', both with an ID of 16384 and a reference count of 0.

```
pi@raspberrypi3:~ $ lsmod | grep spi
spidev                 16384      0
spi_bcm2835            16384      0
pi@raspberrypi3:~ $
```

Después de que tengamos todo configurado podemos comenzar con el primer programa de pruebas. Este permite leer el id único de la tarjeta.

```
#!/usr/bin/env python
import RPi.GPIO as GPIO
from mfrc522 import SimpleMFRC522
reader = SimpleMFRC522()
try:
    id, text = reader.read()
    print(id)
    print(text)
finally:
    GPIO.cleanup()
```

Este simple código importa las librerías para el uso del gpio y la anteriormente instalada mfrc522. Creo una variable llamada reader que hace una llamada a la función SimpleMFRC522(). Dentro del try igualo la variable id, text a la función read() para posteriormente imprimir por pantalla el id y el texto si tuviera alguno escrito en la tarjeta. Por ultimo se pone la palabra reservada finally para cuando termine toda la ejecución deje libre los puertos GPIO. Esta línea es crucial, ya que la falta de limpieza puede evitar que otros scripts funcionen correctamente.

Mi software se basa en este simple scripts anteriormente mencionado. Se ramifica en dos, uno gestiona el registro en la base datos y el otro contiene un bucle infinito donde abrirá la puerta después de verificar al usuario. Antes de explicar como funcionan ambos programas necesito comentar las diferentes categorías de seguridad que tiene la autentificación.

1. El usuario tiene que estar registrado en la base datos.
2. La tarjeta puede estar activada o no. Si dicha tarjeta no esta activada no se puede acceder. Este caso contempla si el usuario se pone de baja, esta de vacaciones etc.

El software de registro llamado “**RegistroCard.py**” registra nuevas tarjetas para un nuevo usuario. Tiene las siguientes características:



Me gustaría comentar primero la base datos, ya que mas adelante mostrare las consultas que se realizan. Esta base datos MYSQL esta en un hosting llamado Hostinger. La base datos depende de un dominio llamado [www.ilockpanel.tk](http://www.ilockpanel.tk). La base datos tiene un nombre que asigna el hosting aleatorio. Para este software creo una tabla que tiene las siguientes características (La id es auto incrementable):

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	id	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar  Eliminar ▾ Más
2	nombre	varchar(200)	utf8mb4_unicode_ci		Sí	NULL			Cambiar  Eliminar ▾ Más
3	uid	varchar(16)	utf8mb4_unicode_ci		Sí	NULL			Cambiar  Eliminar ▾ Más
4	fechaRegistro	datetime			No	Ninguna			Cambiar  Eliminar ▾ Más
5	foto	varchar(300)	utf8mb4_unicode_ci		No	Ninguna			Cambiar  Eliminar ▾ Más
6	activo	int(2)			No	Ninguna			Cambiar  Eliminar ▾ Más

```
# Librerias
import RPi.GPIO as GPIO
import mysql.connector

from mfrc522 import SimpleMFRC522
from releOFF import releApaOFF
import datetime
import time
import lcddriver
```

Empiezo en primer lugar instanciando las librerías que voy hacer uso. Importo la librería **GPIO**, esta es la encargada de activar los puertos GPIO. Importo la librería **MYSQL**, esta va ser la encargada de realizar las consultas a la base datos. Seguidamente llamo a la librería encargada del RFID 522. Por otro lado, importo **releOFF** este script se encarga de solucionar un problema que hacia, dicho problema recae en que el relé actuaba de forma inversa. Lo mostrare en el **capítulo de problemas encontrados**.

Continuando con el programa el declaro las constantes encargada de poder encender los LED RGB y declaro las variables para la salida de los mensajes por la LCD. También declaro la variable datoDB este se encarga de recoger los datos de inicio sobre la base datos.

```
# GPIO PARA LOS LED
GPIO.setmode(GPIO.BCM) # Variable para la LCD
GPIO.setwarnings(False) lcd = lcddriver.lcd()
GPIO.setup(26, GPIO.OUT) # led azul
GPIO.setup(13, GPIO.OUT) # led rojo # Variable para la conexion con la libreria MFRC522
GPIO.setup(19, GPIO.OUT) # led Verde reader = SimpleMFRC522()

# Carga de la BaseDatos
datoDB = {
    'user': 'u974320120_ilock',
    'password': 'Coin2019@',
    'database': 'u974320120_ilock',
    'host': 'sql23.main-hosting.eu'
}
```

El programa empezara limpiando la LCD con la función lcd.lcd\_clear(). La LCD informara al usuario que ponga la tarjeta sobre el lector. Esta función permite distinguir en fila de la LCD muestre el mensaje. El mensaje también lo muestra mediante la terminal.

```
lcd.lcd_clear()
lcd.lcd_display_string("Ponga la tarjeta",1)
lcd.lcd_display_string("sobre el lector",2)

print("\n#####")
print("Ponga la tarjeta sobre el lector")
print("\n#####")
```

Declaro una variable llamada idcard que recogerá el código único recogido de la tarjeta. Una vez leída esa id es debido a que el usuario coloco la tarjeta en el lector. Este id lo mostrara en pantalla del terminal y por la LCD.

```
idcard = reader.read_id()
print("El codigo leido es: ",idcard)
co = str(idcard)
lcd.lcd_clear()
lcd.lcd_display_string("Codigo: ",1)
lcd.lcd_display_string(co,2)
time.sleep(5)
```

Realizo la conexión a la base datos, le paso por parámetro la variable anteriormente declarada datoDB. La consulta que lanza es muy simple, se trata de un **select** que comprueba si el id esta ya registrado. El cursor ejecuta la consulta anterior. Si la respuesta del cursor devuelve vacío te informará al usuario mediante la LCD, por la terminal y encenderá el led en color **verde**. Eso quiere decir que se puede inscribir datos nuevos sobre ella.

```

conexión = mysql.connector.connect(** datoDB)
cursor = conexión.cursor()
sql = "SELECT * FROM personal WHERE uid='"+ str(idcard) + "'"

# Ejecuta el sql
cursor.execute(sql)

# Filas obtenidas como respuesta a la consulta
filas = cursor.fetchall()

if filas == []:
    print("\nTarjeta Vacia")
    lcd.lcd_clear()
    lcd.lcd_display_string("Tarjeta Vacia: ",1)
    ledVerde(17)
    escrituraCard(datoDB, idcard)
    GPIO.cleanup()
else:
    print("\nTarjeta ya registrada")
    lcd.lcd_clear()
    lcd.lcd_display_string(" Tarjeta ",1)
    lcd.lcd_display_string(" ya registrada",2)
    ledRojo(13)
    renuevo(idcard)

conexión.close()

```

simple función llamada renuevo que le asigna al usuario la posibilidad de cancelar la operación en caso de que sea un error, si por lo contrario no lo fuera este procederá a borrar toda la información de la tabla personal relacionada con esta tarjeta. Llegado a este punto el programa lanzara una nueva función encargada de pedir al usuario un nombre de registro.

Dentro de la función escrituraCard llamará a datosDB que es la encargada de realizar un **insert into** a la tabla personal. Esta función recibe por parámetro el nombre, el id y los parámetros de la base datos. La query también inserta la fecha actual con la función datetime. Esta fecha es la actual del sistema. Esta dentro de un try, catch que informara al usuario que no se pudo insertar los datos en caso de falla. También

```

# Insertar datos en la base datos.
def datosDB(a, b, datoDB, idcard):

    # Fecha y hora para insertarlo en la base datos
    x = datetime.datetime.now()

    conexión = mysql.connector.connect(** datoDB)
    cursor = conexión.cursor()

    try:
        query = "INSERT INTO personal (nombre, uid, fechaRegistro) VALUES (%s, %s, %s);"
        cursor.execute(query, (a, b, x))
        conexión.commit()
        print("Se insertaron de forma correcta")
        lcd.lcd_clear()
        lcd.lcd_display_string("Datos insertados",1)
        lcd.lcd_display_string("En la Base Datos",2)
        time.sleep(4)
        ledAzul(26)
    
```

encenderá el led en color **azul** si todo fue bien. La siguiente ejecución del programa informara que es necesario ir a la web ilockpanel.tk para subir una foto de la persona registrada.

```
#activa la card en la base datos y permitira el acceso
def activarCard(datoDB, idcard):

    conexion = mysql.connector.connect(** datoDB)

    cursor = conexion.cursor()

    icard = "UPDATE personal SET activo = %s WHERE uid = %s"
    val = ("1",(idcard))

    try:
        cursor.execute(icard, val)
        conexion.commit()
        print("Tarjeta Activada, Esta Lista para usarse\n\n")
        lcd.lcd_clear()
        lcd.lcd_display_string("Tarjeta Activada",1)
        time.sleep(5)
        lcd.lcd_clear()
        GPIO.cleanup()

    except:
        print("No se ha podido activar la tarjeta")

    conexion.close()
```

Se le preguntara al usuario si desea activar la tarjeta. Si no lo activa esta no será valida para posteriormente acceder al sistema. Esta opción esta diseñada por si un usuario no esta temporalmente en la oficina (por vacaciones, despido o otros motivos). En caso de querer activarla ahora el programa llamara a una función encargada de hacer un **update** en la base datos. Esta actualiza en la columna activo de 0 a 1. Si se inserta un 1 este permitirá acceder posteriormente. Como siempre esta dentro de un try, catch para controlar las excepciones e informar al usuario.

activa la card en la base datos y permitira el acceso

Me gustaría explicar por ultimo el funcionamiento de la función borrarRegistro anteriormente mencionada, esta recibe por parámetro los datos de acceso de la base datos y la id de la tarjeta. Realiza un **delete** en la tabla personal donde coincide el id que se le a pasado previamente. Por último, informa al usuario que se a borrado los datos de forma correcta.

```
def borrarRegistro(datosDB, idcard):

    conexion = mysql.connector.connect(** datosDB)

    cursor = conexion.cursor()

    icardelete = "DELETE FROM personal WHERE uid = %s"

    val = ((idcard),)

    try:
        cursor.execute(icardelete, val)
        conexion.commit()
        print("\nDatos borrados, esta lista para ser registrada\n")
        lcd.lcd_clear()
        lcd.lcd_display_string("Datos Borrados",1)
        time.sleep(5)
        lcd.lcd_clear()

    except:
        print("No se ha podido borrar la tarjeta")

    conexion.close()
```

Ahora hablare del programa que hace toda la magia. Este es llamado es llamado “**AccesoCard.py**” es esta basado en el programa anteriormente explicado. La gran diferencia es que incorpora un bucle infinito. Este bucle terminara cuando se detecte la señal Ctrl+c.

```
def main():

    releApaOFF()

    # Enlaza SIGINT (teclas Ctrl+C) con la funcion end_read()
    signal.signal(signal.SIGINT, finalizar)

    while bucle:
        print("\n")
        consulta(datoDB)
```

El bucle ejecuta la función consulta, esta función pide al usuario que ponga la tarjeta sobre el lector. Realiza la consulta pertinente a la tabla personal pasándole como dato el id de la tarjeta. Una vez realizado este proceso se procesa el resultado obtenido por la query realizada. Si esta devuelve vacío, se imprimirá denegado dando a entender que no existe registro alguno. Como medida extra y meramente asustadiza el led que lleva incorporado la LCD se apagara unos segundos, causando confusión al intruso no deseado.

```
if filas == []:
    print("Acceso denegado")
    lcd.lcd_clear()
    lcd.lcd_display_string("Acceso denegado",1)
    time.sleep(5)
    lcd.lcd_clear()
    lcd.lcd_backlight("off")
    ledRojo(13)
```

Por lo contrario, si esta devuelve algo, como medida extra de seguridad realiza una nueva consulta a la base datos. En esta consulta se quiere comprobar si en la columna activo existe “1” de activo o un “0” de no activo. En caso de encontrar 1 este informara de que el acceso esta permitido, activara el led en color **verde** y activara el relé. Este lo que permitirá es accionar el mecanismo de apertura de la puerta, dándole carga positiva y desbloqueando el perno que lleva en su interior. También como ultimo paso realiza un registro de la persona que accedió, id de esa tarjeta y la fecha en la tabla acceso. Todo esto mediante un **insert into** que lo realiza la función registroDB(), (el nombre lo obtiene haciendo un select a la tabla persona donde se le pasa el id de la tarjeta).

```

activosql = "SELECT activo FROM personal WHERE uid='" + str(idcard) + "'"
cursor.execute(activosql)
filasActivo = cursor.fetchall()

for x in filasActivo:
    if x == (1,):
        print("Acceso Permitido")

        lcd.lcd_clear()
        lcd.lcd_display_string("Acceso Permitido",1)
        time.sleep(5)

        ledVerde(19)
        rele(20)
        a = nombre(datoDB, idcard)
        registroDB(a,idcard, datoDB)

    else:
        print("Tarjeta no activada, Acceso denegado")

        lcd.lcd_clear()
        lcd.lcd_display_string("Acceso denegado",1)
        lcd.lcd_display_string("Card no activada",2)
        time.sleep(5)

        ledRojo(13)

```

```

def nombre(datoDB, idcard):

    conexion = mysql.connector.connect(** datoDB)

    cursor = conexion.cursor()

    try:
        activosql = "SELECT nombre FROM personal WHERE uid='" + str(idcard) + "'"
        cursor.execute(activosql)

        filasActivo = cursor.fetchall()

        for row in filasActivo:
            return row[0]

    except:
        print("Error al cargar el nombre")

    conexion.close()

```

La función que activa el relé básicamente lo que hace es enviar una señal de apagado al pin 20 donde esta conectado. Esta señal dura 3 segundos y vuelve activar el pin para apagar el relé.

```

def rele(pin):
    # Activa relay
    GPIO.output(20,False)
    print ("Relay activado.")
    time.sleep(3)

    # Desactiva relay
    GPIO.output(20,True)
    print ("Relay desactivado.")

```

Como anteriormente dije el relé funciona de forma inversa. Cuando se le da corriente este corta el paso de corriente y cuando no tiene corriente deja pasar la energía. Esto lo soluciono mas adelante y se explica en el **Capítulo Problemas Encuentados**.

Me gustaría también explicar el funcionamiento de instalación de la pantalla LCD. Es un modelo 16x2 y funciona con I2C. Esto quiere decir que lleva acoplada un circuito impreso que hace de comunicación con la Raspberry Pi. Como es un protocolo en BUS, es decir, podemos intercalar más dispositivos I2C si quisiéramos. Antiguamente eran necesarios 16 cables, con esta mejora solo usamos 4 que serían:

- GND
- VCC
- SDA
- SCL



Pin 3 es SDA y 5 es SCL

```
Raspberry Pi Software Configuration Tool (raspi-config) |  
A1 Overscan You may need to configure oversca  
A2 Hostname Set the visible name for this Pi  
A3 Memory Split Change the amount of memory made  
A4 SSH Enable/Disable remote command lin  
A5 Device Tree Enable/Disable the use of Device  
A6 SPI Enable/Disable automatic loading  
A7 I2C Enable/Disable automatic loading  
A8 Serial Enable/Disable shell and kernel m  
A9 Audio Force audio out through HDMI or 3  
AA GL Driver Enable/Disable experimental deskt
```

Para configurarla en la Raspberry Pi necesitamos que activar previamente la interfaz I2C. Después de esto necesitamos instalar unas dependencias al sistema como:

```
sudo apt-get install python-smbus i2c-tools
```

Esto instalara lo necesario para que nuestra pantalla funcione de forma correcta. Reiniciamos y pasamos a la siguiente fase. Vamos a ver si el sistema detecta nuestro dispositivo I2C en el bus número 1. En las primeras versiones de Raspberry el bus estaba numerado como cero. El Id auto asignado por el dispositivo I2C es el 27. Suele ser ese número para estas pantallas. No puede haber dos con el mismo ID porque no sería posible la comunicación.

```
pi[Raspberry Pi 3]~ sudo i2cdetect -y 1
0 1 2 3 4 5 6 7 8 9 a b c d e f
00: -- - - - - - - - - - - - - - - - -
10: - - - - - - - - - - - - - - - - - -
20: - - - - - - - - - - 27 - - - - - - - -
30: - - - - - - - - - - - - - - - - - -
40: - - - - - - - - - - - - - - - - - -
50: - - - - - - - - - - - - - - - - - -
60: - - - - - - - - - - - - - - - - - -
. 70: - - - - - - - - - - - - - - - -
```

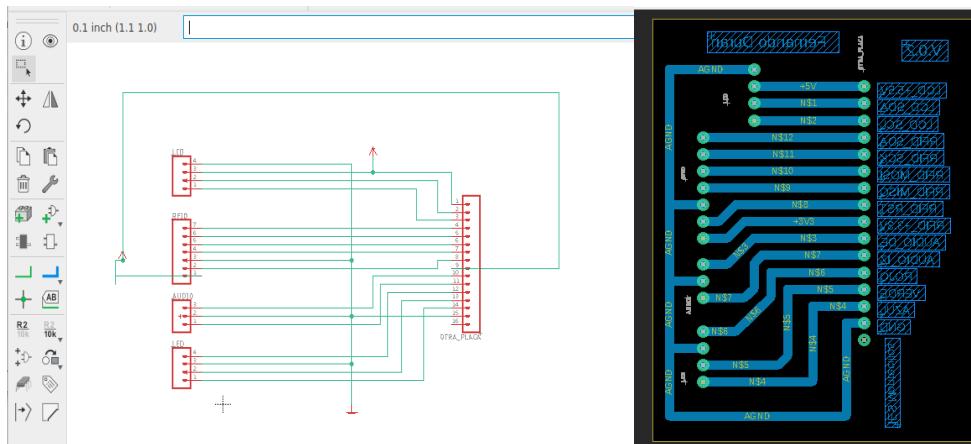
Una vez realizado esto es necesario descargarse la librería necesaria. En ella hay que editarla y poner el id de I2C que en mi caso es el 27.

```
import sys
sys.path.append("./lib")

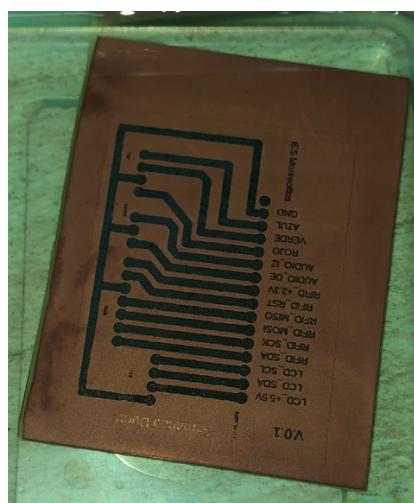
import i2c_lib
from time import *

# LCD Address
ADDRESS = 0x27
```

El siguiente paso es diseñar una placa lógica donde poder conectar todos los componentes que llevara la caja donde ira alojado el sensor. En este caso usare el programa Eagle. Este software permite diseñar PCB de forma fácil. Primero hay que arrastrar los componentes necesarios y realizar las uniones. En este caso serán conectar todos los componentes como la LCD, el lector NFC, el LCD y audio. El siguiente paso es realizar las uniones o pistas de la PCB.



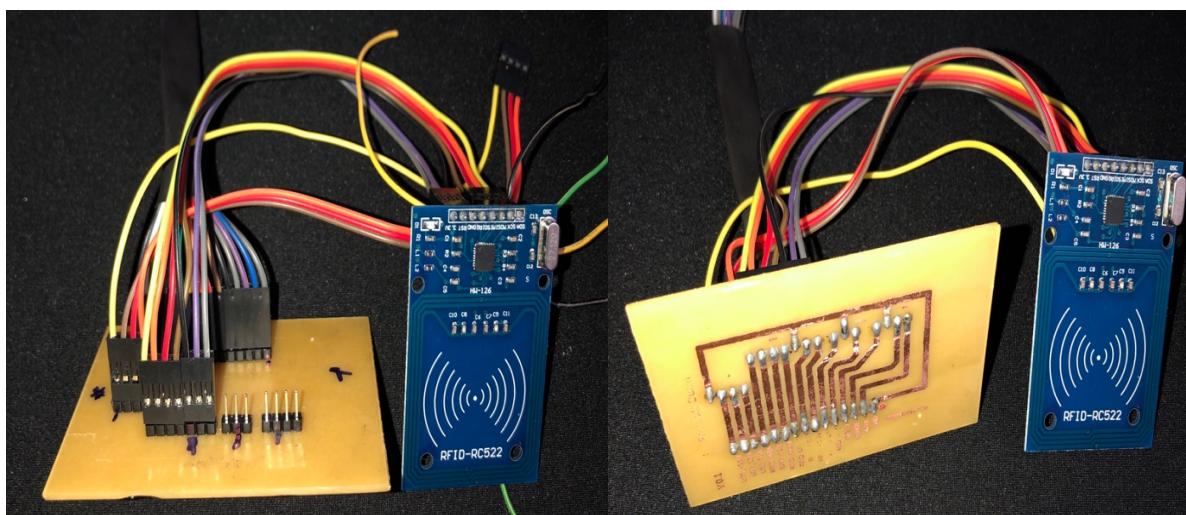
Una vez impreso en papel estucado y usar el método del planchado que consiste en pasar el tóner a la PCB mediante el calor. Después hay que pasarlo por una solución que contiene agua oxigenada, agua y agua fuerte. Este proceso quitara todo el cobre de la PCB dejando solo el cobre pintado con el tóner.



Una vez realizado el proceso químico ya se puede perforar la placa. En este caso fue con la ayuda de mi cuñado Pablo Carrasco, con una broca de 1 mm. Ya solo queda soldar los pines de comunicación.



Este seria el resultado final de todo el proceso.



La fabricación de la caja tubo su pequeña tarea. Se trata de una caja de aluminio. Ya que el aluminio es un material blando me dispuse a hacer las medidas y realizar los cortes con una segueta.



Resultado final:



Terminada la explicación de la parte física y la parte funcional de la Raspberry Pi toca explicar la parte administrativa online de este proyecto.

En las prácticas desarrolladas en Kibo Studios, pude ganar algo de soltura con PHP, CSS y HTML. Desarrollé una plataforma llamada ilockpanel.tk



Usuario: \*

Contraseña: \*

[Entrar](#)

[¿Has olvidado tu contraseña?](#)

```

Puedelogin.php
<?php
class Puedelogin extends CI_Model
{
    function can_login($email, $pass)
    {
        $this->db->where('email', $email);
        $query = $this->db->get('administradores');
        if($query->num_rows() > 0)
        {
            foreach($query->result() as $row){
                if($row->email_validado == '1')
                {
                    return true;
                }
                else
                {
                    return false;
                }
            }
        }
        else
        {
            return 'Wrong Email Address';
        }
    }

    function index()
    {
        $this->load->view('login_view');
    }

    function auth()
    {
        $email = $this->input->post('correoUserlogin', TRUE);
        $password = md5($this->input->post('passlogin', TRUE));
        $validate = $this->login_model->validate($email, $password);

        $result = $this->Puedelogin->can_login($this->input->post('correoUserlogin'), $this
            ->input->post('passlogin'));
    }
}

```

La plataforma tiene de base el framework llamado Codeigniter. Este usa el modelo, vista, controlador. La parte grafica esta desarrollada en código HTML que seria la vista, en el modelo recibe los datos mediante el método post de para realizar las comprobaciones.

Una vez validado el formulario se llama al modelo. Este se encargará de hacer una llamada a la base datos. En este caso se llama Puedelogin.



Este devuelve true si el email fue encontrado y devuelve falso en caso de que no. Una vez autenticado la plataforma mostrara información de las tablas. El proyecto tiene en concreto 3 tablas. Tabla Administradores, tabla personal y tabla acceso. Con el mismo funcionamiento anteriormente explicado se realiza las consultas para poder mostrar la información.

Para poder mostrar toda información necesite realizar una consulta a la base datos. Dicha consulta combina la información de la tabla personal con la tabla acceso. Para que los datos mostrados tengan un poco de sentido necesitaba mostrar la ultima fecha de acceso. Todo esto lleva a una mega consulta como esta:

```

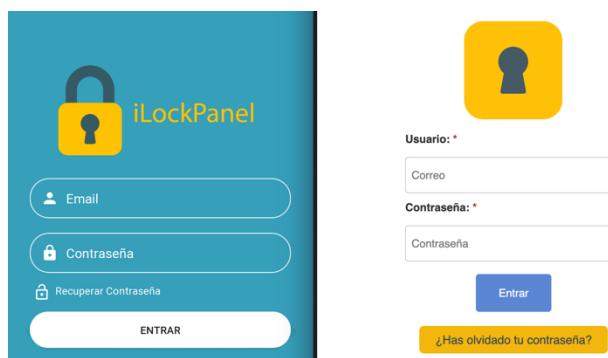
$query = 'SELECT personal.*,acceso.fechaEntrada FROM personal left join acceso
          on personal.uid = acceso.uid where acceso.fechaEntrada in (select max(
          fechaEntrada) from acceso group by acceso.uid) or acceso.fechaEntrada is
          NULL';

$resultados = $this->db->query($query);
return $resultados->result();

```

Esta consulta esta realizada en el modelo. El botón activo funciona mediante una consulta Ajax. Esta consulta realiza un update en la columna activo en la tabla personal. Este botón es 100%, actualmente en la consulta esta desactivado. Al tener este estado si quiera autenticarme con este UID no se podría acceder.

Por ultimo se realizo una app muy básica en Android. Dicha app esta diseñada para mostrar solo información. Por ahora no permite las mismas funcionalidades que la web. Si se esta en un dispositivo móvil se puede hacer uso de la capacidad responsive de la plataforma online. La plataforma también se desarrollo usando Bootstrap, este framework de CSS permite diseñar una web fácil mente y adaptar a todas las pantallas. A la izquierda se puede ver la app de Android y a la derecha el diseño web responsive.



```
@Override
protected String doInBackground(String... params) {
    hashMap.put("email",params[0]);
    hashMap.put("password",params[1]);
    finalResult = httpParse.postRequest(hashMap, HttpURL);
```

Nada mas abrir la app podemos ver una ventana de login. Dicha ventana se valida al momento de hacer evento click sobre el botón entrar. Este dispara una función que realiza una consulta a la base datos, a la tabla administradores. La función usa json para poder comunicarse, dicha consulta se hace en el lado del servidor mediante PHP. La url del servidor es la siguiente:

```
String HttpURL = "http://app.ilockpanel.tk/login/UserLogin.php";
```

Se declara como un String. Decidí crear un subdominio llamado app. Esto me permitirá diferenciar los archivos de framework Codeigniter y los de la app de Android.



La función recibe por parámetro el correo y la contraseña. Realiza un hashMap para enviarlo a la función PHP. Si esta es devuelve los parámetros correctos cargara la vista dashboard. Esta vista contiene un menú con diferentes opciones. Entre ellas visualiza las diferentes tablas y datos de la base datos mediante un Recycler View.

La función PHP encargada de la consulta es la siguiente. Este recibe por POST el email y la contraseña. La contraseña se le hace un proceso extra que es convertirla a md5, esta esta cifrado en la base datos. La query es simple y devuelve si el email este o no esta entre las tablas.

Si existe un match este informara mediante un echo y hará que cargue la nueva actividad.

```
$email = $_POST['email'];
$password = $_POST['password'];

$passwordMD = md5($password);

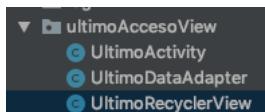
$sql_Query = "select * from administradores where email = '$email' and pass = '$passwordMD' ";

$check = mysqli_fetch_array(mysqli_query($con,$sql_Query));

if(isset($check)){
    echo "Datos Encontrados";
}
```

La actividad que carga es la siguiente, esta actividad mostrara el correo de la persona logueada. Posteriormente se muestra 4 botones. Estos botones llevan a otra actividad, estas están previamente cargada en el archivo AndroidManifest.xml.

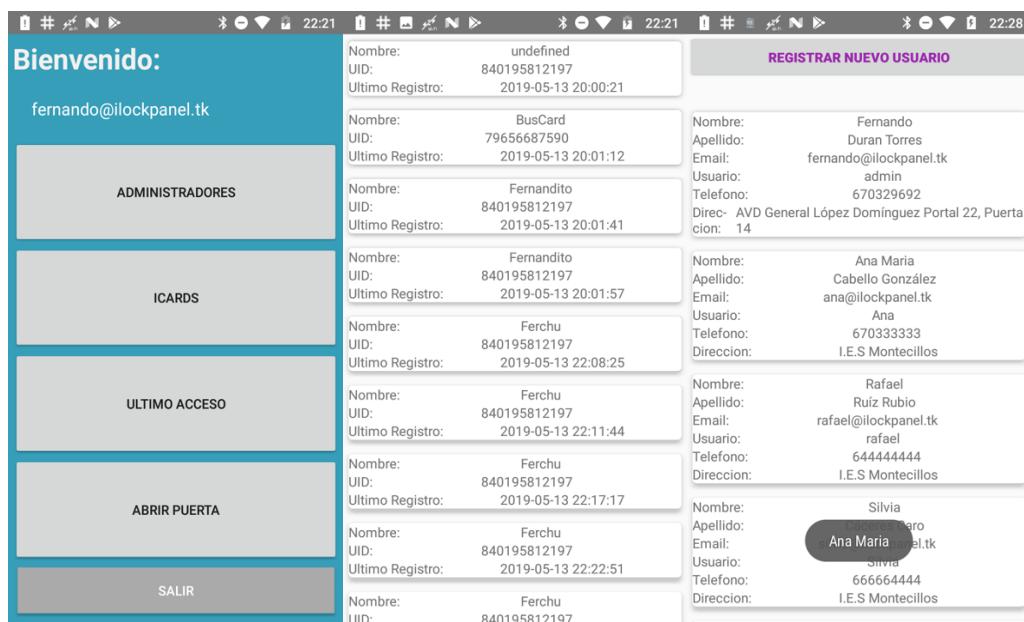
```
<activity android:name=".DashboardActivity"/>
<activity android:name=".RegistrarActivity"/>
<activity android:name=".icardview.ActivityCard"/>
<activity android:name=".adminView.AdminActivity"/>
<activity android:name=".ultimoAccesoView.UltimoActivity"/>
```



Para hacer funcionar el Recycler View me hizo falta tener un adaptador. Para alimentar todos sus datos a la lista, debe ampliar la RecyclerView.Adapterclase. Este objeto crea vistas para los elementos y reemplaza el contenido de algunas de las vistas con nuevos elementos de datos cuando el elemento original ya no es visible.

El administrador de diseño llama al onCreateViewHolder(). Ese método necesita construir RecyclerView.ViewHolder. El tipo de ViewHolder debe coincidir con el tipo declarado en la firma de la clase del Adaptador.

Dentro de un paquete tengo 3 archivos que son la clase que arranca la actividad y recibe los datos de la consulta mediante json y php. Este es el mismo que explico anteriormente para la ventana login el data adapter que tiene declarado los String pertinentes con sus get y set. Por último, tenemos el RecyclerView, encargado de traducir y mostrar todos los datos.





## CAPÍTULO 5. PRUEBAS DE FUNCIONAMIENTO.

Es hora de realizar pruebas de funcionamiento. Comenzare por la prueba de **registro de una nueva tarjeta**. En este caso realizare 3 pruebas de funcionamiento.

1. Tarjeta limpia sin registro.
2. Tarjeta registrada, el software debería detectar que esta ya activa en el sistema. Me preguntara si quiero registrar una nueva. En este caso diré que no para que el software termine.
3. Mismo caso con el anterior, pero realizare la funcionalidad de registro, haciendo que borre sus datos.

Arranco el software **RegistroCard.py**, este lo primero que hará es indicarme que ponga la tarjeta en el lector NFC. Pondré una tarjeta limpia, el programa responde de la siguiente manera:

Terminal	Pantalla LCD
<pre>##### Ponga la tarjeta sobre el lector ##### El codigo leido es: 316238160840  Tarjeta Vacia  Nombre para la iCard: Fernando Escritura Realizada Los datos son: Codigo: 316238160840 Nombre: Fernando Se insertaron de forma correcta  ##### Se recomienda ir a ilockpanel.tk para añadir una foto al perfil #####  Quieres activar la tarjeta ahora(s/n): s Tarjeta Activada, Esta Lista para usarse</pre>	 <p>Ponga la tarjeta sobre el lector</p> <p>Codigo: 584189249134</p> <p>Tarjeta Vacia:</p> <p>Nombre para iCard</p> <p>Nombre: Fernando</p> <p>Datos insertados En la Base Datos</p> <p>Se recomienda ir a ilockpanel.tk</p> <p>✓Activar Ahora? (s/n)</p> <p>Tarjeta Activada</p>

Led **verde** Activo, posteriormente en **azul** tras insertar los datos.

Reviso en la base datos si se inserto esta nueva tarjeta, lo consulto conectándome mediante phpmyadmin que me ofrece mi hosting:

<input type="checkbox"/>		Editar		Copiar		Borrar	12	Fernando	316238160840	2019-06-13 23:10:50		1
--------------------------	--	--------	--	--------	--	--------	----	----------	--------------	---------------------	--	---

Llegado a este punto doy la primera prueba como satisfactoria.

Realizo la prueba con una tarjeta ya registrada. En este caso al realizar la query detecta que el UID leído esta registrado en la base datos. Se encienda el led de color rojo. El programa esta diseñado para prevenir un fallo humano al intentar registrar una nueva tarjeta ya en uso.

```
#####
Ponga la tarjeta sobre el lector
#####
El codigo leido es: 316238160840
Tarjeta ya registrada
¿Quieres volver a registrar esta tarjeta de nuevo?(s/n): r
adios!
```

Llegado a este punto doy la primera prueba como satisfactoria.

Tercera y ultima prueba de este software. Voy a intentar registrar de nuevo esta tarjeta. Cuando me pregunte si la quiere registrar de nuevo diré que si. Este reaccionara borrando los datos relacionado con este UID.

Como explique en el **capítulo 4**, este hace un delete en la tabla personal.

```
#####
Ponga la tarjeta sobre el lector
#####
El codigo leido es: 316238160840
Tarjeta ya registrada
¿Quieres volver a registrar esta tarjeta de nuevo?(s/n): s
Datos borrados, esta lista para ser registrada

Nombre para la iCard: Rocio
Escritura Realizada
Los datos son:
Codigo: 316238160840
Nombre: Rocio
Se insertaron de forma correcta
#####
Se recomienda ir a ilockpanel.tk para añadir una foto al perfil
#####
Quieres activar la tarjeta ahora(s/n): s
Tarjeta Activada, Esta Lista para usarse
```

Continuando con el software, me preguntara por el nuevo nombre a registrar.

Volverá a informarme que acuda a ilockpanel.tk para añadir una foto. Me pregunta si quiero activarla y diré que si. Llegado a este punto la tarjeta esta perfectamente lista para usar.

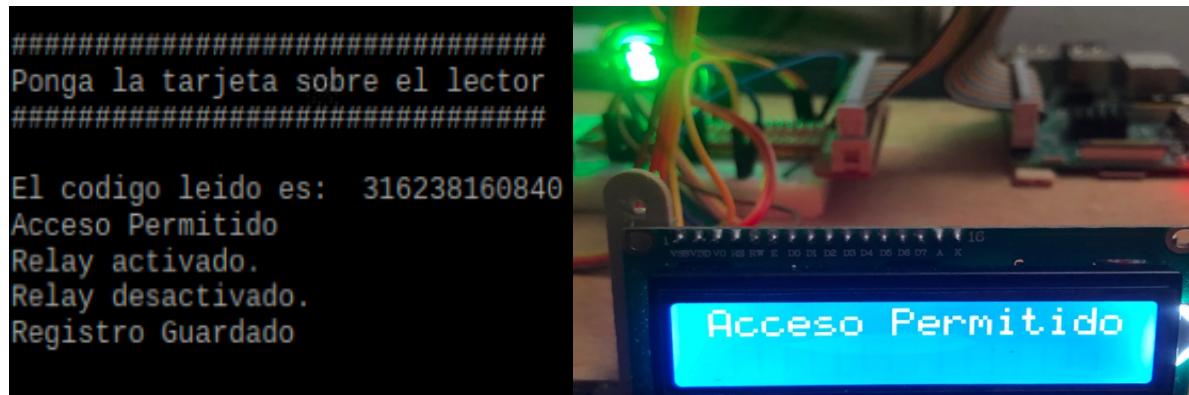
Reviso la base datos para comprobar que efectivamente los cambios se realizaron.

<b>id</b>	<b>nombre</b>	<b>uid</b>	<b>fechaRegistro</b>	<b>foto</b>	<b>activo</b>
6	Nando	840195812197	2019-05-13 22:05:23	upload/17906547505cdbba9d4d0f2.jpg	0
7	BusCoin	79656687590	2019-05-17 10:07:15	upload/19880059635cde7e5c563dd.jpeg	1
9	Ana	710565407360	2019-05-17 11:21:00		1
11	kola	584187271314	2019-05-30 11:13:16		0
14	Rocio	316238160840	2019-06-14 00:20:00		1

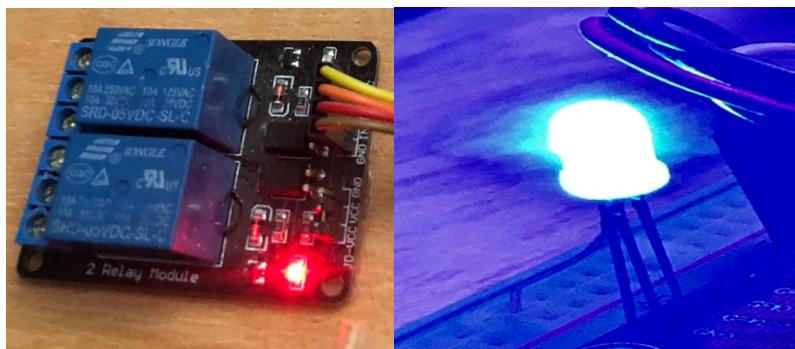
Llegado a este punto doy la primera prueba como satisfactoria. 

Toca probar el software **AccesoCard.py**. Como antes hice, realizare diferentes pruebas con el fin de demostrar que funciona de forma correcta.

1. Pasare por el lector una tarjeta ya registrada, activada y funcional.
2. Esta prueba sera con una tarjeta registrada pero no activada. Se espera que el software responda denegando el acceso.
3. Esta tarjeta no esta registrada en la base datos. Es ajena al sistema. Se espera que el software responda denegando el acceso y ademas apagando la LCD.



Este responde encendiendo el LED verde y activando el relé. En la imagen se puede ver color rojo que eso indica que esta dejando pasar la corriente. Lo desactiva al paso de unos segundos y registra quien fue la ultima persona que accedió al sistema encendiendo el LED de color azul.

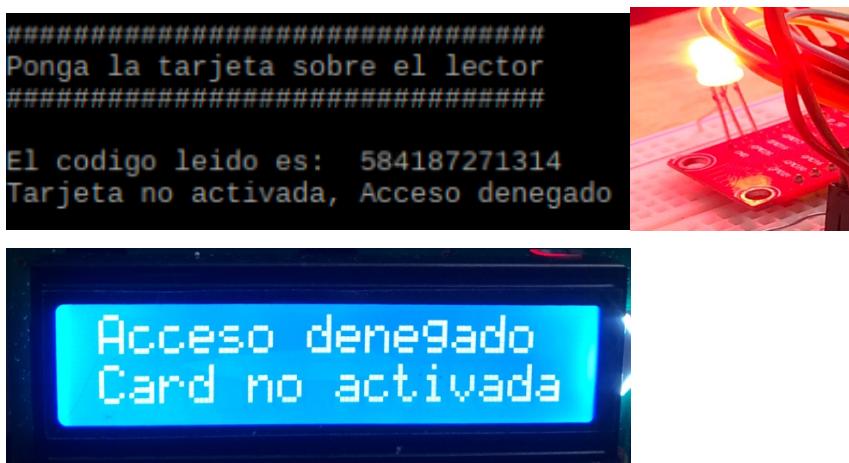


En la base datos se a tenido que registrar el acceso como si fuera un log. Se queda registrado, nombre, UID de la tarjeta y la fecha de entrada.

<b>id</b>	<b>nombre</b>	<b>uid</b>	<b>fechaEntrada</b>
22	Ana	710565407360	2019-05-29 22:39:16
23	BusCoin	79656687590	2019-05-30 10:51:49
24	Ana	710565407360	2019-05-30 10:52:20
25	Fernando	840195812197	2019-05-30 10:54:00
26	Fernando	840195812197	2019-05-30 10:54:32
27	Rocio	316238160840	2019-06-14 00:51:27
28	Rocio	316238160840	2019-06-14 00:52:00
29	Rocio	316238160840	2019-06-14 00:52:28

Llegado a este punto doy la primera prueba como satisfactoria.

La siguiente prueba que realizare será pasar una tarjeta que esta registrada en el sistema, pero no esta activada.



Llegado a este punto doy la primera prueba como satisfactoria.

Toca el turno a una tarjeta no registrada en la base datos. El programa responde de forma

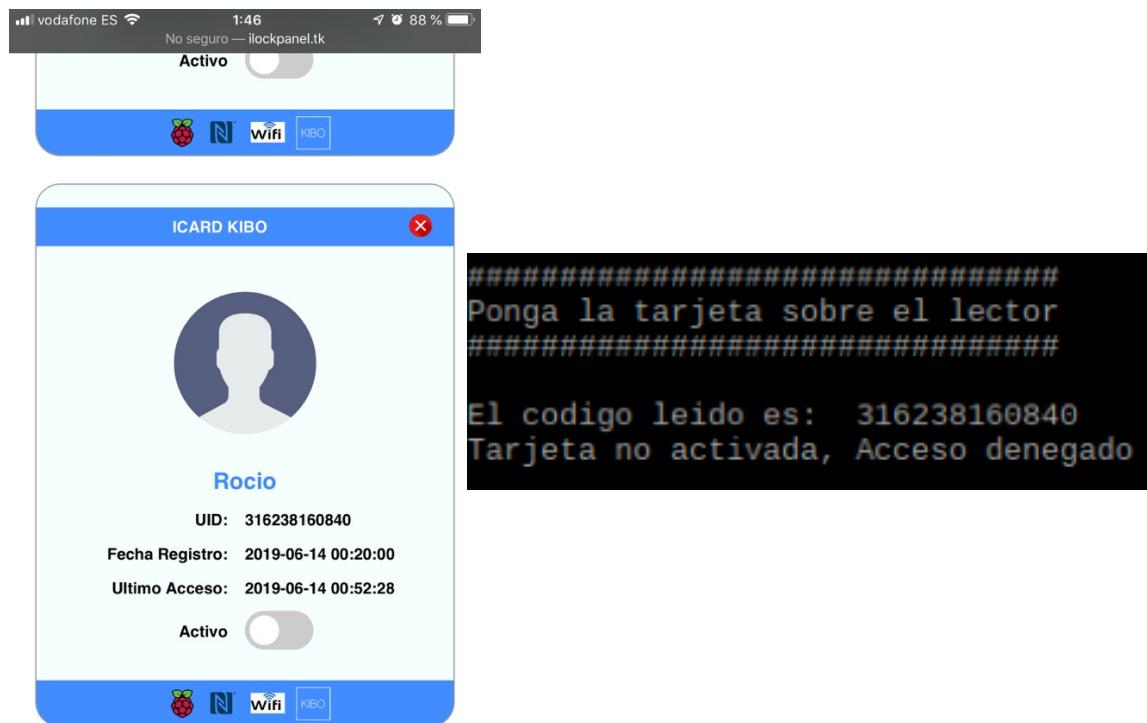
correcta. No permitiendo el acceso, apagando la LCD y encendiendo el led en color rojo.



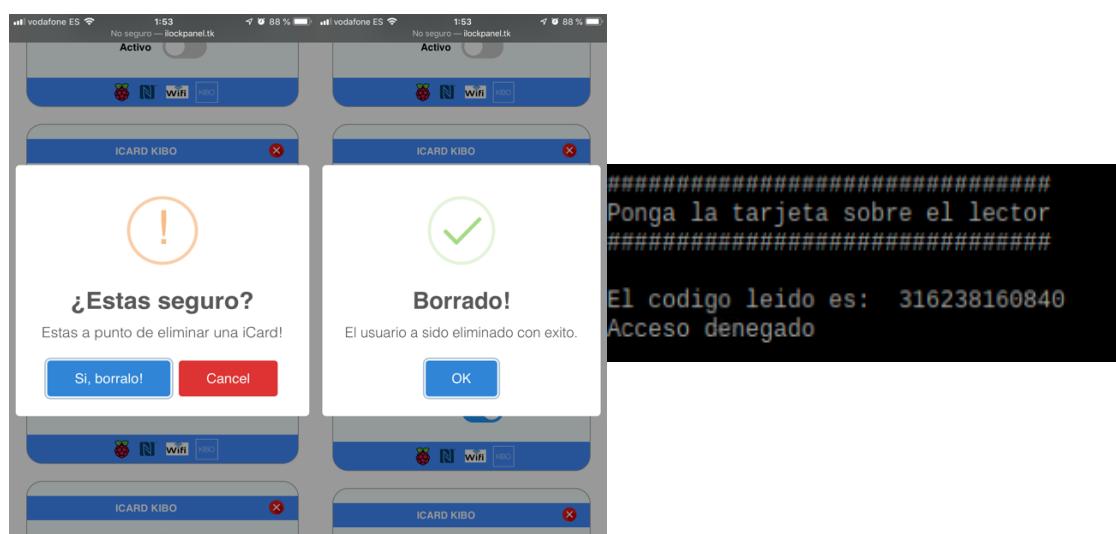
Llegado a este punto doy la primera prueba como satisfactoria. ✓

## iLockPi

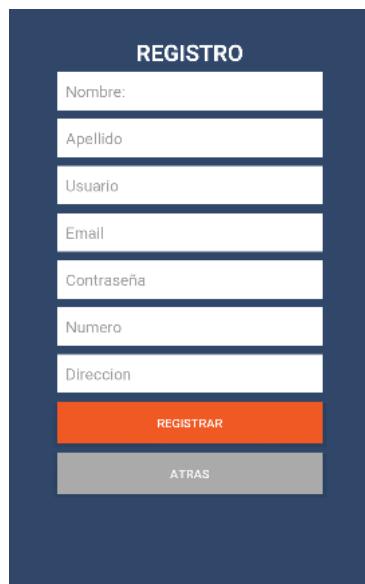
Con la plataforma ilockpanel.tk podemos desactivar cualquier tarjeta en cualquier momento. En este caso voy a desactivar a Roció.



Desde la plataforma también puedo eliminar a una tarjeta. Pulsando en la **X** en la parte superior derecha. En este caso eliminare a Roció y volveré a intentar acceder al sistema de nuevo. El sistema advierte que estas a punto de eliminar la tarjeta. Si aceptamos se borrará.



Desde la app en Android podemos visualizar la misma información que por el panel. Esta esta programada en código nativo java. No se puede editar, solo visualizar el contenido. Pero desde del botón administradores, podemos registrar a un nuevo usuario administrador.





## CAPÍTULO 6. PROBLEMAS ENCONTRADOS.

Principalmente el problema que me topado fue de software. Dominar el lenguaje Python fue algo complejo. Este lenguaje no tiene punto y coma, también es muy delicado con las tabulaciones. Compilaba el software y me daba errores relacionado con la tabulación. Este fue solucionado dirigiéndome a **Format/Tabify Región** en el IDE de Python. Dentro de este menú ponemos los espaciados a 8. Esto soluciona los problemas de tabulaciones.

En la consulta de MYSQL no sabia como solventar la respuesta que devolvía la query. Mi solución fue igualar el resultado del cursor de esta forma. if filas == []. Esto me funciono.

También debido a problemas de copiado de un bloque a otro arrastré elementos con errores de sintaxis. Aprendí a fijarme tranquilamente en el código y depurar para solventar estos fallos.

Al principio de empezar el proyecto me encontré que la librería inicial que estaba usando para la RFID 522 no era compatible con la versión de Raspbian que tenia. Uso la versión 4.14 del kernel de Linux, en concreto la versión lanzada en abril del 2019. La solución fue buscar una librería actualizada y compatible con Python 3.

```
# Librerías
import RPi.GPIO as GPIO
import time
import lcddriver

#Inicializar
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(20, GPIO.OUT)      # Relé
GPIO.setup(16, GPIO.OUT)      # Relé
lcd = lcddriver.lcd()

# GPIO PARA LOS LED
def reléApaOFF():

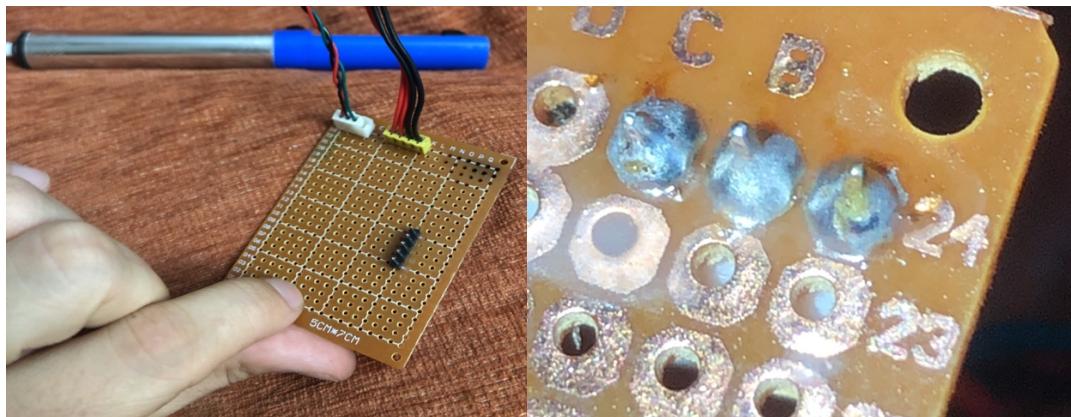
    GPIO.output(20,True)
    GPIO.output(16,True)
    lcd.lcd_clear()
    lcd.lcd_display_string("Reles Apagado",1)
    time.sleep(6)
    lcd.lcd_clear()

if __name__ == "__main__":
    reléApaOFF()
```

Cuando por fin estaba todo funcionando el relé no trabajo de forma correcta. Este trabajaba de forma inversa, al arrancar por primera vez el programa este disparaba el relé permitiendo el acceso sin el uso de la tarjeta. Esto era debido al principio del programa activaba los puertos GPIO de la Raspberry Pi para la misma RFID y otros elementos como los leds. La solución fue crear un pequeño script en Python que su

principal función es reiniciar este estado.

Al intentar crear el circuito eléctrico con una placa ya perforada, el estaño se unía haciendo corto circuito un elemento con otro. Mi solución fue crear una PCB mediante un software informático llamado EAGLE y llevarla a cabo mediante el método del planchado como mostré en el **Capítulo 4**



## CAPÍTULO 7. MEJORAS PROPUESTAS.

Este proyecto es inmenso, debido a la complejidad que lleva y a la gran curva de aprendizaje que requiere alguna mejora.

1. Una de las principales mejoras es la fluidez del software que realiza la apertura de la puerta. Este debido a los delay de los LEDS retrasa el accionamiento del relé.
2. Una interfaz grafica en el software de registro o en el software de acceso podría hacer al usuario entender mejor el sistema. La interfaz grafica puede ser también en el entorno Python.
3. Esta mejora fue algo soñada y planteada en un principio. La posibilidad de desbloquear la puerta mediante un dispositivo móvil. Si el dispositivo tiene chip NFC sería perfecto que se usara como método de desbloqueo, incluso como método de registro de nuevas tarjetas.
4. Implementar las mismas funcionalidades de la web en la app de Android con código nativo.
5. Creación de una app para IOS.
6. Mejora en la PCB o cableado. Haciéndola más profesional y no tan casera.
7. Añadir nuevos sensores como por ejemplo uno de temperatura. Esta mejora sería para el estudio del centro laboral. Mediría la temperatura exterior y la interior de la puerta.
8. Sensor proximidad detecta cuando hay una persona delante del sistema de desbloqueo, esto haría que la vida útil del led que incorpora la LCD durase un poco mas.
9. Reconocimiento mediante lector de huellas. Este sistema trabajaría de forma paralela a la tarjeta. Se podría elegir entre los sistemas de autenticación.
10. Sonido al desbloqueo de la puerta. La idea inicial fue añadir un sistema de sonido que al desbloquear la LCD se pudiera sentir mediante el oído.



## CAPÍTULO 8. CONCLUSIONES.

Con este proyecto quería demostrar que un mini pc llamado Raspberry y 4 componentes comprado en una tienda online china se podría llegar a crear un sistema de autentificación por NFC. Con este proyecto queda demostrado de que es posible esto.

Llegue a aprender Python con algo de soltura, PHP y mejore mis conocimientos sobre Raspberry pi. También conseguir aprender a diseñar y crear una PCB. Esta fue una asignatura para mi dentro del mundo de la electrónica.

Este proyecto a sido una afición mas que un trabajo mas, disfrute con el. Me encanto ver como evolucionaba poco a poco consiguiendo mejoras y avances importantes.

iLockPi puede ser usado en la vida real, la versión creada por mi es bastante estable. Es un sistema barato y fácil de crear. También se podría adaptar para en una empres los trabajadores fichen a la hora de entrada y salida de su puesto de trabajo.



## CAPÍTULO 9. REFERENCIA Y BIBLIOGRAFÍA.

**Duran torres, Fernando (Autor) – [2019] – iLockPi, Sistema de autentificación mediante NFC.**

**Sistema operativo Raspberry:**

<https://www.raspberrypi.org/downloads/raspbian/>

<https://www.raspberrypi.org/forums/>

**Hosting:**

<https://www.hostinger.es/>

**Dominio:**

<https://www.freenom.com/es>

**Framework webs:**

<https://www.codeigniter.com/>

<https://getbootstrap.com/>

**RFID 522:**

<https://blog.bricogeek.com/noticias/diy/cerradura-electronica-rfid-controlada-con-raspberry-pi/>

<https://www.instructables.com/id/Attendance-system-using-Raspberry-Pi-and-NFC-Tag-r/>

<https://raspmer.blogspot.com/2015/07/how-to-use-rfid-rc522-on-raspbian.html>

<https://www.teachmemicro.com/rfid-login-raspberry-pi/>

<https://pimylifeup.com/raspberry-pi-rfid-rc522/>

<https://stackoverflow.com/questions/54847012/raspberry-pi-3b-and-rfid-rc522-python-typeerror/55024642#55024642>

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

**LCD:**

<http://www.circuitbasics.com/raspberry-pi-i2c-lcd-set-up-and-programming/>

<http://www.multicopterx.es/configurar-un-lcd-16x2-por-i2c-en-la-raspberry-pi/>

**Android:**

<https://androidjson.com/android-json-parsing-recyclerview-image-text/>

<https://androidjson.com/android-server-login-registration-php-mysql/>

**Librerías:**

<https://github.com/Sacredgamer/RFID-RC552-Interface>  
[https://github.com/simonmonk/clever\\_card\\_kit](https://github.com/simonmonk/clever_card_kit)  
<https://gist.github.com/DenisFromHR/cc863375a6e19dce359d>

**Diseño PCB:**

<https://www.autodesk.com/products/eagle/overview>

**Método del planchado para crear la PCB;**

<https://www.youtube.com/watch?v=sMFswuSh4FY&t=476s>