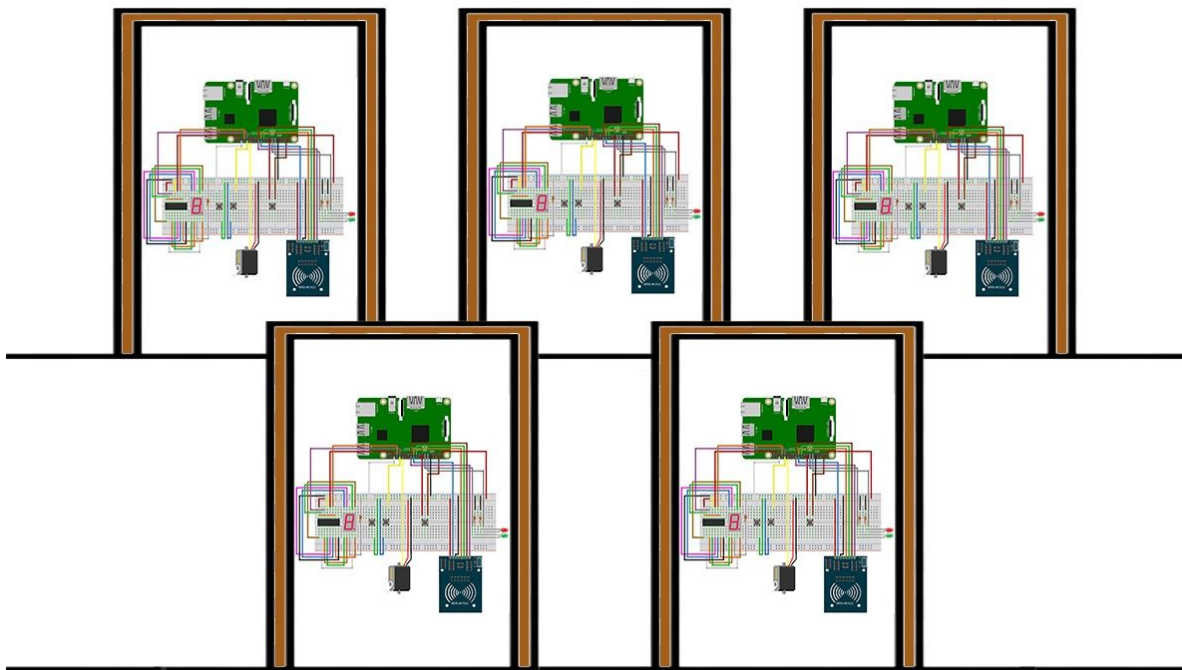
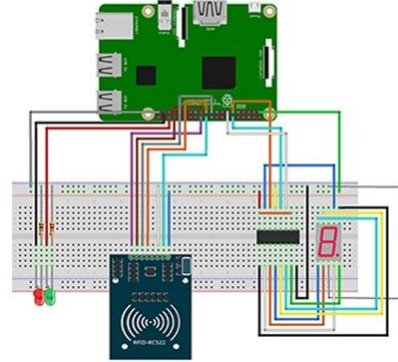


# SUALPA'S-PI



Autor: Jose Miguel Sualdea Serrano - Alba Pasamontes Obispo  
Tutor TFG: Carlos Contreras  
ASIR (Administración de Sistemas informáticos en RED)  
Informática

# Índice

<b>1. Introducción</b>	<b>3</b>
<b>2. Servicios: Instalación y utilidad</b>	<b>4</b>
Apache	4
PHP	4
Python	4
MariaDB	4
phpMyAdmin	5
LXTerminal	5
Suricata	5
<b>3. Bases de datos: Contenido</b>	<b>6</b>
<b>4. Administración Web</b>	<b>10</b>
Inicio de sesión	10
Índice y menú global	10
Insertar	11
Listar	13
Buscar	13
Borrar	13
Actualizar	14
Historial	15
Cierre de sesión	15
<b>5. Autenticación en tres pasos generalizado</b>	<b>17</b>
Autenticación 1 - Algo que tu tienes	17
Cómo configurar un chip RFID RC522 Raspberry:	17
Autenticación 2 - Algo que tu eres	23
Autenticación 3 - Algo que tu sabes	23
<b>6. Autenticador.py</b>	<b>26</b>
<b>7. Suricata en nuestro proyecto</b>	<b>27</b>
<b>8. Bibliografía</b>	<b>28</b>
<b>ANEXO I - Administración Web</b>	<b>30</b>
registro_entrar.html	30
index.php	30
insertar1.php	33
copiar.php	33
insertar.php	34
insertar3.php	34

listar.php	36
buscar.php	37
buscar2.php	37
actualizar.php	39
actualizar2.php	40
historial.php	43
general.php	44
principal.php	45
admin.php	46
rrhh.php	46
imasd.php	47
sala.php	48
denegados.php	49
<b>ANEXO II - Lector de ID</b>	<b>50</b>
<b>ANEXO III - Biometría</b>	<b>54</b>
Registro.py	56
Guardado.py	59
Identificar.py	61
<b>ANEXO IV - Contraseña numérica</b>	<b>64</b>
<b>ANEXO V - Autenticador en tres pasos</b>	<b>68</b>
<b>ANEXO VI - Suricata</b>	<b>82</b>
Análisis de la herramienta Suricata	82
Fases en la gestión de amenazas	82
Características	82
Estructura de las reglas	83
Instalación	83

# 1. Introducción

Desde tiempos inmemoriales, las personas hemos buscado la seguridad para proteger lo que más nos importa. En la antigüedad, las ciudades construían grandes murallas a su alrededor para proteger así tanto a sus habitantes como los recursos que producían, del resto de ciudades cuyo fin era enriquecerse a su costa. Cada ciudad tenía sus propios sistemas de defensa: unas construían sus ciudades en el alto de un monte, otras se rodeaban de un extenso foso, otras montaban una guardia permanente para defenderse de los atacantes y, las más seguras, tenían los tres.

Hoy en día, los que poseen estos recursos en forma de datos e información son las empresas. Estos datos necesitan ser protegidos adoptando un conjunto de medidas tanto preventivas como reactivas ante los ataques externos para mantener su confidencialidad, disponibilidad e integridad y evitar de esta forma que puedan ser robados, divulgados, mal utilizados, borrados e incluso sabotados pues hoy en día, la información es poder.

Para proteger esta información, las organizaciones necesitan adoptar un conjunto mínimo de controles de seguridad, aunque ya se sabe que toda seguridad es poca. El primer paso a controlar en una empresa es quien entra y sale de ella, además de donde accede una vez que está dentro. Para llevar a cabo este control, se utilizan distintos métodos de autenticación. La autenticación es el acto o proceso de confirmar que algo o alguien es quien dice ser. A la parte que se identifica se le llama probador. A la parte que verifica la identidad se la llama verificador. En el caso de las organizaciones, el probador es el usuario o trabajador que quiere acceder a ciertos recursos de la empresa y el verificador es un sistema tecnológico que protege el acceso a dichos recursos y se encarga de verificar que quien pretende entrar es un usuario que tiene permisos para acceder a tales recursos.

El objetivo de este proyecto es desarrollar un sistema de autenticación de máxima seguridad. Al igual que en la antigüedad, el sistema más seguro es el que conlleva más requisitos para acceder a sus datos y por ello implementaremos la llamada “autenticación en tres pasos”, el método más seguro a día de hoy. Este sistema de autenticación será acompañado de un un servicio de administración intuitivo a la par que avanzado.

## 2. Servicios: Instalación y utilidad

### **Apache**

Es un servidor web HTTP de código abierto compatible con diversos sistemas operativos como Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh... entre otros.

Está diseñado para transferir datos de hipertexto, es decir, páginas web con todos sus elementos (textos, widgets, banners, etc).

```
sudo apt-get install apache2 -y
```

### **PHP**

Acrónimo recursivo: Hypertext Preprocessor. Es un lenguaje de código abierto y multiplataforma para el desarrollo web, y que puede ser incrustado en HTML.

```
sudo apt-get install php -y
```

### **Python**

Es un lenguaje de scripting independiente de plataforma y orientado a objetos, preparado para realizar cualquier tipo de programa, desde aplicaciones Windows a servidores de red o incluso, páginas web. Es un lenguaje interpretado, lo que ofrece ventajas como la rapidez de desarrollo e inconvenientes como una menor velocidad.

```
sudo apt-get install python3-dev python3-pip
```

### **MariaDB**

Es un es un sistema gestor de bases de datos, de software libre, y además, una variante de MySQL (por lo que mantiene compatibilidad con este). Es compatible con sistemas operativos como Unix, Windows, Solaris, Linux, OS X... MariaDB tiene un mejor rendimiento y permite optimizar mejor las bases de datos.

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

Para configurar el acceso remoto a bases de datos MySQL/MariaDB debemos configurar el archivo 50-server.cnf comentando la línea: **bind-address = 127.0.0.1**

```
sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf
```

Para guardar los cambios reiniciamos el servicio:

```
sudo service mariadb restart
```

## phpMyAdmin

Es una herramienta que nos permite dirigir y administrar nuestras bases de datos MySQL/MariaDB. Se pueden crear, eliminar, modificar bases de datos así como gestionar las tablas de una manera más sencilla y visual a la proporcionada por el terminal.

```
sudo apt-get install phpmyadmin
```

Para habilitar phpmyadmin incluimos la línea: **Include /etc/phpmyadmin/apache.conf** en el archivo de configuración de apache:

```
sudo nano /etc/apache2/apache2.conf
```

Para guardar los cambios reiniciamos el servicio:

```
sudo service apache2 restart
```

## LXTerminal

Es el emulador de terminal estándar de LXDE. Por explicarlo de una manera más sencilla, es el programa que se nos abre al iniciar la consola de la Raspberry. Viene preinstalado en el sistema de Raspbian, y consta de un archivo de configuración que nos permitirá programar tareas. Otro método comúnmente utilizado es el administrador regular de procesos en segundo plano crontab. Para editar el programador de tareas de LXterminal escribiremos en la consola el siguiente comando:

```
sudo nano /etc/xdg/lxsession/LXDE-pi/autostart
```

Para que nos ejecute nuestro programa lectorid.py, añadiremos las siguientes líneas al archivo:

```
point-rpi
@lxterminal
@/usr/bin/python3 /home/pi/Desktop/lectorid.py
```

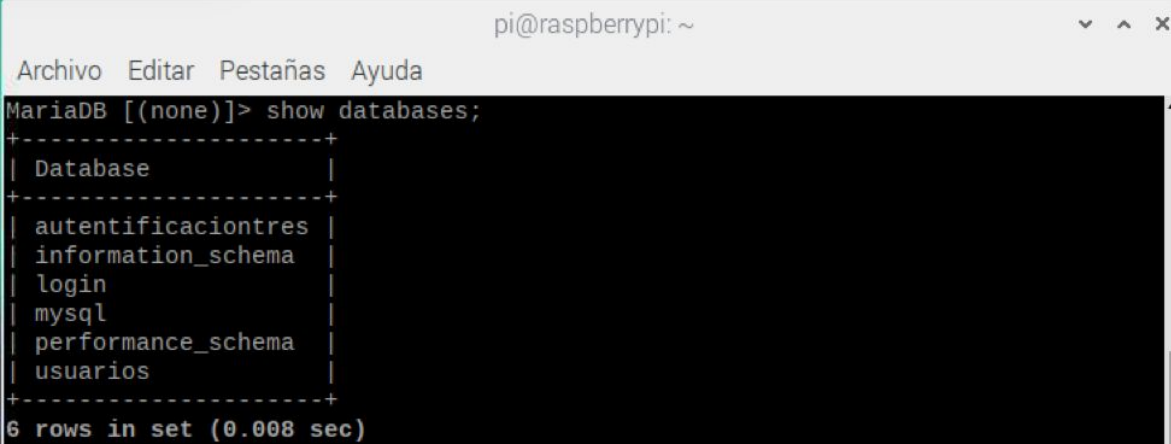
## Suricata

Es un motor de red de alto rendimiento IDS (también puede actuar como IPS) y es una aplicación de código abierto multiplataforma. Está basado en un conjunto de reglas que se desarrollan para controlar el tráfico de red y aportar así alertas al administrador cuando se detecten eventos sospechosos.

```
sudo apt-get install suricata
```

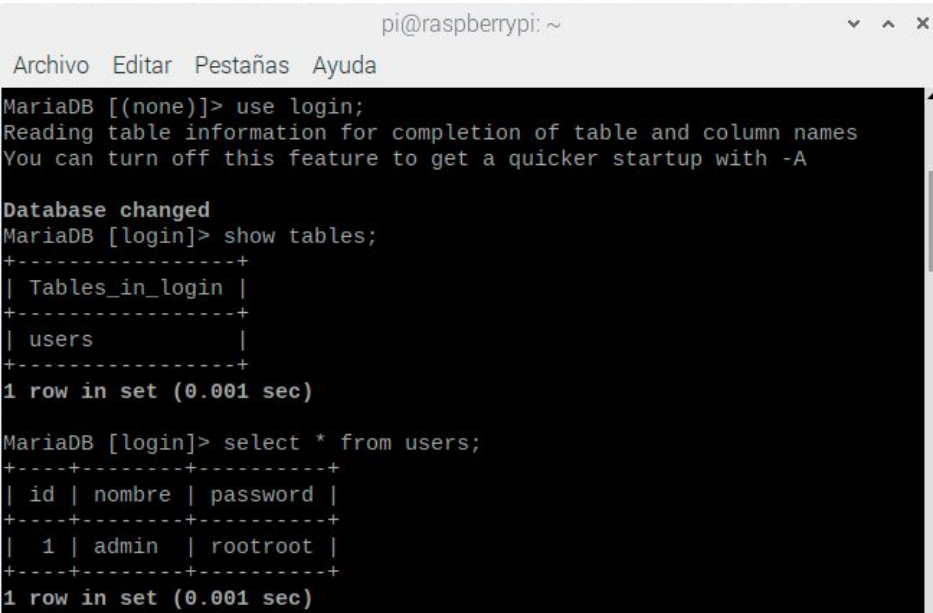
### 3. Bases de datos: Contenido

Una base de datos es una colección de información organizada de forma que un programa de ordenador pueda seleccionar rápidamente los fragmentos de datos que necesite. Una base de datos es un sistema de archivos electrónico. Nuestro proyecto consta de tres bases de datos: login, usuarios y autentificaciontres, las cuales son independientes unas de otras y tienen unos objetivos específicos.



```
pi@raspberrypi: ~
Archivo  Editar  Pestañas  Ayuda
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| autentificaciontres |
| information_schema |
| login |
| mysql |
| performance_schema |
| usuarios |
+-----+
6 rows in set (0.008 sec)
```

La primera base de datos “login” está directamente relacionada con la administración del sistema. En ella se almacena el nombre de usuario y la contraseña de todos aquellos que tienen permiso para insertar, modificar y eliminar información en el resto de las bases de datos.



```
pi@raspberrypi: ~
Archivo  Editar  Pestañas  Ayuda
MariaDB [(none)]> use login;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [login]> show tables;
+-----+
| Tables_in_login |
+-----+
| users |
+-----+
1 row in set (0.001 sec)

MariaDB [login]> select * from users;
+----+-----+-----+
| id | nombre | password |
+----+-----+-----+
| 1 | admin | rootroot |
+----+-----+-----+
1 row in set (0.001 sec)
```

La segunda base de datos, “usuarios”, se relaciona con la información personal de los usuarios registrados y el historial de entradas correspondientes a cada puerta. Esta se divide en diferentes tablas:

```

pi@raspberrypi: ~
Archivo Editar Pestañas Ayuda
MariaDB [(none)]> use usuarios
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [usuarios]> show tables;
+-----+
| Tables_in_usuarios |
+-----+
| admitidos           |
| historial           |
| historial_a         |
| historial_i         |
| historial_p         |
| historial_r         |
| historial_s         |
| historial_x         |
+-----+
8 rows in set (0.001 sec)

```

Por un lado, en la tabla admitidos podemos encontrar la información personal recogida de cada usuario: el id personal (cada usuario posee una llave RFID con un id exclusivo), el nombre, el apellido, el dni y su correo electrónico, además de las puertas a las que tienen acceso (marcadas con un 1) y las que tienen restringidas (marcadas con un 0).

```

pi@raspberrypi: ~
Archivo Editar Pestañas Ayuda
MariaDB [usuarios]> select * from admitidos;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| numero_de_usuario | id | nombre | apellido | dni | correo | puerta_principal | dpto_administracion | dpto_rrhh | dpto_imasd | sala_de_reuniones |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | 1 | admin | Miguel | 123456789 | arr@pun.com | 0 | 0 | 0 | 0 | 0 |
| 8 | 2 | samuel | reides | 222222222 | samu@el | 0 | 0 | 0 | 0 | 0 |
| 9 | 584197271971 | Jose | Miguel | 123456779 | arr@pun.com | 0 | 0 | 0 | 0 | 0 |
| 10 | 584197271971 | Alba | pasa | 94249424A | al@ba | 0 | 0 | 0 | 0 | 0 |
| 13 | 123456789123 | prueba | puerta | 66666666B | a@.com | 1 | 0 | 1 | 0 | 1 |
| 14 | 584197271971 | pepito | pepon | 99999999P | arr@pun.com | 1 | 1 | 1 | 1 | 1 |
| 33 | 444444444444 | pruebaerrores2 | dossss | 21212121P | s@l.c | 1 | 0 | 1 | 0 | 0 |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
7 rows in set (0.000 sec)

```

Por otro lado, están las tablas de los historiales de acceso. La tabla historial es una tabla global en la que se guardan los accesos a cualquiera de las puertas. Para una información más detallada, están el resto de tablas que nos indican mediante la columna de “numero\_de\_usuario” el id de la tarjeta RFID con el que se ha entrado, seguido de la columna “fecha\_hora” que nos indicará como bien indica su propio nombre, la fecha y hora del acceso.

Si queremos acceder a la información privada del usuario que ha entrado en cualquiera de las puertas, solo tendríamos que enfrentar las tablas con la siguiente consulta:

```

select * from admitidos, historial where admitidos.id=historial.numero_de_usuario;

```



```

pi@raspberrypi:~
Archivo Editar Pestañas Ayuda
MariaDB [usuarios]> select * from historial;
+-----+-----+-----+
| id | numero_de_usuario | fecha_hora |
+-----+-----+-----+
| 1 | 4 | 0000-00-00 00:00:00 |
| 2 | 4 | 0000-00-00 00:00:00 |
| 5 | 584197271971 | 0000-00-00 00:00:00 |
| 6 | 584197271971 | 2020-03-09 23:23:52 |
| 8 | 584197271971 | 2020-03-09 23:28:10 |
| 9 | 584197271971 | 2020-03-09 23:29:23 |
| 10 | 584197271971 | 2020-03-09 23:30:20 |
| 11 | 584197271971 | 2020-03-09 23:33:26 |
| 12 | 584197271971 | 2020-03-09 23:35:05 |
| 13 | 584197271971 | 2020-03-09 23:36:00 |
| 14 | 592962217132 | 2020-03-11 02:53:32 |
| 15 | 584197271971 | 2020-06-06 17:49:23 |
| 16 | 584197271971 | 2020-06-07 01:21:58 |
+-----+-----+-----+
13 rows in set (0.000 sec)

```

```

pi@raspberrypi:~
Archivo Editar Pestañas Ayuda
MariaDB [usuarios]> select * from historial_a;
+-----+-----+-----+
| id | numero_de_usuario | fecha_hora |
+-----+-----+-----+
| 1 | 584197271911 | 2020-03-11 21:12:30 |
+-----+-----+-----+
1 row in set (0.000 sec)

```

```

pi@raspberrypi:~
Archivo Editar Pestañas Ayuda
MariaDB [usuarios]> select * from historial_i;
+-----+-----+-----+
| id | numero_de_usuario | fecha_hora |
+-----+-----+-----+
| 1 | 584197271965 | 2020-03-11 21:14:40 |
+-----+-----+-----+
1 row in set (0.000 sec)

```

```

pi@raspberrypi:~
Archivo Editar Pestañas Ayuda
MariaDB [usuarios]> select * from historial_p;
+-----+-----+-----+
| id | numero_de_usuario | fecha_hora |
+-----+-----+-----+
| 1 | 584197271922 | 2020-03-11 21:12:54 |
| 2 | 584197271922 | 2020-03-11 21:13:01 |
+-----+-----+-----+
2 rows in set (0.000 sec)

```

```

pi@raspberrypi:~
Archivo Editar Pestañas Ayuda
MariaDB [usuarios]> select * from historial_r;
+-----+-----+-----+
| id | numero_de_usuario | fecha_hora |
+-----+-----+-----+
| 1 | 584197271944 | 2020-03-11 21:14:40 |
+-----+-----+-----+
1 row in set (0.000 sec)

```

```

pi@raspberrypi:~
Archivo Editar Pestañas Ayuda
MariaDB [usuarios]> select * from historial_s;
+-----+-----+-----+
| id | numero_de_usuario | fecha_hora |
+-----+-----+-----+
| 1 | 584197271933 | 2020-03-11 21:13:39 |
+-----+-----+-----+
1 row in set (0.000 sec)

```

```

pi@raspberrypi:~
Archivo Editar Pestañas Ayuda
MariaDB [usuarios]> select * from historial_x;
+-----+-----+-----+
| id | numero_de_usuario | fecha_hora |
+-----+-----+-----+
| 5 | 918597563886 | 2020-06-09 04:46:41 |
| 6 | 584197271971 | 2020-06-09 05:00:38 |
| 7 | 584197271971 | 2020-06-09 05:01:11 |
+-----+-----+-----+
3 rows in set (0.001 sec)

```

Para todas estas tablas de la base de datos en “usuarios”, se ha creado en MariaDB un usuario específico con los permisos de mostrar e insertar. Este usuario es el que está introducido en cada cerradura para que éstas puedan consultar en la tabla admitidos si la persona que desea entrar tiene el acceso permitido y para poder insertar en las tablas de los historiales los nuevos intentos de acceso.

La tercera base de datos “autenteticaciontres” está relacionada con el tercer paso de autenticación de nuestro programa. Esta se encarga de guardar las contraseñas de cada una de las puertas.

```
pi@raspberrypi: ~
Archivo Editar Pestañas Ayuda
MariaDB [(none)]> use autenticaciontres;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [autenticaciontres]> show tables;
+-----+
| Tables_in_autenticaciontres |
+-----+
| puertas                       |
+-----+
1 row in set (0.001 sec)

MariaDB [autenticaciontres]> select * from puertas;
+-----+-----+-----+-----+-----+
| puerta_principal | dpto_administracion | dpto_rrhh | dpto_imasd | sala_de_reuniones |
+-----+-----+-----+-----+-----+
| 1234             | 1111                | 2222     | 3333       | 4444              |
+-----+-----+-----+-----+-----+
1 row in set (0.002 sec)
```

Cada puerta tiene un código de acceso diferente, si el usuario no sabe dicho código, no puede completar nuestras tres fases de autenticación, por lo que se le deniega el acceso a la puerta deseada.

## 4. Administración Web

Una vez encendemos nuestra Raspberry, automáticamente se abre nuestro servidor web con la página de administración del sistema. Para ello hemos editado el archivo de configuración desde la consola con el siguiente comando:

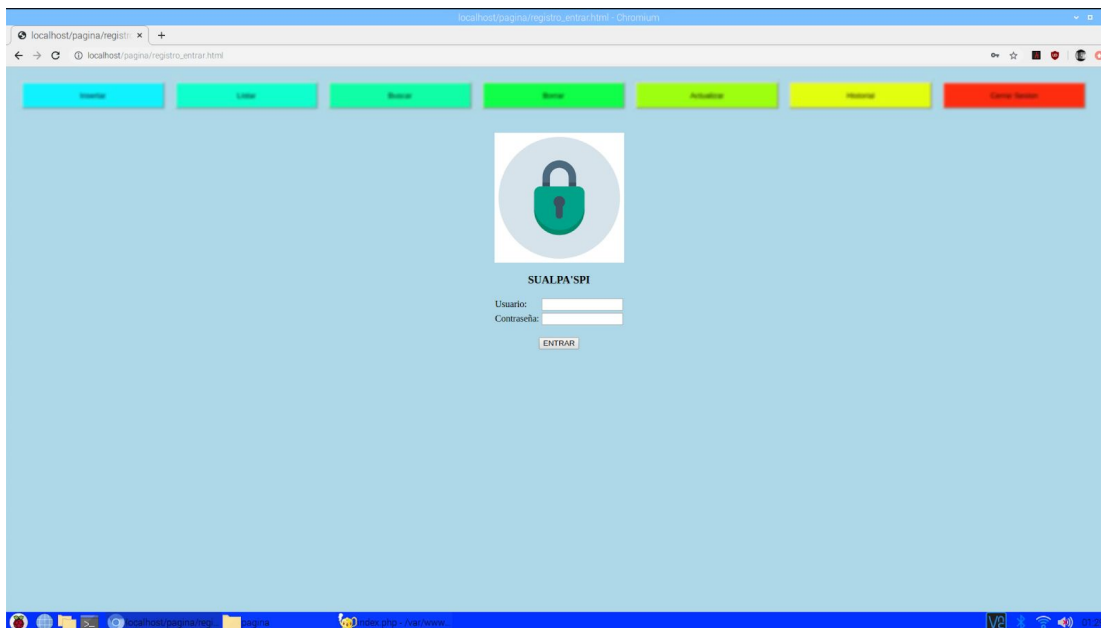
```
sudo nano /etc/xdg/lxsession/LXDE-pi/autostart
```

Y le hemos añadido la siguiente línea:

```
@chromium-browser localhost/pagina/index.php
```

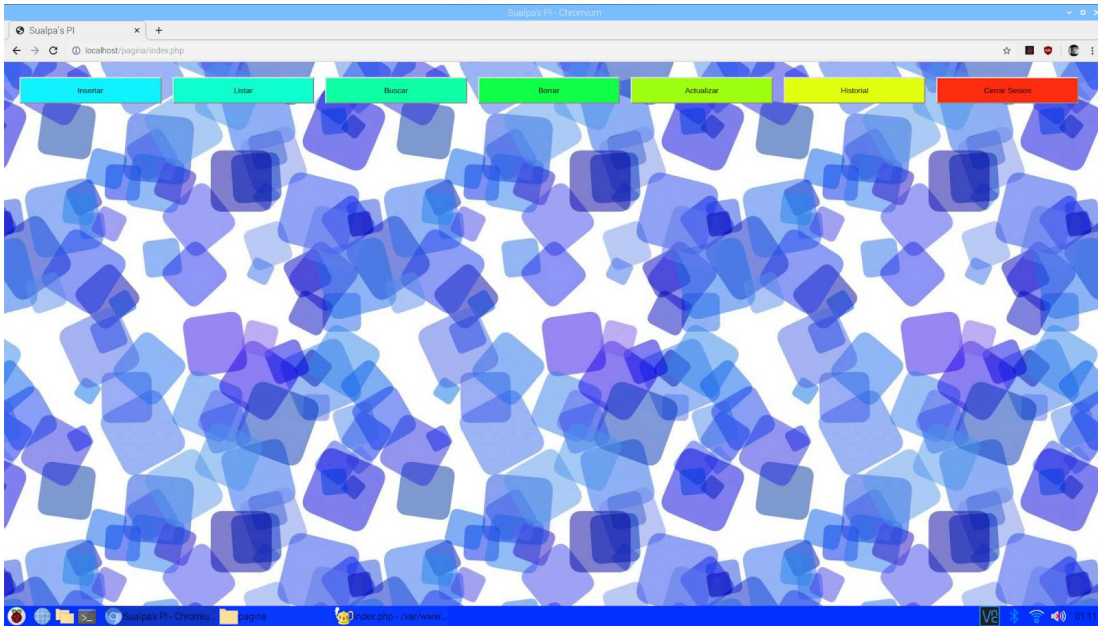
### - Inicio de sesión

La imagen que vemos a continuación, es la página que se nos abre. En ella se pide un nombre de usuario junto con su contraseña para poder pasar a la administración. Esta información se contrasta con la base de datos de “login” en la que aparecen los administradores, y en caso de no estar dentro, seremos devueltos a esta página.



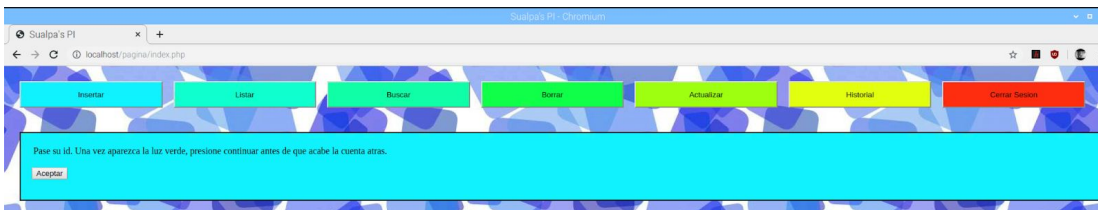
### - Índice y menú global

Una vez estamos dentro de la página, el menú global que nos aparece es el siguiente. En él, vemos que tenemos los permisos de insertar, listar, buscar, borrar y actualizar la base de datos de “usuarios”, además de poder consultar el historial y en caso de finalizar, cerrar sesión.

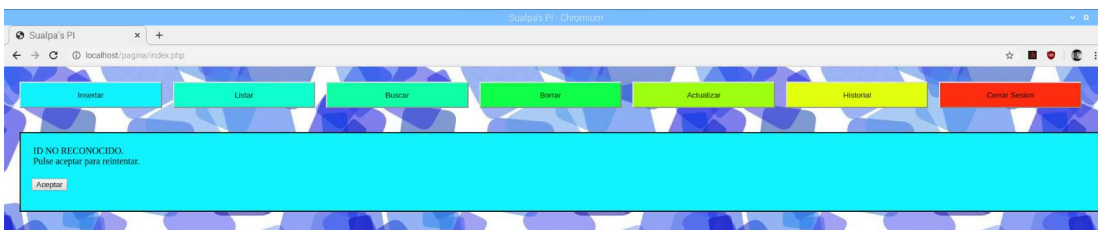


## - Insertar

Cuando pulsamos en insertar, nos carga la siguiente página. En ella se nos pide que pasemos la tarjeta RFID de la persona a la que queremos insertar por el lector de códigos (consultar funcionamiento en [ANEXO II - lectorid.py](#)) Cuando se nos enciende la luz verde, significa que la tarjeta RFID ha sido detectada, y automáticamente comienza una cuenta atrás de borrado en un visualizador de siete segmentos. Esta cuenta es la encargada de apagar la luz verde y de borrar los datos de este ID pasados 10 segundos, para evitar que la siguiente identificación que se haga, recoja un ID detectado anteriormente.



Esto significa que si tardamos demasiado en presionar el aceptar, o si lo intentamos hacer sin pasar ninguna tarjeta, en el siguiente paso nos indicará que ningún ID ha sido recogido por el programa, dejándonos las opción de volverlo a intentar o pulsar en otra seccion del menu.



Si por ende, nosotros presionamos “Aceptar” antes de que la cuenta atrás termine, nos cargara la siguiente página. En ella se nos confirma que la página ha recogido el ID, y que ahora nuestras opciones son copiar el código y continuar con la inserción del usuario, o abandonarla y volver al inicio.



Si le damos a copiar y continuar, nos salta el formulario mostrado a continuación. En él, nos aparece ya marcada la puerta de entrada en los posibles accesos, ya que es de sentido común que si puedes entrar a alguna sala del edificio, antes puedes entrar al edificio, pero se puede desmarcar si las circunstancias lo requieren. Además encontraremos la casilla del ID con éste ya escrito en modo password. Si por alguna razón borramos éste sin querer, solo tendríamos que darle a “pegar” dentro del recuadro, ya que en el apartado anterior, al darle a copiar y continuar, se copió el ID automáticamente en nuestro portapapeles. Una vez que le demos nuevamente a continuar, este ID será borrado del portapapeles.

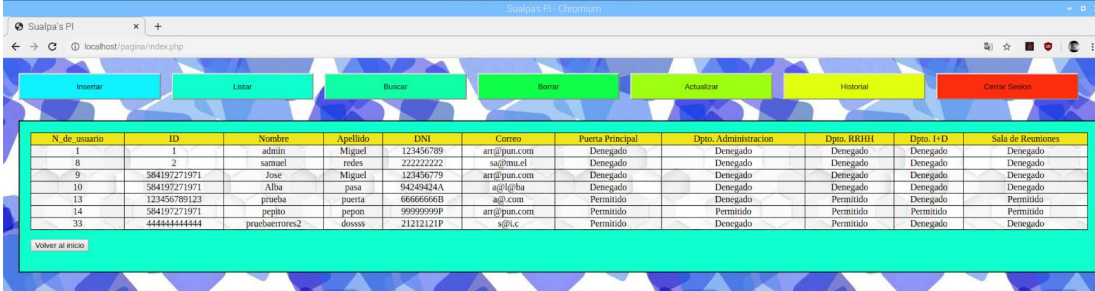


Una vez rellenamos los datos correctamente y le damos a continuar, como bien señalamos anteriormente, nuestro ID queda borrado del portapapeles, y nuestro usuario insertado en la tabla de “admitidos” de la base de datos “usuarios”. Si por el contrario nos hemos dejado algún dato sin rellenar, o los hemos introducido erróneamente (como por ejemplo poner un correo sin @), nos saltará el control de errores y la inserción del usuario quedará cancelada.



## - Listar

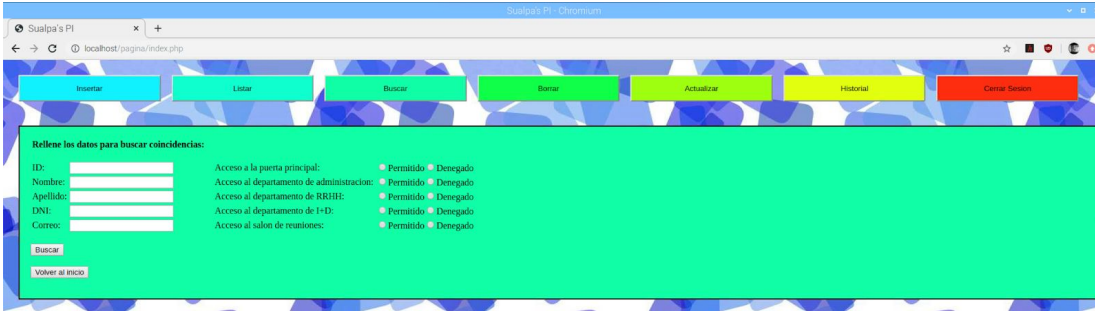
La siguiente opción de nuestro menú global es la de Listar. Si presionamos sobre ella, nos carga una tabla en la que se refleja toda la información de todos los usuarios. El número de usuario nos indica en un orden ascendente, en qué posición fue insertado tal usuario. El id nos muestra el ID de su tarjeta RFID. A continuación se nos indican tanto su nombre y apellidos como su DNI y su correo. Finalmente se nos muestran con la palabra “Permitido” las puertas a las que tiene el acceso disponible y con la palabra “Denegado” las puertas a las que tiene el acceso restringido y por tanto su tarjeta RFID dará error.



N. de usuario	ID	Nombre	Apellido	DNI	Correo	Puerta Principal	Dpto. Administración	Dpto. RRHH	Dpto. I+D	Sala de Reuniones
1	1	adrián	Miguel	123456789	am@pun.com	Denegado	Denegado	Denegado	Denegado	Denegado
8	2	samuel	rodr	222222222	as@pma.cl	Denegado	Denegado	Denegado	Denegado	Denegado
9	584197271971	Jose	Miguel	12345675	am@pun.com	Denegado	Denegado	Denegado	Denegado	Denegado
10	584197271971	Alba	pasa	94249424A	al@iba	Denegado	Denegado	Denegado	Denegado	Denegado
13	123456789123	prueba	puerta	66666666B	ag@com	Permitido	Denegado	Permitido	Denegado	Permitido
14	584197271971	pepito	pepito	99999999F	am@pun.com	Permitido	Permitido	Permitido	Permitido	Permitido
33	444444444444	pruebasreus2	dosss	21212121F	sg@i.c	Permitido	Denegado	Permitido	Denegado	Denegado

## - Buscar

Si nos dirigimos a la opción de buscar, automáticamente se nos abre el siguiente formulario. Este funciona a modo de criba, por lo que si no establecemos ninguna condición, se nos muestran todos los datos de la base de datos de “usuarios”, siendo el mismo resultado que en la opción de “listar”. Si ponemos un nombre determinado, por ejemplo Juan, nos salen todos los Juanes de la base de datos, junto con el resto de sus datos. Si marcamos “Permitido” en “Acceso a la puerta principal” y “Denegado” en “acceso al departamento de administración”, nos salen las personas que pueden entrar por la puerta principal, las que no pueden entrar en el departamento de administración y las que pueden o no entrar por cada una de las otras puertas. Como vemos, cuantas más condiciones pongamos, más restrictiva será la búsqueda, hasta el punto en que solo nos salga la persona de quien hemos puesto sus datos específicos, o en caso de no haber tal usuario, nos salga una tabla vacía.



Rellene los datos para buscar coincidencias:

ID:

Nombre:

Apellido:

DNI:

Correo:

Acceso a la puerta principal:  Permitido  Denegado

Acceso al departamento de administración:  Permitido  Denegado

Acceso al departamento de RRHH:  Permitido  Denegado

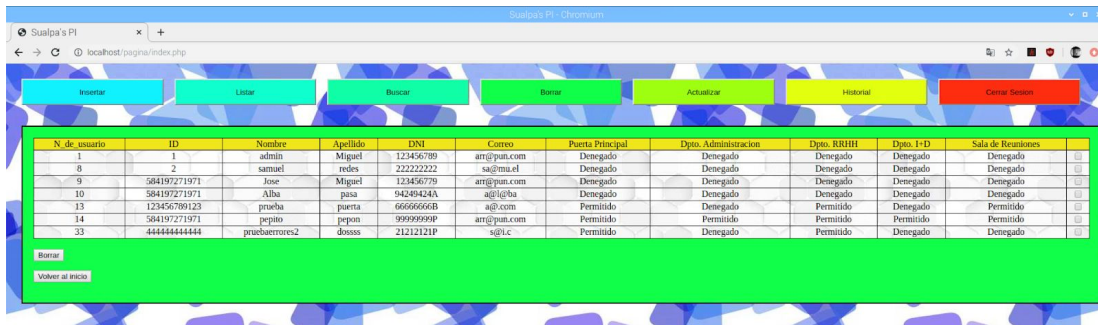
Acceso al departamento de I+D:  Permitido  Denegado

Acceso al salon de reuniones:  Permitido  Denegado

## - Borrar

Si la opción seleccionada es la de borrar, se nos carga un listado de todos los usuarios de la base de datos “usuarios” junto a todos sus datos. A la derecha de cada usuario, aparece un

campo de tipo checkbox, el cual podemos marcar para eliminar tantos usuarios como queramos de una sola vez. Para ello, una vez marquemos los deseados, solo tendremos que pulsar el botón de “Borrar” y estos serán eliminados junto con sus datos de la base de datos “usuarios”

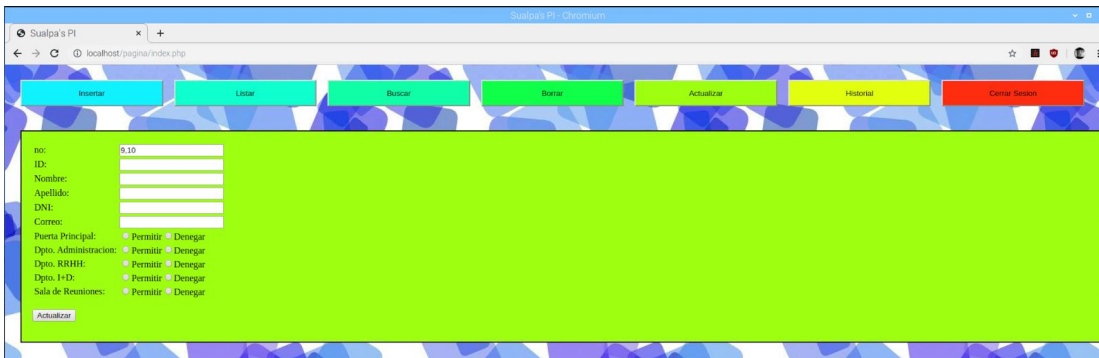


## - Actualizar

La opción de actualizar es la última de las opciones en la base de datos “usuarios”. Al igual que en la opción de borrar, nos aparece un checkbox para poder actualizar uno o más usuarios de una sola vez. Para ello, solo tendremos que marcar los checkbox de quienes queremos actualizar y pulsar en siguiente.



Una vez hemos pulsado siguiente, nos aparece el siguiente formulario. En él, se nos muestra el número de usuario de quien estamos actualizando, ya sean uno o varios. La edición llegados a este punto es bastante abierta, ya que se permite que varias personas tengan el mismo ID, lo que significa que un grupo de trabajo tiene compartida la llave de una puerta, o por el contrario, que una persona tenga varias llaves. Los campos que no rellenemos quedarán exactamente igual que estaban antes de la actualización del usuario, por lo que si una llave pasa de un usuario a otro, bastará con cambiar el nombre, apellido, DNI y correo del propietario, siendo innecesario poner las puertas a las que tiene acceso y el ID de la puerta.

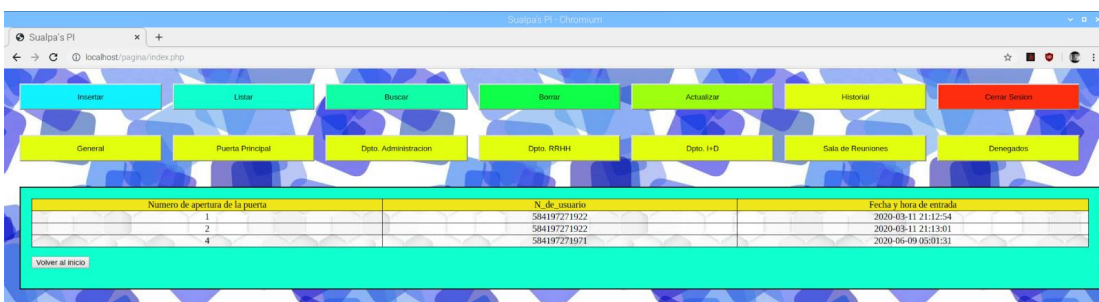


## - Historial

Al pulsar sobre la opción de historial se nos despliegan las distintas puertas de las que consultar el historial. En nuestro supuesto, solo contamos con cinco puertas, así que de las siete casillas mostradas, cinco son para las puertas, una para el historial conjunto de estas, y la restante para el historial de intentos de acceso denegados.



Si pulsamos sobre cualquiera de las opciones, se nos muestra el historial correspondiente de uso de cada puerta, en el que vemos el número de veces que se ha utilizado la puerta, seguido del ID de quien ha hecho tal uso y completado con la fecha y hora exactas del momento en que se hizo.



## - Cierre de sesión

La opción restante del menú global, es el cierre de sesión. Una vez pinchamos sobre esta opción, seremos redirigidos a la página de inicio de sesión, y no podremos volver atrás. Para poder acceder de nuevo a la administración, deberemos introducir nuestras



credenciales una vez más.



## 5. Autenticación en tres pasos generalizado

En el contexto en el que vivimos, la cantidad de datos valiosos que las empresas deben almacenar son muchísimos y crecen a un ritmo sin precedentes. En consecuencia, sucede que la cantidad de personas que están al acecho de estos datos e información aumenta proporcionalmente, pues ya sabemos... la información es poder.

Para proteger estos datos, la forma más segura es establecer una autenticación en 3 pasos, de tal manera que nadie pueda suplantar a los usuarios que tengan acceso a estos datos. La autenticación en 3 pasos responde a 3 tipos distintos de datos que el usuario ha de introducir: algo que el usuario tiene, algo que el usuario es y algo que el usuario sabe.

Estos tres factores suelen ir por separado: cuando queremos abrir nuestro coche o nuestra casa, usamos unas llaves (sistemas basados en algo que posee el usuario, “algo que tenemos”). Estas llaves pueden ser con una forma física determinada que abre una cerradura, o por el contrario, llaves digitales/electrónicas con un código establecido en un dispositivo físico (como una llave de radiofrecuencia, un código de barras o un código BIDI), el cual utilizaremos para acceder a un recurso restringido electrónicamente al pasarlo por un lector de códigos. Por otro lado, cuando queremos desbloquear nuestro móvil, cada vez tendemos a usar más la autenticación biométrica (sistema basado en características físicas del usuario, “algo que somos”). Diferentes opciones son, el uso de una huella dactilar, el reconocimiento facial e incluso el reconocimiento de voz. Por último, tenemos las contraseñas clásicas (sistemas basados en algo conocido por el usuario, “algo que sabemos”). Este tipo de autenticación es el más conocido, además de ser el más utilizado hasta la fecha y por ende, también el más fácil de sustraer mediante técnicas de phishing. Las contraseñas pueden ser numéricas, alfabéticas o una mezcla de ambas (alfanuméricas) y dentro de cada una, más o menos complejas. Las más sustraibles, son las que se basan en asuntos personales (como el nombre o fecha de nacimiento de algún ser querido) o en patrones conocidos (como “1234” o “abcd”). Por otro lado, las más complejas son las que se generan de manera aleatoria y sus caracteres no tienen nada que ver, siendo totalmente independientes unos de otros.

Pese a los pros y contras de cada tipo de autenticación, requerir los tres métodos es actualmente la forma más segura de proteger cualquier tipo de dato.

### **Autenticación 1 - Algo que tu tienes**

Para implementar la filosofía de “algo que tu tienes”, usaremos las tarjetas o llaves RFID. El modo de funcionamiento de los sistemas RFID es simple: la tarjeta RFID contiene unos datos de identificación numérica. Esta tarjeta, genera una señal de radiofrecuencia de manera constante con dichos datos. Cuando esta señal es captada por un lector RFID, éste se encarga de leer la información y pasarla en formato digital a la aplicación específica que utiliza RFID.

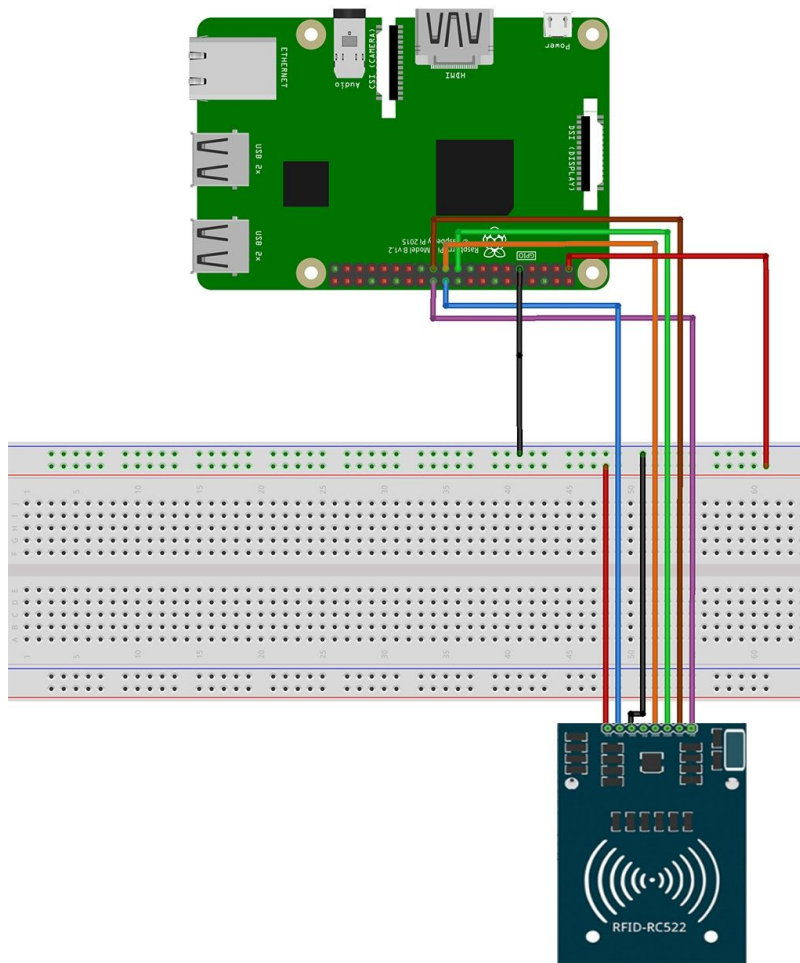
## Cómo configurar un chip RFID RC522 Raspberry:

### Lista del equipo necesario:

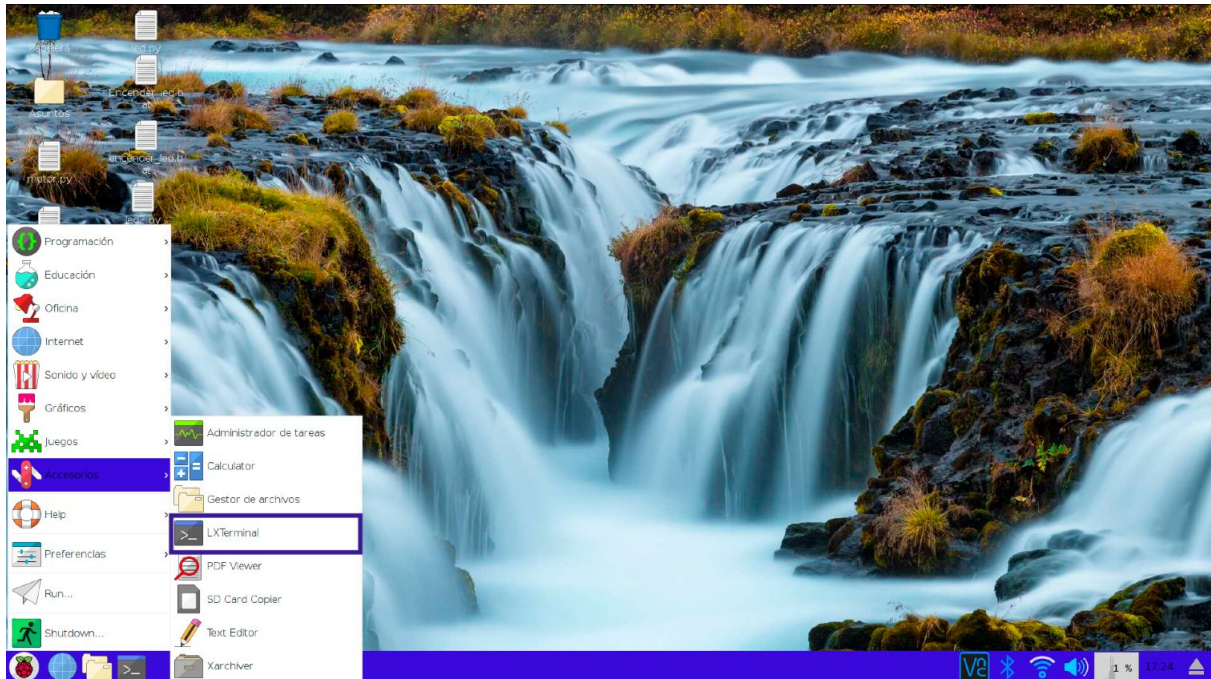
- Raspberry Pi 2, 3, o 3 B+
- Fuente de alimentación
- Tarjeta SD
- Protoboard (es opcional, ya que podemos conectar nuestro lector a la Raspberry directamente mediante conexiones hembra-hembra)
- Jump Wires (macho-hembra ó hembra-hembra)
- Lector RFID RC522
- Llave RFID

### Montaje en la RaspBerry:

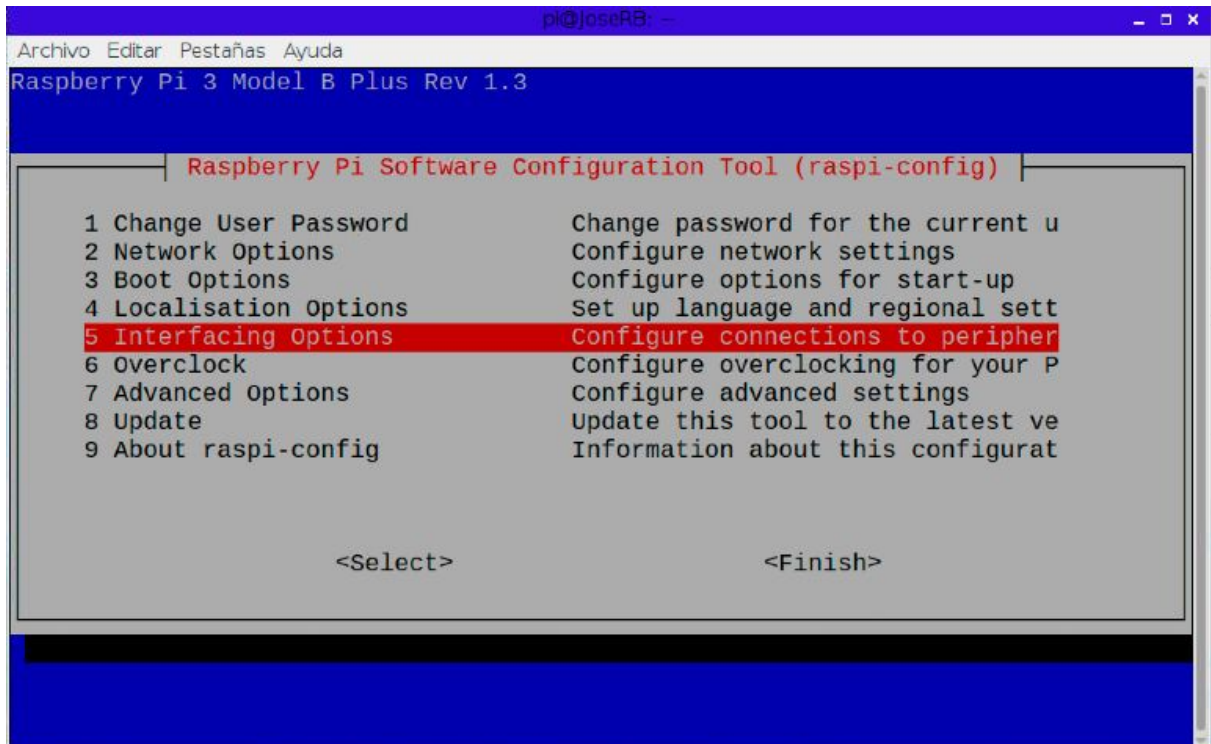
- SDA se conecta al Pin 24
- SCK se conecta al Pin 23
- MOSI se conecta al Pin 19
- MISO se conecta al Pin 21
- GND se conecta al Pin 9
- RST se conecta al Pin 22
- 3.3v se conecta al Pin 1

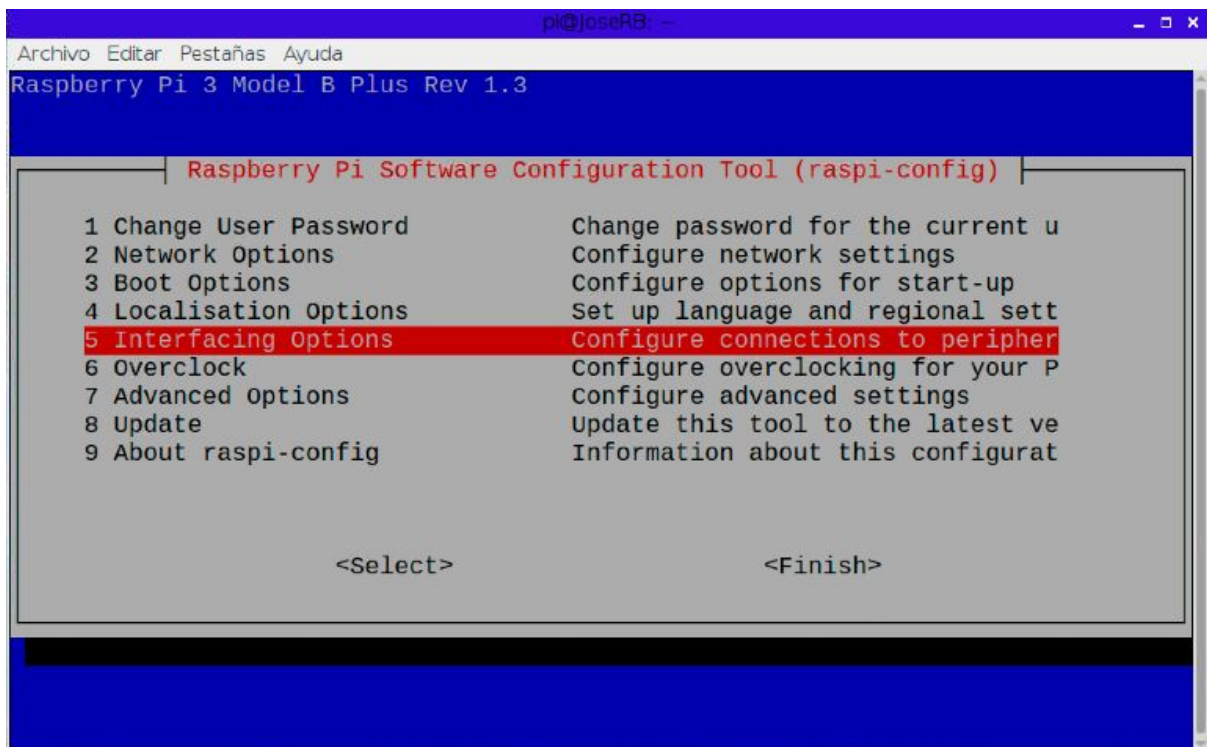
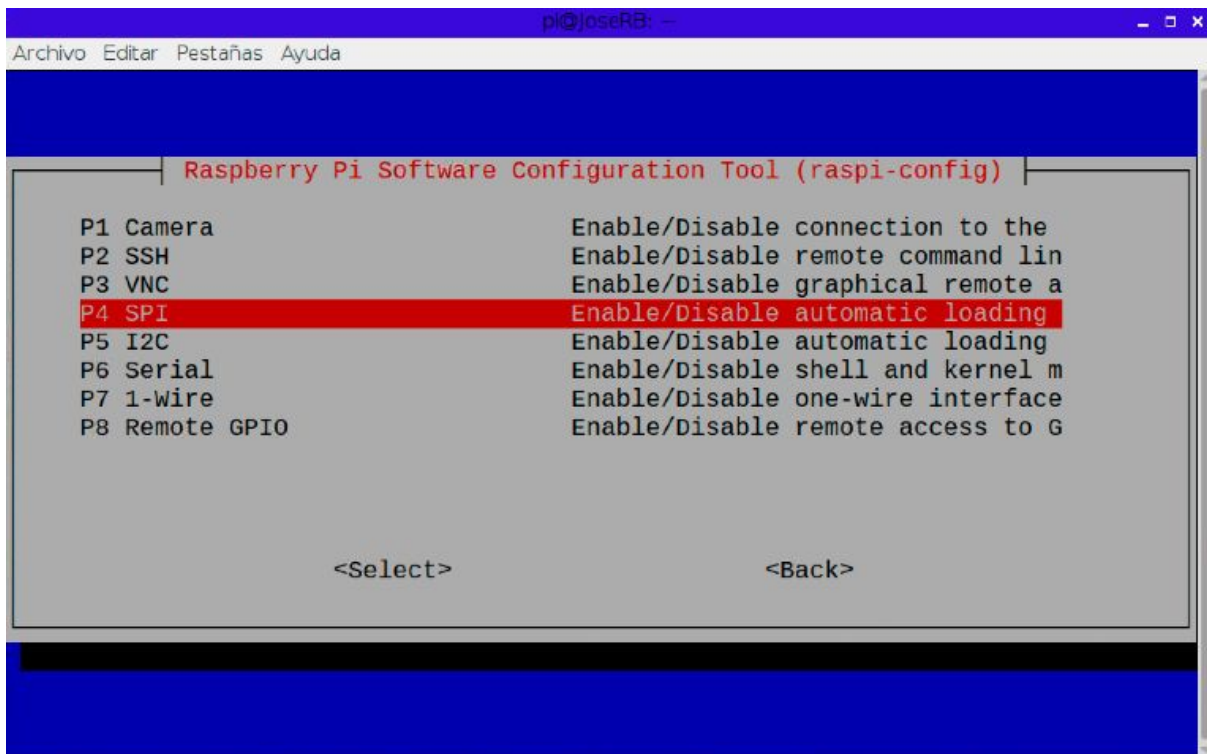


-Entramos en el terminal de la raspberry:



1º Debemos activar la interfaz SPI : `sudo raspi-config`





2º Reiniciar la raspberry: `sudo reboot`

3º Para ver si hemos habilitado SPI introducimos: `lsmod | mod grep spi` (en la pantalla aparecerá `spi_bcm2835` si está activado)

- Si por alguna razón SPI no se ha activado, podemos editar el archivo de configuración de inicio manualmente ejecutando el siguiente comando en nuestra raspberry:

```
sudo nano /boot/config.txt
```

- Buscamos: `dtoverlay=spi=on`, si delante de este hay una `#` deberemos quitárselo, y si no aparece, escribir: `dtoverlay=spi=on` al final del archivo.

- Cuando realizamos los cambios, presionamos Ctrl+X y después presionar Enter.

- Reiniciamos la Raspberry.

4º Antes de realizar la programación:

- Debemos tener actualizada nuestra Raspberry:

```
sudo apt-get update y sudo apt-get upgrade
```

- Instalar como hemos mostrado anteriormente los paquetes `python3-dev`, `python3-pip` y `git`:

```
sudo apt-get install python3-dev python3-pip
```

- Además instalaremos el `spidev` de Python Library (La biblioteca `spidev` ayuda a manejar las interacciones con el SPI y es un componente clave, ya que lo necesitamos para que la Raspberry interactúe con el RFID RC522):

```
sudo pip3 install spidev
```

- Ahora que hemos instalado la biblioteca `spidev` en nuestra Raspberry, podemos proceder a instalar la biblioteca `MFRC522` usando `pip` también:

```
sudo pip3 install mfrc522
```

5º “Escribiendo con el RFID RC522”:

Cómo escribir datos del RC522 en sus etiquetas RFID:

- Creamos una carpeta (`pi-rfid`) para guardar los scripts:

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

- Cambiamos de directorio a la carpeta creada y empezamos a escribir el script **Write.py**:

```
cd ~/pi-rfid
```

```
sudo nano Write.py
```

- Dentro del archivo escribiremos:

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

```
import RPi.GPIO as GPIO
```

```
from mfrc522 import SimpleMFRC522
```

```
reader = SimpleMFRC522()
```

```
try:
```

```
    text = input('New data:')
```

```
    print("Now place your tag to write")
```

```
    reader.write(text)
```

```
    print("Written")
```

```
finally:
```

```
GPIO.cleanup()
```

- Presionamos Ctrl+X para guardar y después presionar Enter para salir del archivo.
- Para comprobar el script Write.py:

```
sudo python3 Write.py
```

```
pi@raspberrypi:~/pi-rfid $ sudo python3 Write.py
New data: Usuario 1
Now place your tag to write (pasamos la etiqueta RFID)
Written
```

- Con este script asignamos el nombre (Usuario 1) a la etiqueta RFID.

6° “Leyendo con el RFID RC522”:

Ahora que hemos escrito nuestro script para escribir en etiquetas RFID usando nuestro RC522, ahora podemos escribir un script que leerá estos datos de la etiqueta.

- Sin movernos de la carpeta `pi-rfid`, escribiremos el script **Read.py**:

```
cd ~/pi-rfid
```

```
sudo nano Read.py
```

- Dentro del archivo escribiremos:

```
#!/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()
```

- Para comprobar el script Read.py:

```
sudo python3 Read.py
```

```
pi@raspberrypi:~/pi-rfid $ sudo python3 Read.py
827843705425
Usuario 1
```

- Con este script comprobamos que al pasar la etiqueta RFID por el lector tiene asignado el Usuario 1 que anteriormente creamos.

## Autenticación 2 - Algo que tu eres

Con “algo que tu eres” nos referimos al campo de la biometría. Éste es un campo que aún tiene un largo camino que explorar y en el que sin duda habrá grandes avances. Hoy por hoy, las tres pruebas biométricas más destacables son el reconocimiento por voz, la huella dactilar y el reconocimiento facial.

En nuestro caso, hemos escogido el reconocimiento facial, ya que presenta ciertas características indispensables que las otras opciones carecen. En primer lugar, el reconocimiento por voz presenta una fiabilidad bastante más baja que el reconocimiento facial ya que esta nos puede cambiar por motivos como un simple catarro, además de no abastecer a la población que no conste, por unos motivos u otros, de voz propia. En comparación, todo el mundo consta de cara y los cambios en esta son o muy difíciles de darse, o muy lentos. En segundo lugar, el uso de la huella dactilar nos proporciona una muy alta fiabilidad y para temas personales como puede ser un teléfono móvil, es ideal. Sin embargo, cuando se trata de un acceso de uso cotidiano por el que pasan multitud de personas, es un método bastante antihigiénico, mientras que el reconocimiento facial evita cualquier tipo de contacto ya que es una cámara la que se encarga del proceso y no un sensor táctil.

Dado que la implementación de la biometría es un proceso largo a la vez que complicado, se facilitará toda la información en el anexo biometría.

## Autenticación 3 - Algo que tu sabes

Finalmente, nos encontramos con la parte de “algo que tú sabes”, en este caso una contraseña numérica. Para ello, habilitamos dos botones y una contraseña que, simulando un candado numérico, consta de cuatro números. Con el primer botón, estableceremos la cifra que queremos marcar, mientras que con el segundo la confirmaremos, teniendo que confirmar un total de 4 veces. De esta manera, si nuestra contraseña es “1234” tendremos que pulsar el botón1 una vez y luego el botón2 para confirmarlo, luego el botón1 dos veces y el botón2 para confirmar de nuevo, botón1 tres veces seguido de botón2 y finalmente botón1 cuatro veces seguido de botón2. Si por algún casual, nos equivocamos y excedemos la cifra que queremos meter, no habrá problema ya que una vez que lleguemos al 9 y pulsemos el 1 de nuevo, empezaremos en el 0 una vez más.

Primero importamos la librería de GPIO y establecemos el modo de lectura en BCM:

```
import RPi.GPIO as GPIO
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
```

A continuación declaramos los botones en los pins BCM correspondientes y los ponemos en modo salida:

```
BtnA=12
BtnB=16
```



```
GPIO.setup(BtnA, GPIO.OUT)
GPIO.setup(BtnB, GPIO.OUT)
```

A seguido crearemos la función que definirá cada cifra. Ponemos un contador a 0 y hacemos que cada vez que se pulse el botón, sume 1 al contador:

```
def numeroClave():
    contador=0
    while True:
        if GPIO.input(BtnA)==True:
            contador +=1
```

Creamos una condición que haga que si el número sobrepasa el 9, que vuelva a empezar desde 0:

```
        if contador > 9:
            contador=0
            print(contador)
```

Dado que el contador incrementa cuando el pulsador está presionado, éste puede incrementar inintencionadamente con una sola pulsación, ya que el tiempo que tardamos en soltar cuenta como pulsación. Es por ello que crearemos un bucle while infinito, que solo se romperá cuando se deje de pulsar el botón, haciendo así imposible que cuente más de una pulsación sin haber soltado el pulsador.

```
        while GPIO.input(BtnA)==True:
            if GPIO.input(BtnA)==False:
                break
```

Para finalizar la función, hacemos que si se presiona el segundo botón, devuelva como valor el contador de pulsaciones a una variable.

```
        if GPIO.input(BtnB)==True:
            return contador
```

Finalmente, crearemos un bucle while, en el que llamaremos 4 veces a la función anterior, metiendo el resultado que ésta nos da en 4 variables distintas, consiguiendo así la contraseña buscada.

```
while True:
    if GPIO.input(BtnB)==False:
        contadorA=numeroClave()
        print("Su numero a es ", contadorA)
        while True:
            if GPIO.input(BtnB)==False:
                contadorB=numeroClave()
                print("Su numero b es ", contadorB)
                while True:
```

```

        if GPIO.input(BtnB)==False:
            contadorC=numeroClave()
            print("Su numero c es ", contadorC)
            while True:
                if GPIO.input(BtnB)==False:
                    contadorD=numeroClave()
                    print("Su numero d es ", contadorD)
                    break
            break
        break
    break
print("Su contraseña es", contadorA, contadorB, contadorC, contadorD)

```

A continuación se muestra una foto tanto del código y como de la implementación:

```

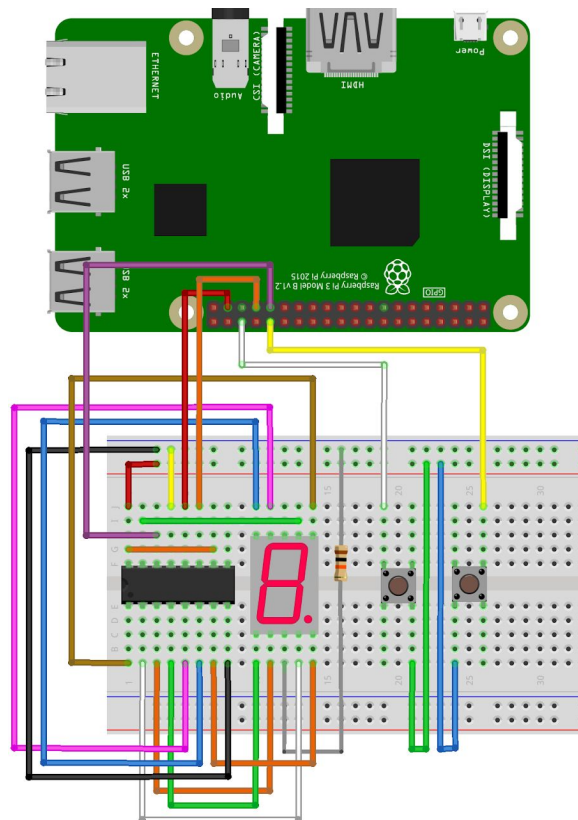
import RPi.GPIO as GPIO
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

BtnA=12
BtnB=16

GPIO.setup(BtnA, GPIO.OUT)
GPIO.setup(BtnB, GPIO.OUT)
def numeroClave():
    contador=0
    while True:
        if GPIO.input(BtnA)==True:
            contador +=1
            print(contador)
            while GPIO.input(BtnA)==True:
                if GPIO.input(BtnA)==False:
                    break
            if GPIO.input(BtnB)==True:
                return contador

while True:
    if GPIO.input(BtnB)==False:
        contadorA=numeroClave()
        print("Su numero a es ", contadorA)
        while True:
            if GPIO.input(BtnB)==False:
                contadorB=numeroClave()
                print("Su numero b es ", contadorB)
                while True:
                    if GPIO.input(BtnB)==False:
                        contadorC=numeroClave()
                        print("Su numero c es ", contadorC)
                        while True:
                            if GPIO.input(BtnB)==False:
                                contadorD=numeroClave()
                                print("Su numero d es ", contadorD)
                                break
                            break
                        break
                    break
                break
            break
        break
print("Su contraseña es", contadorA, contadorB, contadorC, contadorD)

```



## 6. Autenticador.py

El programa que engloba toda la autenticación del sistema, es “autenticador.py”. La intención es que nada más encender el sistema, éste empiece a funcionar automáticamente, y para ello hemos añadido la línea:

```
@/usr/bin/python3 /home/pi/Desktop/autenticador.py
```

al archivo:

```
/etc/xdg/lxsession/LXDE-pi/autostart
```

Lo primero que hace el sistema, es asegurarse de que las puertas están bloqueadas. Para ello, comprueba la posición del cerrojo y acciona el servomotor para que queden cerradas si no lo estaban ya. A la par, se enciende la luz led de color rojo indicándonos que actualmente el acceso no está inhabilitado. En este mismo instante, el verificador RFID empieza a funcionar, de forma que si detecta algún emisor de radiofrecuencia, ya sea porque queremos entrar con nuestro smartphone a través del NFC o con nuestra llave electrónica RFID, comprueba si está registrado en la base de datos. Si no es así, el programa notifica en una base de datos del servidor central el ID con el que se ha intentado entrar junto con la fecha y hora, mientras la luz roja del led empieza a parpadear para indicarnos que con esa llave no tenemos permitido el acceso.

En caso de tener acceso con esa llave o smartphone, se enciende automáticamente la cámara del sistema para comenzar con la prueba biométrica de reconocimiento facial y comprobar así que no se trata de un artículo robado. Si este es el caso, una vez más se inicializará el proceso de notificación en la base de datos, y la luz roja volverá a parpadear momentáneamente. Si por el contrario el reconocimiento facial es correcto, el programa nos deja pasar al tercer y último paso.

Por si el paso anterior hemos conseguido superarlo “milagrosamente” mediante una foto, una réplica facial de silicona, un hermano gemelo y demás casos que todos conocemos de películas de robos (pero que perfectamente se pueden dar), se nos pedirá que introduzcamos una contraseña numérica que solo las personas con acceso al interior conocerán, dando como resultado que la puerta se abra si somos quien decimos ser o que por el contrario, se encienda una vez más la parpadeante luz roja y sea notificado a la base de datos nuestro intento de intrusión.

Si efectivamente superamos los tres controles de autenticación, automáticamente se apaga la luz roja dando paso a la verde a la vez que se acciona el servomotor que abre la cerradura liberando la puerta y dejándonos pasar.

En el mismo instante en que cerremos la puerta, el servomotor bloqueará de nuevo la cerradura automáticamente, y la luz verde volverá de nuevo a ser sustituida por la luz roja.

## 7. Suricata en nuestro proyecto

Suricata es un tipo de IDS (sistema de detección de intrusos) que nos permite incrementar la seguridad de nuestra red y se encarga de analizar el tráfico mediante unas reglas establecidas previamente, las cuales detectan los comportamientos extraños y las actividades sospechosas o dañinas y alertan de estas al administrador.

Dado que nuestras cerraduras tienen todas una IP estática, la primera regla que tenemos establecida es para que alerte y deniegue cualquier tipo de tráfico procedente de una IP desconocida. De esta manera, pese a que la red de seguridad sea una red privada, si alguien consigue entrar en ella no podrá acceder al servidor central.

La segunda regla es para que, desde nuestras IPs conocidas, solo pueda circular tráfico de MySQL a través del puerto 3306, de manera que cada cerradura pueda acceder a la base de datos central, consultar en ella si el usuario que intenta acceder está registrado para esa cerradura y obtener una respuesta. A su vez, también tendrá el permiso de establecido previamente de insertar nuevos datos en la tabla historial para notificar todos los intentos de acceso. Para más detalles sobre su instalación y uso, consultar [ANEXO VI - Suricata](#)

## 8. Bibliografía

- How to setup a Raspberry Pi RFID RC522 Chip.  
<https://pimylifeup.com/raspberry-pi-rfid-rc522/>
- ¡Asignación de pines! La guía detallada sobre la asignación de pines GPIO para Raspberry Pi.  
<https://es.pinout.xyz/>
- El sitio web de desarrollador más grande del mundo.  
<https://www.w3schools.com/>
- Sunfounder.  
<https://www.sunfounder.com/>
- MariaDB - “Configuring MariaDB with Option Files”  
<https://mariadb.com/kb/en/configuring-mariadb-with-option-files/>
- Controlar un servo motor con Raspberry Pi.  
<https://www.nociones.de/controlar-un-servo-con-raspberry-pi-usando-rpio-pwm-y-dma/>
- Suricata User Guide Release 4.10-dev OISF.  
<https://buildmedia.readthedocs.org/media/pdf/suricata/suricata-4.1.2/suricata.pdf>
- CERTSI GUIA SCI 004 - Configuración IPS, IDS y SIEM 2017 v1.  
[https://www.incibe-cert.es/sites/default/files/contenidos/guias/doc/certsi\\_diseno\\_configuracion\\_ips\\_ids\\_siem\\_en\\_sci.pdf](https://www.incibe-cert.es/sites/default/files/contenidos/guias/doc/certsi_diseno_configuracion_ips_ids_siem_en_sci.pdf)
- Guía del usuario de Suricata.  
<https://suricata.readthedocs.io/en/suricata-4.1.3/index.html>
- Blog “Follow The White Rabbit” : Suricata IDS – Instalación, puesta en marcha y primera prueba.  
<https://fwhibbit.es/suricata-ids-instalacion-puesta-en-marcha-y-primera-prueba>
- <https://suricata-ids.org/>
- “Ventajas e implementación de un sistema IDS en el ámbito familiar” - José Antonio Salom Martín.  
<http://openaccess.uoc.edu/webapps/o2/bitstream/10609/95087/6/joansaTFM0619memoria.pdf>
- Blog “Kimaldi” : Reconocimiento facial.  
[https://www.kimaldi.com/blog/biometria/reconocimiento\\_facial/](https://www.kimaldi.com/blog/biometria/reconocimiento_facial/)
- Techno business guide “The Advantages of Face Recognition Technology”  
<https://techbusinessguide.com/benefits-facial-recognition-everyone-should-know/>

- Techno business guide “Disadvantages of Facial Recognition”  
<https://techbusinessguide.com/disadvantages-facial-recognition/>
- Blog “imesd electrónica, s.l” : Reconocimiento facial vs huellas dactilares.  
<https://imesd.es/es/blog/reconocimiento-facial-vs-huellas-dactilares/>
- Proenter : Reconocimiento facial vs huellas dactilares.  
<https://proenter.es/reconocimiento-facial-vs-huella-dactilar>
- Otros programas para capturar fotos y videos con la webcam en GNU/Linux.  
<https://javiertectos.wordpress.com/2016/10/14/instalacion-de-webcam-usb-en-raspberry-pi/>
- Face\_RecognitionOpenCv2: [https://github.com/jorge190588/face\\_recognitionOpenCv2](https://github.com/jorge190588/face_recognitionOpenCv2)
- Introducción python: <https://www.mclibre.org/consultar/python/>
- OpenCV raspberry: <https://descubrearduino.com/opencv-en-raspberry-pi/>
- OpenCV: <https://pypi.org/project/opencv-python/>
- OpenCV: <https://github.com/milq/milq/blob/master/scripts/bash/install-opencv.sh>

# ANEXO I - Administración Web

## registro\_entrar.html

```
1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 <style>
5 .a{
6 position: relative;
7 top: 25%;
8 }
9 .b{
10 width:20%;
11 background-color:white;
12 padding-top: 0.3%;
13 padding-bottom: 0.1%;
14 }
15 </style>
16 </head>
17 <body bgcolor="lightblue">
18 <br>
19 <form action="index.php" method="POST">
20 <table>
21 <tr><INPUT TYPE="button" NAME="insertar" VALUE="Insertar" style ="width: 13%; height: 5%; float: left;background-color: #11F3FF; margin-left: 1%; filter: blur(2px);"></tr>
22 </form>
23 <form action="index.php" method="POST">
24 <tr><INPUT TYPE="button" NAME="listar" VALUE="Listar" style ="width: 13%; height: 5%; float: left; background-color: #11FFCD; margin-left: 1%; filter: blur(2px);"></tr>
25 </form>
26 <form action="index.php" method="POST">
27 <tr><INPUT TYPE="button" NAME="buscar" VALUE="Buscar" style ="width: 13%; height: 5%; float: left; background-color: #11FFA7; margin-left: 1%; filter: blur(2px);"></tr>
28 </form>
29 <form action="index.php" method="POST">
30 <tr><INPUT TYPE="button" NAME="borrar" VALUE="Borrar" style ="width: 13%; height: 5%; float: left; background-color: #11FF48; margin-left: 1%; filter: blur(2px);"></tr>
31 </form>
32 <form action="index.php" method="POST">
33 <tr><INPUT TYPE="button" NAME="actualizar" VALUE="Actualizar" style ="width: 13%; height: 5%; float: left; background-color: #9FFF11; margin-left: 1%; filter: blur(2px);"></tr>
34 </form>
35 <form action="index.php" method="POST">
36 <tr><INPUT TYPE="button" NAME="pagina" VALUE="Historial" style ="width: 13%; height: 5%; float: left; background-color: #E3FF11; margin-left: 1%; filter: blur(2px);"></tr>
37 </form>
38 <form action="index.php" method="POST">
39 <tr><INPUT TYPE="button" NAME="salir" VALUE="Cerrar Sesion" style ="width: 13%; height: 5%; float: left; background-color: #FF2D11; margin-left: 1%; filter: blur(2px);"></tr>
40 </form><br><br>
41 <table align="center" class="a" >
42 <div>
43 
44 <div align="center" >SUALPA'S PI</div>
45 <form ACTION="registro.php" METHOD="POST">
46 <table align="center">
47 <tr>
48 <td>Usuario: </td> <td><INPUT TYPE="text" NAME="nombre" size=15<BR></td>
49 </tr>
50 <tr>
51 <td>Contraseña: </td><td> <INPUT TYPE="password" NAME="contra"size=15<BR></td>
52 </tr>
53 </table>
54 <br>
55 <INPUT TYPE="submit" name="enviar" VALUE="ENTRAR">
56 </div>
57 </div>
58 </form>
59 </body>
60 </html>
```

## index.php

```
1 <?php
2 session_start();
3 ?>
4 <?PHP
5 if(isset($_SESSION["nombre"])){
6 }
7 <HTML>
8 <HEAD>
9 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
10
11 <title>Sualpa's PI</title>
12 <style>
13 body{
14 background-image: url('descarga.png');
15 }
16 </style>
17 </HEAD>
18 <BODY bgcolor= "lightblue">
19 <br>
20 <table>
21 <tr><form action="index.php" method="POST">
22 <tr><INPUT TYPE="submit" NAME="insertar1" VALUE="Insertar" style ="width: 13%; height: 5%; float: left; background-color: #11F3FF; margin-left: 1%;"></tr>
23 </form>
24 <form action="index.php" method="POST">
25 <tr><INPUT TYPE="submit" NAME="listar" VALUE="Listar" style ="width: 13%; height: 5%; float: left; background-color: #11FFCD; margin-left: 1%;"></tr>
26 </form>
27 <form action="index.php" method="POST">
28 <tr><INPUT TYPE="submit" NAME="buscar" VALUE="Buscar" style ="width: 13%; height: 5%; float: left; background-color: #11FFA7; margin-left: 1%;"></tr>
29 </form>
30 <form action="index.php" method="POST">
31 <tr><INPUT TYPE="submit" NAME="borrar" VALUE="Borrar" style ="width: 13%; height: 5%; float: left; background-color: #11FF48; margin-left: 1%;"></tr>
32 </form>
33 <form action="index.php" method="POST">
34 <tr><INPUT TYPE="submit" NAME="actualizar" VALUE="Actualizar" style ="width: 13%; height: 5%; float: left; background-color: #9FFF11; margin-left: 1%;"></tr>
35 </form>
36 <form action="index.php" method="POST">
37 <tr><INPUT TYPE="submit" NAME="pagina" VALUE="Historial" style ="width: 13%; height: 5%; float: left; background-color: #E3FF11; margin-left: 1%;"></tr>
38 </form>
39 <form action="index.php" method="POST">
40 <tr><INPUT TYPE="submit" NAME="salir" VALUE="Cerrar Sesion" style ="width: 13%; height: 5%; float: left; background-color: #FF2D11; margin-left: 1%;"></tr>
41 </form><br><br>
42 </table>
```

```

43 <?PHP
44 //-----
45 if (isset ($_REQUEST["insertar1"])) {
46     require_once "insertar1.php";
47 }
48 else if (isset ($_REQUEST["copiar"])) {
49     require_once "copiar.php";
50 }
51 else if (isset ($_REQUEST["insertar"])) {
52     require_once "insertar.php";
53 }
54 else if (isset ($_REQUEST["insertar3"])) {
55     require_once "insertar3.php";
56 }
57 //-----
58 else if (isset ($_REQUEST["listar"])) {
59     require_once "listar.php";
60 }
61 //-----
62 else if (isset ($_REQUEST["buscar"])) {
63     require_once "buscar.php";
64 }
65 else if (isset ($_REQUEST["buscar2"])) {
66     require_once "buscar2.php";
67 }
68 //-----
69 else if (isset ($_REQUEST["borrar"])) {
70     require_once "borrar.php";
71 }
72 else if (isset ($_REQUEST["borrar2"])) {
73     require_once "borrar2.php";
74 }
75 //-----
76 else if (isset ($_REQUEST["actualizar"])) {
77     require_once "actualizar.php";
78 }
79 else if (isset ($_REQUEST["actualizar2"])) {
80     require_once "actualizar2.php";
81 }
82 else if (isset ($_REQUEST["actualizar3"])) {
83     require_once "actualizar2.php";
84 }
85 //-----
86 else if (isset ($_REQUEST["index"])) {
87     require_once "index.php";
88 }
89 //-----
90 else if (isset ($_REQUEST["pagina"])) {
91     require_once "dos_enclaces.php";
92 }
93 elseif (isset ($_REQUEST["general"])) {
94     require_once "dos_enclaces.php";
95     require_once "general.php";
96 }
97 elseif (isset ($_REQUEST["principal"])) {
98     require_once "dos_enclaces.php";
99     require_once "principal.php";
100 }
101 elseif (isset ($_REQUEST["admin"])) {
102     require_once "dos_enclaces.php";
103     require_once "admin.php";
104 }
105 elseif (isset ($_REQUEST["rrhh"])) {
106     require_once "dos_enclaces.php";
107     require_once "rrhh.php";
108 }
109 elseif (isset ($_REQUEST["imasd"])) {
110     require_once "dos_enclaces.php";
111     require_once "imasd.php";
112 }
113 elseif (isset ($_REQUEST["sala"])) {
114     require_once "dos_enclaces.php";
115     require_once "sala.php";
116 }
117 elseif (isset ($_REQUEST["denegados"])) {
118     require_once "dos_enclaces.php";
119     require_once "denegados.php";
120 }
121 ?>
122 <?PHP
123 if(isset($_REQUEST['salir'])) {
124     session_destroy ();
125     header ('Location: ../registro_entrar.html');
126 }
127 ?>
128 <?PHP
129 }
130 else{
131     ?>
132 <html>
133 <head>
134 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>

```



```

135 <style>
136 .a{
137 position: relative;
138 top: 25%;
139 }
140 .b{
141 width:20%;
142 background-color:white;
143 padding-top: 0.3%;
144 padding-bottom: 0.1%;
145 }
146 </style>
147 </head>
148 <body bgcolor="lightblue">
149 <br>
150 <form action="index.php" method="POST">
151 <table>
152 <tr><INPUT TYPE="button" NAME="insertar" VALUE="Insertar" style ="width: 13%; height: 5%; float: left;
153 background-color: #11F3FF; margin-left: 1%; filter: blur(2px);"></tr>
154 </form>
155 <form action="index.php" method="POST">
156 <tr><INPUT TYPE="button" NAME="listar" VALUE="Listar" style ="width: 13%; height: 5%; float: left;
157 background-color: #11FFCD; margin-left: 1%; filter: blur(2px);"></tr>
158 </form>
159 <form action="index.php" method="POST">
160 <tr><INPUT TYPE="button" NAME="buscar" VALUE="Buscar" style ="width: 13%; height: 5%; float: left;
161 background-color: #11FFA7; margin-left: 1%; filter: blur(2px);"></tr>
162 </form>
163 <form action="index.php" method="POST">
164 <tr><INPUT TYPE="button" NAME="borrar" VALUE="Borrar" style ="width: 13%; height: 5%; float: left;
165 background-color: #11FF48; margin-left: 1%; filter: blur(2px);"></tr>
166 </form>
167 <form action="index.php" method="POST">
168 <tr><INPUT TYPE="button" NAME="actualizar" VALUE="Actualizar" style ="width: 13%; height: 5%; float: left;
169 background-color: #9FFF11; margin-left: 1%; filter: blur(2px);"></tr>
170 <form action="index.php" method="POST">
171 <tr><INPUT TYPE="button" NAME="pagina" VALUE="Historial" style ="width: 13%; height: 5%; float: left;
172 background-color: #E3FF11; margin-left: 1%; filter: blur(2px);"></tr>
173 </form>
174 <form action="index.php" method="POST">
175 <tr><INPUT TYPE="button" NAME="salir" VALUE="Cerrar Sesion" style ="width: 13%; height: 5%; float: left;
176 background-color: #FF2D11; margin-left: 1%; filter: blur(2px);"></tr>
177 </form><br><br>
178 </table></form><br><br>
179 <div align="center" class="a" >
180 <div>
181 
182 <H3 align="center" >SUALPA'SPI</H3>
183 <form ACTION="registros.php" METHOD="POST">
184 <table align="center">
185 <tr>
186 <td>Usuario: </td> <td><INPUT TYPE="text" NAME="nombre" size=15><BR></td>
187 </tr>
188 <tr>
189 <td>Contraseña: </td><td> <INPUT TYPE="password" NAME="contra"size=15><BR></td>
190 </tr>
191 </table>
192 <br>
193 <INPUT TYPE="submit" name="enviar" VALUE="ENTRAR">
194 </div>
195 </div>
196 </form>
197 </body>
198 </html>
199 <?PHP
200 }
201 ?>
202 </BODY>
203 </HTML>

```

## insertar1.php

```
1 <html>
2 <head>
3   <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4   <script language="Javascript">
5     function copiarTexto(id_elemento) {
6       var aux = document.createElement("input");
7       aux.setAttribute("value", document.getElementById(id_elemento).innerHTML);
8       document.body.appendChild(aux);
9       aux.select();
10      document.execCommand("copy");
11      document.body.removeChild(aux);
12    }
13  </script>
14 </head>
15 <body bgcolor= "lightblue"> <br><br>
16 <div style ="background-color: #11F3FF; margin-left: 1%; border: 2px solid black; padding: 1%;">
17 <table>
18   <tr><td><label>Pase su id. Una vez aparezca la luz verde, presione continuar antes de que acabe la cuenta atras.</label></td></tr>
19   <td><INPUT TYPE="hidden" NAME="id" value="<?PHP print($id); ?>"></td></tr>
20 </table>
21 <p id="p1" style="visibility: hidden"><?PHP print($id); ?></p>
22 <form action="index.php" method="POST">
23   <INPUT TYPE="submit" NAME="copiar" VALUE="Aceptar">
24 </form>
25 </div>
26 </body>
27 </html>
```

## copiar.php

```
1 <html>
2 <head><meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
3 <script language="Javascript">
4   function copiarTexto(id_elemento) {
5     var aux = document.createElement("input");
6     aux.setAttribute("value", document.getElementById(id_elemento).innerHTML);
7     document.body.appendChild(aux);
8     aux.select();
9     document.execCommand("copy");
10    document.body.removeChild(aux);
11  }
12 </script>
13 </head>
14 <body bgcolor= "lightblue">
15   <br><br>
16 <div style ="background-color: #11F3FF; margin-left: 1%; border: 2px solid black; padding: 1%;">
17   <?PHP
18   $leer = fopen("/home/pi/Desktop/lecturaid.txt", "r");
19   while(!feof($leer)){
20     $linea=fgets($leer);
21     $a=$linea;
22   }
23   fclose($leer);
24   $id = $a;
25   if ($id>0){
26     ?>
27     <table>
28     </table>
29     <form ACTION="index.php" METHOD="POST">
30     <table>
31     <tr><td><label>ID RECONOCIDO CORRECTAMENTE. <br>Pulse aceptar para continuar.</label></td>
32     <td><INPUT TYPE="hidden" NAME="id" value="<?PHP print($id); ?>"></td><td>
33     <p id="p1" style="visibility: hidden"><?PHP print($id); ?></p></td></tr>
34     </table><br>
35     <button name="insertar" onclick="copiarTexto('p1')">Copiar y Continuar</button>
36     </form> <form action="index.php" method="POST"><INPUT TYPE="submit" NAME="index" VALUE="Volver al inicio">
37     <?PHP
38   }
39   else{
40     ?>
41     <table>
42     <tr><td><label>ID NO RECONOCIDO. <br>Pulse aceptar para reintentar.</td>
43     <td><INPUT TYPE="hidden" NAME="id" value="<?PHP print($id); ?>"></td></tr>
44     </table>
45     <br>
46     <form action="index.php" method="POST">
47     <INPUT TYPE="submit" NAME="insertar1" VALUE="Aceptar">
48     </form>
49     <?PHP
50   }
51   ?>
52 </div>
53 </body>
54 </html>
```

## insertar.php

```
1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 <script language="Javascript">
5 function copiarTexto(id_elemento) {
6     var aux = document.createElement("input");
7     aux.setAttribute("value", document.getElementById(id_elemento).innerHTML);
8     document.body.appendChild(aux);
9     aux.select();
10    document.execCommand("copy");
11    document.body.removeChild(aux);
12 }
13 </script>
14 </head>
15 <body bgcolor= "lightblue"><br><br>
16 <div style ="background-color: #11F3FF; margin-left: 1%; border: 2px solid black; padding: 1%;">
17 <?PHP
18     if (isset($_REQUEST['insertar'])) {
19         $id = $_REQUEST['id'];
20     }
21     ?>
22 <form ACTION="index.php" METHOD="POST">
23 <table>
24 <tr><td colspan=2><b>Rellene los siguientes datos:</b></td><td style="width: 8%;></td>
25 <td colspan=2><b>Marque los accesos permitidos:</b></td></tr>
26 <tr style="height: 15px;"><td colspan=4></td><td><p id="p1" style="visibility: hidden"> </p></td></tr>
27 <tr><td><label>Nombre:</label></td><td><input type="text" NAME="nombre"></td></tr>
28 <tr><td><label>Entrada:</label></td><td><input type="checkbox" name="entrada" checked><br></td></tr>
29 <tr><td><label>Apellido:</label></td><td><input type="text" NAME="apellido"></td></tr>
30 <tr><td><label>Administracion:</label></td><td><input type="checkbox" name="administracion"><br></td></tr>
31 <tr><td><label>DNI:</label></td><td><input type="text" NAME="dni"></td></tr>
32 <tr><td><label>RRHH:</label></td><td><input type="checkbox" name="rrhh"><br></td></tr>
33 <tr><td><label>Correo:</label></td><td><input type="email" NAME="correo"></td></tr>
34 <tr><td><label>I+D:</label></td><td><input type="checkbox" name="imasd"><br></td></tr>
35 <tr><td><label>Pegue aqui su ID:</label></td><td><input type="password" NAME="id" value="<?PHP print($id); ?>"></td>
36 <tr><td><label>Reuniones:</label></td><td><input type="checkbox" name="reuniones"><br></td></tr>
37 </table>
38 <br><button name="insertar3" onclick="copiarTexto('p1')>Continuar</button>
39 </form> <form action="index.php" method="POST"><input type="submit" NAME="index" VALUE="Volver al inicio">
40 </form>
41 </div>
42 </body>
43 </html>
```

## insertar3.php

```
1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 </head>
5 <body bgcolor= "lightblue">
6 <br><br>
7 <div style ="background-color: #11F3FF; margin-left: 1%; border: 2px solid black; padding: 1%;">
8 <?PHP
9     if (isset($_REQUEST['insertar3']))
10    {
11        $id = $_REQUEST['id'];
12        $nombre = $_REQUEST['nombre'];
13        $apellido = $_REQUEST['apellido'];
14        $dni = $_REQUEST['dni'];
15        $correo = $_REQUEST['correo'];
16        $entrada = $_REQUEST["puerta_principal"];
17        $administracion = $_REQUEST["dpto_administracion"];
18        $rrhh = $_REQUEST["dpto_rrhh"];
19        $imasd = $_REQUEST["dpto_imasd"];
20        $reuniones = $_REQUEST["sala_de_reuniones"];
21        $errores = "";
```

```

22     if (trim($nombre) == "")
23         $errores = $errores . " <LI>Se requiere el nombre\n";
24     if (is_numeric($nombre))
25         $errores = $errores . " <LI>El nombre no puede contener numeros\n";
26     if (trim($apellido) == "")
27         $errores = $errores . " <LI>Se requiere el apellido\n";
28     if (is_numeric($apellido))
29         $errores = $errores . " <LI>El apellido no puede contener numeros\n";
30     if (trim($dni) == "")
31         $errores = $errores . " <LI>Se requiere el DNI\n";
32     if (trim($correo) == "")
33         $errores = $errores . " <LI>Se requiere el correo\n";
34     if (trim($id) == "")
35         $errores = $errores . " <LI>Se requiere el ID\n";
36     if (!is_numeric($id))
37         $errores = $errores . " <LI>El ID no puede contener letras\n";
38     if ($errores != "")
39     {
40         print ("<H3>No se ha podido realizar la inserción debido a los siguientes errores:</H3>\n");
41         print ("<UL>\n");
42         print ($errores);
43         print ("</UL>\n");
44     }
45     else
46     {
47         $hostname_db = "localhost";
48         $database_db = "usuarios";
49         $username_db = "ja";
50         $password_db = "rootroot";
51         $conexion = mysqli_connect($hostname_db, $username_db, $password_db, $database_db);
52         if (!$conexion) {
53             die ("La conexión ha fallado:" . mysqli_connect_error());
54         }
55         $id = $_REQUEST['id'];
56         $nombre = $_REQUEST['nombre'];
57         $apellido = $_REQUEST['apellido'];
58         $dni = $_REQUEST['dni'];
59         $correo = $_REQUEST['correo'];
60         if (isset($_REQUEST['entrada'])) {
61             $entrada = 1;
62         }
63         else {
64             $entrada=0;
65         }
66         if (isset($_REQUEST['administracion'])) {
67             $administracion = 1;
68         }
69         else {
70             $administracion=0;
71         }
72         if (isset($_REQUEST['rrhh'])) {
73             $rrhh = 1;
74         }
75         else {
76             $rrhh=0;
77         }
78         if (isset($_REQUEST['imasd'])) {
79             $imasd = 1;
80         }
81         else {
82             $imasd=0;
83         }
84         if (isset($_REQUEST['reuniones'])) {
85             $reuniones = 1;
86         }
87         else {
88             $reuniones=0;
89         }
90         $saccion = "insert into admitidos (`id`, `nombre`, `apellido`, `dni`, `correo`, `puerta_principal`,
91 `dpto_administracion`, `dpto_rrhh`, `dpto_imasd`, `sala_de_reuniones`) values ($id,$nombre`,
92 `$apellido`, `$dni`, `$correo`, $entrada, $administracion, $rrhh, $imasd, $reuniones)";
93         if (mysqli_query($conexion, $saccion) ) {
94             echo "Se ha insertado correctamente";
95         }
96         else {
97             echo "Error al insertar: Usted se dejó campos sin rellenar o los rellenó indebidamente.";
98         }
99         mysqli_close($conexion);
100     }
101 }
102 ?>
103 <form action="index.php" method="POST">
104 <br><INPUT TYPE="submit" NAME="index" VALUE="Volver al inicio">
105 </form>
106 </form>
107 </div>
108 </body>
109 </html>

```

## listar.php

```
1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 <style>
5 table {
6     width:100%;
7     border-collapse: collapse;
8     border: 1px solid black;
9 }
10 td {
11     border-collapse: collapse;
12     border: 1px solid black;
13 }
14 tr {
15     border-collapse: collapse;
16     border: 1px solid black;
17 }
18 .a {
19     background-color: #EDE61C ;
20 }
21 </style>
22 </head>
23 <body bgcolor= "lightblue">
24 <br><br><div style ="background-color: #11FFCD; margin-left: 1%; border: 2px solid black; padding: 1%;">
25 <?PHP
26 $hostname_db = "localhost";
27 $database_db = "usuarios";
28 $username_db = "ja";
29 $password_db = "rootroot";
30 $conexion = mysqli_connect($hostname_db, $username_db, $password_db, $database_db);
31 if (!$conexion) {
32     die ("La conexion ha fallado:" . mysqli_connect_error());
33 }
34 $pregunta = "select * from admitidos";
35 $resultado = mysqli_query($conexion,$pregunta);
36 if (mysqli_num_rows($resultado) > 0 ) {
37     //MUESTRA LOS DATOS FILA FILA
38     echo "<form ACTION='select.php' METHOD='POST'><table background='im.jpg'>";
39     echo " <tr align=center class='a' >
40         <td class='a'>N_de_usuario</td>
41         <td class='a'>ID</td>
42         <td class='a'>Nombre</td>
43         <td class='a'>Apellido</td>
44         <td class='a'>DNI</td>
45         <td class='a'>Correo</td>
46         <td class='a'>Puerta Principal</td>
47         <td class='a'>Dpto. Administracion</td>
48         <td class='a'>Dpto. RRHH</td>
49         <td class='a'>Dpto. I+D</td>
50         <td class='a'>Sala de Reuniones</td>
51     </tr>";
52     while ($row = mysqli_fetch_assoc($resultado)) {
53         echo "<tr >
54             <td align=center>$row[numero_de_usuario]</td>
55             <td align=center>$row[id]</td>
56             <td align=center>$row[nombre]</td>
57             <td align=center>$row[apellido]</td>
58             <td align=center>$row[dni]</td>
59             <td align=center>$row[correo]</td>
60             <td align=center>; if ($row[puerta_principal]==1){echo "Permitido";}else{echo "Denegado";} echo "</td>
61             <td align=center>; if ($row[dpto_administracion]==1){echo "Permitido";}else{echo "Denegado";} echo "</td>
62             <td align=center>; if ($row[dpto_rrhh]==1){echo "Permitido";}else{echo "Denegado";} echo "</td>
63             <td align=center>; if ($row[dpto_imasd]==1){echo "Permitido";}else{echo "Denegado";} echo "</td>
64             <td align=center>; if ($row[sala_de_reuniones]==1){echo "Permitido";}else{echo "Denegado";} echo "</td>
65         </tr>";
66     }
67     echo "</table> </form> ";
68 } else {
69     echo "Ningun resultado";
70 }
71 }
72 mysqli_close($conexion);
73 <?>
74 <form action="index.php" method="POST">
75 <INPUT TYPE="submit" NAME="index" VALUE="Volver al inicio">
76 </form>
77 </div>
78 </body>
79 </html>
```

## buscar.php

```
1 <HTML>
2 <HEAD>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 </HEAD>
5 <BODY bgcolor= "lightblue"> <br><br>
6 <div style="background-color: #1FFFA7; margin-left: 1%; border: 2px solid black; padding: 1%;">
7 <form action="index.php" method="POST">
8 <table>
9 <tr><td colspan=5><b>Rellene los datos para buscar coincidencias:</b></td><td style="width: 8%;></td></tr>
10 <tr style="height: 15px;><td colspan=4></td></tr>
11 <tr><td><label>ID:</label></td><td><INPUT TYPE="text" NAME="id"></td><td style="width: 8%;></td>
12 <td><label>Acceso a la puerta principal:</label></td>
13 <td><INPUT TYPE="radio" NAME="principal" value="1">Permitido<INPUT TYPE="radio" NAME="principal" value="0">Denegado<br></td></tr>
14 <tr><td><label>Nombre:</label></td><td><INPUT TYPE="text" NAME="nombre"></td><td></td>
15 <td><label>Acceso al departamento de administracion:</label></td>
16 <td><input type="radio" name="administracion" value="1">Permitido<input type="radio" name="administracion" value="0">Denegado<br></td></tr>
17 <tr><td><label>Apellido:</label></td><td><INPUT TYPE="text" NAME="apellido"></td><td><label>Acceso al departamento de RRHH:</label></td>
18 <td><input type="radio" name="rrhh" value="1">Permitido<input type="radio" name="rrhh" value="0">Denegado<br></td></tr>
19 <tr><td><label>DNI:</label></td><td><INPUT TYPE="text" NAME="dni"></td><td><label>Acceso al departamento de I+D:</label></td>
20 <td><input type="radio" name="imasd" value="1">Permitido<input type="radio" name="imasd" value="0">Denegado<br></td></tr>
21 <tr><td><label>Correo:</label></td><td><INPUT TYPE="text" NAME="correo"></td><td><label>Acceso al salon de reuniones:</label></td>
22 <td><input type="radio" name="reuniones" value="1">Permitido<input type="radio" name="reuniones" value="0">Denegado<br></td></tr>
23 </table>
24 <br><INPUT TYPE="submit" NAME="buscar2" VALUE="Buscar"><br><br>
25 <form action="index.php" method="POST">
26 <INPUT TYPE="submit" NAME="index" VALUE="Volver al inicio">
27 </form>
28 </form>
29 </div>
30 </BODY>
31 </HTML>
```

## buscar2.php

```
1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 <style>
5 table {
6 width:100%;
7 border-collapse: collapse;
8 border: 1px solid black;
9 background-color: #FFFEE3
10 }
11 td {
12 background-color: #FFFEE3
13 border-collapse: collapse;
14 border: 1px solid black;
15 }
16 tr {
17 background-color: #FFFEE3
18 border-collapse: collapse;
19 border: 1px solid black;
20 }
21 .a {
22 background-color: #EDE61C ;
23 }
24 </style>
25 </head>
26 <body bgcolor= "lightblue">
27 <br><br><div style="background-color: #1FFFA7; margin-left: 1%; border: 2px solid black; padding: 1%;">
28 <?PHP
29 $hostname_db = "localhost";
30 $database_db = "usuarios";
31 $username_db = "ja";
32 $password_db = "rootroot";
33 $conexion = mysqli_connect($hostname_db, $username_db, $password_db, $database_db);
34 if (!$conexion) {
35 die ("La conexion ha fallado:" . mysqli_connect_error());
36 }
37 $id = $_REQUEST["id"];
38 $nombre = $_REQUEST["nombre"];
39 $apellido = $_REQUEST["apellido"];
40 $dni = $_REQUEST["dni"];
41 $correo = $_REQUEST["correo"];
42 $principal = $_REQUEST["principal"];
43 $administracion = $_REQUEST["administracion"];
44 $rrhh = $_REQUEST["rrhh"];
45 $imasd = $_REQUEST["imasd"];
46 $reuniones = $_REQUEST["reuniones"];
47 $z="";
```

```

48  if ($id!=$z){
49      $a="and id = '$id'";
50  }
51  else {
52      $a="";
53  }
54  if ($nombre!=$z){
55      $b="and nombre = '$nombre'";
56  }
57  else {
58      $b="";
59  }
60  if ($apellido!=$z){
61      $c="and apellido = '$apellido'";
62  }
63  else {
64      $c="";
65  }
66  if ($dni!=$z){
67      $d="and dni = '$dni'";
68  }
69  else {
70      $d="";
71  }
72  if ($correo!=$z){
73      $e="and correo = '$correo'";
74  }
75  else {
76      $e="";
77  }
78  if ($principal!=$z){
79      $f="and puerta_principal = '$principal'";
80  }
81  else {
82      $f="and puerta_principal >= 0";
83  }
84
85  if ($administracion!=$z){
86      $g="and dpto_administracion = '$administracion'";
87  }
88  else {
89      $g="and dpto_administracion >= 0";
90  }
91  if ($rrhh!=$z){
92      $h="and dpto_rrhh = '$rrhh'";
93  }
94  else {
95      $h="and dpto_rrhh >= 0";
96  }
97  if ($imasd!=$z){
98      $i="and dpto_imasd = '$imasd'";
99  }
100 else {
101     $j="and dpto_imasd >= 0";
102 }
103
104 if ($reuniones!=$z){
105     $k="and sala_de_reuniones = '$reuniones'";
106 }
107 else {
108     $k="and sala_de_reuniones >= 0";
109 }
110 $preguntafinal= "select * from admitidos where numero_de_usuario $a $b $c $d $e $f $g $h $i $j $k ;";
111 $resultado = mysqli_query($conexion,$preguntafinal);

```

```

112 if (mysqli_num_rows($resultado) > 0 ) {
113     echo "<form ACTION='buscar2.php' METHOD='POST'><table border=1 width=30% background='im.jpg'>";
114     echo " <tr align=center class='a'>
115         <td>N_de_usuario</td>
116         <td>ID</td>
117         <td>Nombre</td>
118         <td>Apellido</td>
119         <td>DNI</td>
120         <td>Correo</td>
121         <td>Puerta Principal</td>
122         <td>Dpto. Administracion</td>
123         <td>Dpto. RRHH</td>
124         <td>Dpto. I+D</td>
125         <td>Sala de Reuniones</td>
126     </tr>";
127     while ($row = mysqli_fetch_assoc($resultado)) {
128         echo "<tr >
129             <td>$row[numero_de_usuario]</td>
130             <td>$row[id]</td>
131             <td>$row[nombre]</td>
132             <td>$row[apellido]</td>
133             <td>$row[dni]</td>
134             <td>$row[correo]</td>
135             <td>"; if ($row[puerta_principal]==1){echo "Permitido";}else{echo "Denegado";} echo "</td>
136             <td>"; if ($row[dpto_administracion]==1){echo "Permitido";}else{echo "Denegado";} echo "</td>
137             <td>"; if ($row[dpto_rrhh]==1){echo "Permitido";}else{echo "Denegado";} echo "</td>
138             <td>"; if ($row[dpto_imasd]==1){echo "Permitido";}else{echo "Denegado";} echo "</td>
139             <td>"; if ($row[sala_de_reuniones]==1){echo "Permitido";}else{echo "Denegado";} echo "</td>
140         </tr>" ;
141     }
142     echo "</table> </form> ";
143 } else {
144     echo "Ningun resultado";
145 }
146 mysqli_close($conexion);
147 ?>
148 <form action="index.php" method="POST">
149 <INPUT TYPE="submit" NAME="index" VALUE="Volver al inicio">
150 </form>
151 </div>
152 </body>
153 </html>

```

## actualizar.php

```

1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 <style>
5     table {
6         width:100%;
7         border-collapse: collapse;
8         border: 1px solid black;
9     }
10     td {
11         border-collapse: collapse;
12         border: 1px solid black;
13     }
14     tr {
15         border-collapse: collapse;
16         border: 1px solid black;
17     }
18     .a {
19         background-color: #EDE61C ;
20     }
21 </style>
22 </head>
23 <body bgcolor= "lightblue">
24 <br><br><div style ="background-color: #9FFF11; margin-left: 1%; border: 2px solid black; padding: 1%;">
25 <?PHP
26 $hostname_db = "localhost";
27 $database_db = "usuarios";
28 $username_db = "ja";
29 $password_db = "rootroot";
30 $conexion = mysqli_connect($hostname_db, $username_db, $password_db, $database_db);
31 if (!$conexion) {
32     die ("La conexion ha fallado:" . mysqli_connect_error());
33 }
34 $pregunta = "select * from admitidos";
35 $resultado = mysqli_query($conexion,$pregunta);

```



```

36 if (mysqli_num_rows($resultado) > 0 ) {
37     echo "<form ACTION='index.php' METHOD='POST'><table width=50% background='im.jpg'>";
38     echo " <tr align=center class='a' >
39         <td class='a'>N_de_usuario</td>
40         <td class='a'>ID</td>
41         <td class='a'>Nombre</td>
42         <td class='a'>Apellido</td>
43         <td class='a'>DNI</td>
44         <td class='a'>Correo</td>
45         <td class='a'>Puerta Principal</td>
46         <td class='a'>Dpto. Administracion</td>
47         <td class='a'>Dpto. RRHH</td>
48         <td class='a'>Dpto. I+D</td>
49         <td class='a'>Sala de Reuniones</td>
50         <td class='a'></td>
51     </tr>";
52     while ($row = mysqli_fetch_assoc($resultado)) {
53         echo "<tr >
54             <td align=center>$row[numero_de_usuario]</td>
55             <td align=center>$row[id]</td>
56             <td align=center>$row[nombre]</td>
57             <td align=center>$row[apellido]</td>
58             <td align=center>$row[dni]</td>
59             <td align=center>$row[correo]</td>
60             <td align=center>; if ($row[puerta_principal]==1){echo "Permitido";}else{echo "Denegado";} echo "</td>
61             <td align=center>; if ($row[dpto_administracion]==1){echo "Permitido";}else{echo "Denegado";} echo "</td>
62             <td align=center>; if ($row[dpto_rrhh]==1){echo "Permitido";}else{echo "Denegado";} echo "</td>
63             <td align=center>; if ($row[dpto_imasd]==1){echo "Permitido";}else{echo "Denegado";} echo "</td>
64             <td align=center>; if ($row[sala_de_reuniones]==1){echo "Permitido";}else{echo "Denegado";} echo "</td>";
65             <td align=center><input type='checkbox' name='numuser[]' value='<?PHP echo $row['numero_de_usuario']; ?>'></td>
66         <?PHP
67         echo "</tr>" ;
68     }
69     echo "</table><br><input type='submit' name='actualizar2' value='Siguiete'> </form> ";
70 } else {
71     echo "Ningun resultado";
72 }
73 mysqli_close($conexion);
74 ?>
75 <form action="index.php" method="POST">
76 <INPUT TYPE="submit" NAME="index" VALUE="Volver al inicio">
77 </form>
78 </div>
79 </body>
80 </html>

```

## actualizar2.php

```

1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 </head>
5 <body bgcolor= "lightblue">
6 <br><br>
7 <div style ="background-color: #9FFF11; margin-left: 1%; border: 2px solid black; padding: 1%;">
8 <?PHP
9 $hostname_db = "localhost";
10 $database_db = "usuarios";
11 $username_db = "ja";
12 $password_db = "rootroot";
13 $conexion = mysqli_connect($hostname_db, $username_db, $password_db, $database_db);
14 if (!$conexion) {
15     die ("La conexion ha fallado:" . mysqli_connect_error());
16 }
17 if(isset($_REQUEST['actualizar3'])){
18     $id = $_REQUEST["id"];
19     $nombre= $_REQUEST["nombre"];
20     $apellido = $_REQUEST["apellido"];
21     $dni = $_REQUEST["dni"];
22     $correo = $_REQUEST["correo"];
23     $principal = $_REQUEST["puerta_principal"];
24     $admin = $_REQUEST["dpto_administracion"];
25     $rrhh = $_REQUEST["dpto_rrhh"];
26     $imasd = $_REQUEST["dpto_imasd"];
27     $reuniones = $_REQUEST["sala_de_reuniones"];
28     $z="";
29     $updateid= implode ('', $_REQUEST['numuser']);
30     if ($id == "") {
31         $a="";
32     }
33     else if ($id != "")
34     {
35         $a="id='".$id"'";
36     }

```

```

37 else if ($nombre != $z or $apellido != $z or $dni != $z or $correo != $z or $principal != $z
38 or $admin != $z or $rrhh != $z or $masd != $z or $reuniones != $z)
39 {
40     $a = "id='$id'";
41 }
42 if ($nombre == "")
43 {
44     $b="";
45 }
46 else if ($id != $z )
47 {
48     $b = ", nombre='$nombre'";
49 }
50 else if ($nombre != $z or $id == $z )
51 {
52     $b = "nombre='$nombre'";
53 }
54 if ($apellido == "")
55 {
56     $c="";
57 }
58 else if ($id != $z or $nombre != $z){
59     $c = ", apellido='$apellido'";
60 }
61 else if ($apellido != $z or $id == $z or $nombre == $z )
62 {
63     $c = "apellido='$apellido'";
64 }
65 if ($dni == "") {
66     $d="";
67 }
68 else if ($id != $z or $nombre != $z or $apellido != $z)
69 {
70     $d = ", dni='$dni'";
71 }
72 else if ($dni != $z or $id == $z or $nombre == $z or $apellido == $z)
73 {
74     $d = "dni='$dni'";
75 }
76 if ($correo == ""){
77     $e="";
78 }
79 else if ($id != $z or $nombre != $z or $apellido != $z or $dni != $z)
80 {
81     $e = ", correo='$correo'";
82 }
83 else if ($dni!= $z or $usuario == $z or $contraseña == $z
84 or $nombreu == $z or $dni == $z)
85 {
86     $e = "correo='$correo'";
87 }
88 if ($principal == ""){
89     $f="";
90 }
91 else if ($id != $z or $nombre != $z or $apellido != $z or $dni != $z or $correo != $z)
92 {
93     $f = ", puerta_principal='$principal'";
94 }
95 else if ($principal!= $z or $id == $z or $nombre == $z or $apellido == $z or $dni == $z
96 or $correo == $z )
97 {
98     $f = "puerta_principal='$principal'";
99 }
100 if ($admin == "")
101 {
102     $g="";
103 }
104 else if ($id != $z or $nombre != $z or $apellido != $z or $dni != $z or $correo != $z
105 or $principal != $z)
106 {
107     $g = ", dpto_administracion='$admin'";
108 }
109 else if ($admin!= $z or $id == $z or $nombre == $z or $apellido == $z or $dni == $z
110 or $correo == $z or $principal == $z )
111 {
112     $g = "dpto_administracion='$admin'";
113 }
114 if ($rrhh == ""){
115     $h="";
116 }

```

```

117 else if ($id != $z or $nombre != $z or $apellido != $z or $dni != $z or $correo != $z
118 or $principal != $z or $admin != $z)
119 {
120     $h = ", dpto_rrhh='$rrhh'";
121 }
122 else if ($rrhh!= $z or $id == $z or $nombre == $z or $apellido == $z or $dni == $z
123 or $correo == $z or $principal == $z or $admin == $z)
124 {
125     $h = "dpto_rrhh='$rrhh'";
126 }
127 if ($imasd == ""){
128     $i="";
129 }
130 else if ($id != $z or $nombre != $z or $apellido != $z or $dni != $z or $correo != $z
131 or $principal != $z or $admin != $z or $rrhh != $z )
132 {
133     $i = ", dpto_imasd='$imasd'";
134 }
135 else if ($imasd!= $z or $id == $z or $nombre == $z or $apellido == $z or $dni == $z
136 or $correo == $z or $principal == $z or $admin == $z or $rrhh == $z )
137 {
138     $i = "dpto_imasd='$imasd'";
139 }
140 if ($reuniones == "")
141 {
142     $j="";
143 }
144 else if ($id != $z or $nombre != $z or $apellido != $z or $dni != $z or $correo != $z
145 or $principal != $z or $admin != $z or $rrhh != $z or $imasd != $z )
146 {
147     $j = ", sala_de_reuniones='$reuniones'";
148 }
149 else if ($reuniones!= $z or $id == $z or $nombre == $z or $apellido == $z or $dni == $z
150 or $correo == $z or $principal == $z or $admin == $z or $rrhh == $z or $imasd == $z )
151 {
152     $j = "sala_de_reuniones='$reuniones'";
153 }
154 $saccion = "update admitidos set $a $b $c $d $e $f $g $h $i $j where numero_de_usuario
155 in ($updateid) ";
156 if (mysqli_query($conexion,$saccion)) {
157     echo "Se ha actualizado correctamente";
158 } else {
159     echo "Error al actualizar: Usted no ha seleccionado a nadie";
160 }
161 mysqli_close($conexion);
162 }
163 else {
164     ?>
165     <form action="index.php" method="GET">
166         <table>
167             <tr><td><label>no:</label></td><td><INPUT TYPE="text" NAME="numuser[]" value="<?PHP
168             $updateid= implode (',',$_REQUEST['numuser']); echo "$updateid"; ?>"</td></tr>
169             <tr><td><label>ID:</label></td><td><INPUT TYPE="text" NAME="id"></td></tr>
170             <tr><td><label>Nombre:</label></td><td><INPUT TYPE="text" NAME="nombre"></td></tr>
171             <tr><td><label>Apellido:</label></td><td><INPUT TYPE="text" NAME="apellido"></td></tr>
172             <tr><td><label>DNI:</label></td><td><INPUT TYPE="text" NAME="dni"></td></tr>
173             <tr><td><label>Correo:</label></td><td><INPUT TYPE="text" NAME="correo"></td></tr>
174             <tr><td><label>Puerta Principal:</label></td>
175             <td><INPUT TYPE="radio" NAME="puerta_principal" value="1">
176             Permitir<INPUT TYPE="radio" NAME="puerta_principal" value="0">Denegar</td></tr>
177             <tr><td><label>Dpto. Administracion:</label></td>
178             <td><INPUT TYPE="radio" NAME="dpto_administracion" value="1">
179             Permitir<INPUT TYPE="radio" NAME="dpto_administracion" value="0">Denegar</td></tr>
180             <tr><td><label>Dpto. RRHH:</label></td>
181             <td><INPUT TYPE="radio" NAME="dpto_rrhh" value="1">
182             Permitir<INPUT TYPE="radio" NAME="dpto_rrhh" value="0">Denegar</td></tr>
183             <tr><td><label>Dpto. I+D:</label></td>
184             <td><INPUT TYPE="radio" NAME="dpto_imasd" value="1">
185             Permitir<INPUT TYPE="radio" NAME="dpto_imasd" value="0">Denegar</td></tr>
186             <tr><td><label>Sala de Reuniones:</label></td>
187             <td><INPUT TYPE="radio" NAME="sala_de_reuniones" value="1">
188             Permitir<INPUT TYPE="radio" NAME="sala_de_reuniones" value="0">Denegar</td></tr>
189         </table>
190         <br><INPUT TYPE="submit" NAME="actualizar3" VALUE="Actualizar">
191     </form>
192 </div>
193 <?PHP
194 }
195 ?>
196 </body>

```

## historial.php

```
1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 <style>
5 table {
6     border-collapse: collapse;
7     border: 1px solid black;
8 }
9 td {
10     background-color: #FFFEE3
11     border-collapse: collapse;
12     border: 1px solid black;
13     align:center;
14     width:5%;
15 }
16 tr {
17     background-color: #FFFEE3
18     border-collapse: collapse;
19     border: 1px solid black;
20 }
21 .a {
22     background-color: #EDE61C ;
23 }
24 </style>
25 </head>
26 <body bgcolor= "lightblue"><br><br><br>
27 <table>
28 <form action="index.php" method="POST">
29 <tr><INPUT TYPE="submit" NAME="general" VALUE="General" style ="width: 13%; height: 5%;
30 float: left; background-color: #E3FF11; margin-left: 1%"></tr>
31 </form>
32 <form action="index.php" method="POST">
33 <tr><INPUT TYPE="submit" NAME="principal" VALUE="Puerta Principal" style ="width: 13%; height: 5%;
34 float: left; background-color: #E3FF11; margin-left: 1%"></tr>
35 </form>
36 <form action="index.php" method="POST">
37 <tr><INPUT TYPE="submit" NAME="admin" VALUE="Dpto. Administracion" style ="width: 13%; height: 5%;
38 float: left; background-color: #E3FF11; margin-left: 1%"></tr>
39 </form>
40 <form action="index.php" method="POST">
41 <tr><INPUT TYPE="submit" NAME="rrhh" VALUE="Dpto. RRHH" style ="width: 13%; height: 5%;
42 float: left; background-color: #E3FF11; margin-left: 1%"></tr>
43 </form>
44 <form action="index.php" method="POST">
45 <tr><INPUT TYPE="submit" NAME="imasd" VALUE="Dpto. I+D" style ="width: 13%; height: 5%;
46 float: left; background-color: #E3FF11; margin-left: 1%"></tr>
47 </form>
48 <form action="index.php" method="POST">
49 <tr><INPUT TYPE="submit" NAME="sala" VALUE="Sala de Reuniones" style ="width: 13%; height: 5%;
50 float: left; background-color: #E3FF11; margin-left: 1%"></tr>
51 </form>
52 <form action="index.php" method="POST">
53 <tr><INPUT TYPE="submit" NAME="denegados" VALUE="Denegados" style ="width: 13%; height: 5%;
54 float: left; background-color: #E3FF11; margin-left: 1%"></tr>
55 </form>
56 </table>
57 </body>
58 </html>
```

## general.php

```
1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 <style>
5 table {
6     border-collapse: collapse;
7     border: 1px solid black;
8 }
9 td {
10     background-color: #FFFEE3
11     border-collapse: collapse;
12     border: 1px solid black;
13     align:center;
14     width:5%;
15 }
16 tr {
17     background-color: #FFFEE3
18     border-collapse: collapse;
19     border: 1px solid black;
20 }
21 .a {
22     background-color: #EDE61C ;
23 }
24 </style>
25 </head>
26 <body bgcolor= "lightblue">
27 <br><br><br><br><div style ="background-color: #11FFCD; margin-left: 1%; border: 2px solid black; padding: 1%;">
28 <?PHP
29 $hostname_db = "localhost";
30 $database_db = "usuarios";
31 $username_db = "ja";
32 $password_db = "rootroot";
33 $conexion = mysqli_connect($hostname_db, $username_db, $password_db, $database_db);
34 if (!$conexion) {
35     die ("La conexion ha fallado:" . mysqli_connect_error());
36 }
37 $pregunta = "select * from historial";
38 $resultado = mysqli_query($conexion,$pregunta);
39 if (mysqli_num_rows($resultado) > 0 ) {
40     echo "<form ACTION='general.php' METHOD='POST'><table background='im.jpg'>";
41     echo " <tr align=center class='a' >
42         <td class='a'>Numero de apertura de la puerta</td>
43         <td class='a'>N_de_usuario</td>
44         <td class='a'>Fecha y hora de entrada</td>
45     </tr>";
46     while ($row = mysqli_fetch_assoc($resultado)) {
47         echo "<tr >
48             <td align=center>$row[id]</td>
49             <td align=center>$row[numero_de_usuario]</td>
50             <td align=center>$row[fecha_hora]</td>
51         </tr>" ;
52     }
53     echo "</table> </form> ";
54 } else {
55     echo "Ningun resultado";
56 }
57 mysqli_close($conexion);
58 ?>
59 <form action="index.php" method="POST">
60 <INPUT TYPE="submit" NAME="index" VALUE="Volver al inicio">
61 </form>
62 </div>
63 </body>
64 </html>
```

## principal.php

```
1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 <style>
5 table {
6     border-collapse: collapse;
7     border: 1px solid black;
8 }
9 td {
10     background-color: #FFFEE3
11     border-collapse: collapse;
12     border: 1px solid black;
13     align:center;
14     width:5%;
15 }
16 tr {
17     background-color: #FFFEE3
18     border-collapse: collapse;
19     border: 1px solid black;
20 }
21 .a {
22     background-color: #EDE61C ;
23 }
24 </style>
25 </head>
26 <body bgcolor= "lightblue">
27 <br><br><br><br><div style = "background-color: #11FFCD; margin-left: 1%; border: 2px solid black; padding: 1%;">
28 <?PHP
29 $hostname_db = "localhost";
30 $database_db = "usuarios";
31 $username_db = "ja";
32 $password_db = "rootroot";
33 $conexion = mysqli_connect($hostname_db, $username_db, $password_db, $database_db);
34 if (!$conexion) {
35     die ("La conexion ha fallado:" . mysqli_connect_error());
36 }
37 $pregunta = "select * from historial_p";
38 $resultado = mysqli_query($conexion,$pregunta);
39 if (mysqli_num_rows($resultado) > 0 ) {
40     echo "<form ACTION='imasd.php' METHOD='POST'><table background='im.jpg'>";
41     echo " <tr align=center class='a' >
42         <td class='a'>Numero de apertura de la puerta</td>
43         <td class='a'>N_de_usuario</td>
44         <td class='a'>Fecha y hora de entrada</td>
45     </tr>";
46     while ($row = mysqli_fetch_assoc($resultado)) {
47         echo "<tr >
48             <td align=center>$row[id]</td>
49             <td align=center>$row[numero_de_usuario]</td>
50             <td align=center>$row[fecha_hora]</td>
51         </tr>";
52     }
53     echo "</table> </form> ";
54 } else {
55     echo "Ningun resultado";
56 }
57 mysqli_close($conexion)
58 ?>
59
60 <form action="index.php" method="POST">
61 <INPUT TYPE="submit" NAME="index" VALUE="Volver al inicio">
62 </form>
63 </div>
64 </body>
65 </html>
```

## admin.php

```
1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 <style>
5 table {
6     border-collapse: collapse;
7     border: 1px solid black;
8 }
9 td {
10     background-color: #FFFEE3
11     border-collapse: collapse;
12     border: 1px solid black;
13     align:center;
14     width:5%;
15 }
16 tr {
17     background-color: #FFFEE3
18     border-collapse: collapse;
19     border: 1px solid black;
20 }
21 .a {
22     background-color: #EDE61C ;
23 }
24 </style>
25 </head>
26 <body bgcolor= "lightblue">
27 <br><br><br><br><div style ="background-color: #11FFCD; margin-left: 1%; border: 2px solid black; padding: 1%;">
28 <?PHP
29 $hostname_db = "localhost";
30 $database_db = "usuarios";
31 $username_db = "ja";
32 $password_db = "rootroot";
33 $conexion = mysqli_connect($hostname_db, $username_db, $password_db, $database_db);
34 if (!$conexion) {
35     die ("La conexion ha fallado:" . mysqli_connect_error());
36 }
37 $pregunta = "select * from historial_a";
38 $resultado = mysqli_query($conexion,$pregunta);
39 if (mysqli_num_rows($resultado) > 0 ) {
40     echo "<form ACTION='imasd.php' METHOD='POST'><table background='im.jpg'>";
41     echo " <tr align=center class='a' >
42         <td class='a'>Numero de apertura de la puerta</td>
43         <td class='a'>N_de_usuario</td>
44         <td class='a'>Fecha y hora de entrada</td>
45     </tr>";
46     while ($row = mysqli_fetch_assoc($resultado)) {
47         echo "<tr >
48             <td align=center>$row[id]</td>
49             <td align=center>$row[numero_de_usuario]</td>
50             <td align=center>$row[fecha_hora]</td>
51         </tr>";
52     }
53     echo "</table> </form> ";
54 } else {
55     echo "Ningun resultado";
56 }
57 mysqli_close($conexion);
58 ?>
59 <form action="index.php" method="POST">
60 <input type="submit" name="index" value="Volver al inicio">
61 </form>
62 </div>
63 </body>
64 </html>
65
```

## rrhh.php

```
1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 <style>
5 table {
6     border-collapse: collapse;
7     border: 1px solid black;
8 }
9 td {
10     background-color: #FFFEE3
11     border-collapse: collapse;
12     border: 1px solid black;
13     align:center;
14     width:5%;
15 }
16 tr {
17     background-color: #FFFEE3
18     border-collapse: collapse;
19     border: 1px solid black;
20 }
```

```

21     .a {
22         background-color: #EDE61C ;
23     }
24 </style>
25 </head>
26 <body bgcolor= "lightblue">
27 <br><br><br><br><div style ="background-color: #11FFCD; margin-left: 1%; border: 2px solid black; padding: 1%;">
28 <?PHP
29 $hostname_db = "localhost";
30 $database_db = "usuarios";
31 $username_db = "ja";
32 $password_db = "rootroot";
33 $conexion = mysqli_connect($hostname_db, $username_db, $password_db, $database_db);
34 <?if (!$conexion) {
35     die ("La conexion ha fallado:" . mysqli_connect_error());
36 }
37 $pregunta = "select * from historial_r";
38 $resultado = mysqli_query($conexion,$pregunta);
39 <?if (mysqli_num_rows($resultado) > 0 ) {
40     echo "<form ACTION='imasd.php' METHOD='POST'><table background='im.jpg'>";
41     echo " <tr align=center class='a' >
42         <td class='a'>Numero de apertura de la puerta</td>
43         <td class='a'>N_de_usuario</td>
44         <td class='a'>Fecha y hora de entrada</td>
45     </tr>";
46     <?while ($row = mysqli_fetch_assoc($resultado)) {
47         echo "<tr >
48             <td align=center>$row[id]</td>
49             <td align=center>$row[numero_de_usuario]</td>
50             <td align=center>$row[fecha_hora]</td>
51         </tr>";
52     }
53     echo "</table> </form> ";
54 } else {
55     echo "Ningun resultado";
56 }
57 mysqli_close($conexion);
58 <?>
59 <form action="index.php" method="POST">
60 <input type="submit" name="index" value="Volver al inicio">
61 </form>
62 </div>
63 </body>
64 </html>

```

## imasd.php

```

1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 <style>
5 table {
6     border-collapse: collapse;
7     border: 1px solid black;
8 }
9 td {
10     background-color: #FFEE3;
11     border-collapse: collapse;
12     border: 1px solid black;
13     align:center;
14     width:5%;
15 }
16 tr {
17     background-color: #FFEE3;
18     border-collapse: collapse;
19     border: 1px solid black;
20 }
21 .a {
22     background-color: #EDE61C ;
23 }
24 </style>
25 </head>
26 <body bgcolor= "lightblue">
27 <br><br><br><br><div style ="background-color: #11FFCD; margin-left: 1%; border: 2px solid black; padding: 1%;">
28 <?PHP
29 $hostname_db = "localhost";
30 $database_db = "usuarios";
31 $username_db = "ja";
32 $password_db = "rootroot";
33 $conexion = mysqli_connect($hostname_db, $username_db, $password_db, $database_db);
34 <?if (!$conexion) {
35     die ("La conexion ha fallado:" . mysqli_connect_error());
36 }
37 $pregunta = "select * from historial_i";
38 $resultado = mysqli_query($conexion,$pregunta);
39 <?if (mysqli_num_rows($resultado) > 0 ) {
40     echo "<form ACTION='imasd.php' METHOD='POST'><table background='im.jpg'>";
41     echo " <tr align=center class='a' >
42         <td class='a'>Numero de apertura de la puerta</td>
43         <td class='a'>N_de_usuario</td>

```



```

44         <td class='a'>Fecha y hora de entrada</td>
45     </tr>";
46     while ($row = mysqli_fetch_assoc($resultado)) {
47         echo "<tr >
48             <td align=center>$row[id]</td>
49             <td align=center>$row[numero_de_usuario]</td>
50             <td align=center>$row[fecha_hora]</td>
51         </tr>";
52     }
53     echo "</table> </form> ";
54 } else {
55     echo "Ningun resultado";
56 }
57 mysqli_close($conexion);
58 ?>
59 <form action="index.php" method="POST">
60 <INPUT TYPE="submit" NAME="index" VALUE="Volver al inicio">
61 </form>
62 </div>
63 </body>
64 </html>

```

## sala.php

```

1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 <style>
5 table {
6     border-collapse: collapse;
7     border: 1px solid black;
8 }
9 td {
10     background-color: #FFFE3;
11     border-collapse: collapse;
12     border: 1px solid black;
13     align:center;
14     width:5%;
15 }
16 tr {
17     background-color: #FFFE3;
18     border-collapse: collapse;
19     border: 1px solid black;
20 }
21 .a {
22     background-color: #EDE61C ;
23 }
24 </style>
25 </head>
26 <body bgcolor= "lightblue">
27 <br><br><br><br><div style = "background-color: #11FFCD; margin-left: 1%; border: 2px solid black; padding: 1%;">
28 <?PHP
29 $hostname_db = "localhost";
30 $database_db = "usuarios";
31 $username_db = "ja";
32 $password_db = "rootroot";
33 $conexion = mysqli_connect($hostname_db, $username_db, $password_db, $database_db);
34 if (!$conexion) {
35     die ("La conexion ha fallado:" . mysqli_connect_error());
36 }
37 $pregunta = "select * from historial_s";
38 $resultado = mysqli_query($conexion,$pregunta);
39 if (mysqli_num_rows($resultado) > 0) {
40     echo "<form ACTION='imasd.php' METHOD='POST'><table background='im.jpg'>";
41     echo " <tr align=center class='a' >
42         <td class='a'>Numero de apertura de la puerta</td>
43         <td class='a'>N_de_usuario</td>
44         <td class='a'>Fecha y hora de entrada</td>
45     </tr>";
46     while ($row = mysqli_fetch_assoc($resultado)) {
47         echo "<tr >
48             <td align=center>$row[id]</td>
49             <td align=center>$row[numero_de_usuario]</td>
50             <td align=center>$row[fecha_hora]</td>
51         </tr>";
52     }
53     echo "</table> </form> ";
54 } else {
55     echo "Ningun resultado";
56 }
57 mysqli_close($conexion);
58 ?>
59 <form action="index.php" method="POST">
60 <INPUT TYPE="submit" NAME="index" VALUE="Volver al inicio">
61 </form>
62 </div>
63 </body>
64 </html>

```

## denegados.php

```
1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
4 <style>
5 table {
6     border-collapse: collapse;
7     border: 1px solid black;
8 }
9 td {
10     background-color: #FFFEE3
11     border-collapse: collapse;
12     border: 1px solid black;
13     align:center;
14     width:5%;
15 }
16 tr {
17     background-color: #FFFEE3
18     border-collapse: collapse;
19     border: 1px solid black;
20 }
21 .a {
22     background-color: #EDE61C ;
23 }
24 </style>
25 </head>
26 <body bgcolor= "lightblue">
27 <br><br><br><br><div style ="background-color: #11FFCD; margin-left: 1%; border: 2px solid black; padding: 1%;">
28 <?PHP
29 $hostname_db = "localhost";
30 $database_db = "usuarios";
31 $username_db = "ja";
32 $password_db = "rootroot";
33 $conexion = mysqli_connect($hostname_db, $username_db, $password_db, $database_db);
34 <?if (!$conexion) {
35     die ("La conexion ha fallado:" . mysqli_connect_error());
36 }
37 $pregunta = "select * from historial_x";
38 $resultado = mysqli_query($conexion,$pregunta);
39 <?if (mysqli_num_rows($resultado) > 0 ) {
40     echo "<form ACTION='denegados.php' METHOD='POST'><table background='im.jpg'>";
41     echo " <tr align=center class='a' >
42         <td class='a'>Numero de apertura de la puerta</td>
43         <td class='a'>N_de_usuario</td>
44         <td class='a'>Fecha y hora de entrada</td>
45     </tr>";
46     while ($row = mysqli_fetch_assoc($resultado)) {
47         echo "<tr >
48             <td align=center>$row[id]</td>
49             <td align=center>$row[numero_de_usuario]</td>
50             <td align=center>$row[fecha_hora]</td>
51         </tr>" ;
52     }
53     echo "</table> </form> ";
54 } else {
55     echo "Ningun resultado";
56 }
57 <?mysqli_close($conexion);
58 <?
59 <?form action="index.php" method="POST">
60 <?<input type="submit" name="index" value="Volver al inicio">
61 <?</form>
62 <?</div>
63 <?</body>
64 <?</html>
```

## ANEXO II - Lector de ID

El objetivo del lectorid.py es detectar los nuevos IDs que introducir a la base de datos “usuarios” y gracias a la configuración:

```
sudo nano /etc/xdg/lxsession/LXDE-pi/autostart^C
```

en el archivo:

```
@/usr/bin/python3 /home/pi/Desktop/lectorid.py
```

se iniciara el programa automáticamente. Para ello, lo primero que haremos es importar la librería “GPIO” para la interpretación de los pines de la RaspBerry, la librería “SimpleMFRC522” para el lector RFID, la librería “sleep” para establecer parámetros de tiempo y la librería “os” para poder trabajar con archivos del sistema:

```
import RPi.GPIO as GPIO
from mfrc522 import SimpleMFRC522
import time
import os
from time import sleep
```

Ponemos el modo de interpretación de pines de la RaspBerry en “BCM”, ya que por defecto nos aparece en “BOARD”:

```
GPIO.setmode(GPIO.BCM)
```

Definimos variables para los distintos colores de led y para el visualizador de siete segmentos, además de un array con los números interpretados del 0 al 9 y una variable que llame a la función del RFID dentro de su librería... Además, por definición dejaremos la luz del led rojo encendida y la del verde apagada.

```
GPIO.setwarnings(False)
rojo=26
verde=16
SDI = 17
RCLK = 18
SRCLK = 27
segCode = [0x6f,0x7f,0x07,0x7d,0x6d,0x66,0x4f,0x5b,0x06,0x3f]
GPIO.setup(verde, GPIO.OUT)
GPIO.output(verde, False)
GPIO.setup(rojo, GPIO.OUT)
GPIO.output(rojo, True)
GPIO.setup(SDI, GPIO.OUT, initial=GPIO.LOW)
GPIO.setup(RCLK, GPIO.OUT, initial=GPIO.LOW)
GPIO.setup(SRCLK, GPIO.OUT, initial=GPIO.LOW)
reader = SimpleMFRC522()
```

Creamos una función que recorra el array de números, los cuales previamente hemos puesto en orden descendente.

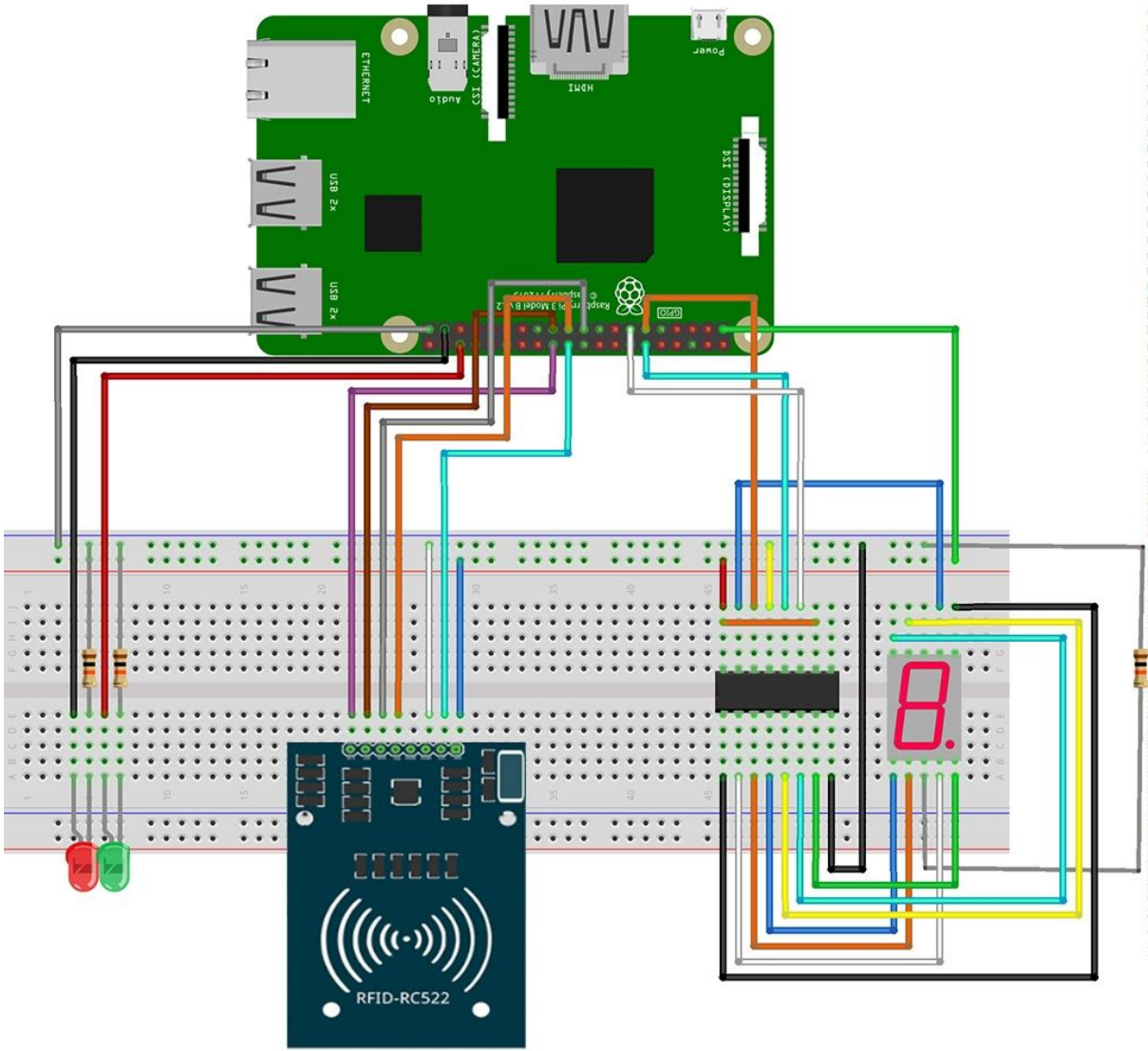
```
def cuentaAtras(dat):  
    for bit in range(0, 8):  
        GPIO.output(SDI, 0x80 & (dat << bit))  
        GPIO.output(SRCLK, GPIO.HIGH)  
        time.sleep(0.001)  
        GPIO.output(SRCLK, GPIO.LOW)  
        GPIO.output(RCLK, GPIO.HIGH)  
        time.sleep(0.001)  
        GPIO.output(RCLK, GPIO.LOW)
```

Creamos un bucle infinito en el cual se le pide al RFID que busque emisores de frecuencia de manera constante. Cuando este detecta un número mayor o igual a uno (lo que significa que ha detectado una llave RFID, pues estas tienen secuencias de números de más de diez cifras) recoge el ID de esta, lo escribe en un documento tipo “.txt” y llama a la función que inicializa la cuenta atrás mostrándonos la en el visualizador de siete segmentos. Una vez esta cuenta atrás termine, el programa abrirá de nuevo el archivo tipo “.txt” y lo reescribirá dejándolo en blanco.

```
try:  
    while True:  
        id, text = reader.read()  
        if(id>=1):  
            f = open ('/home/pi/Desktop/lecturaaid.txt', 'w')  
            f.write(str(id))  
            f.close()  
            GPIO.output(rojo, False)  
            GPIO.output(verde, True)  
            for code in segCode:  
                cuentaAtras(code)  
                time.sleep(2)  
            GPIO.output(rojo, True)  
            GPIO.output(verde, False)  
            f = open ('/home/pi/Desktop/lecturaaid.txt', 'w')  
            f.write(" ")  
            f.close()  
            os.system('clear')  
finally:  
    GPIO.cleanup()
```

A continuación se presenta una foto de como queda el programa y su ensamblado:

```
1 import RPi.GPIO as GPIO
2 from mfr522 import SimpleMFRC522
3 import time
4 import os
5 from time import sleep
6 GPIO.setmode(GPIO.BCM)
7 GPIO.setwarnings(False)
8 rojo=26
9 verde=16
10 SDI = 17
11 RCLK = 18
12 SRCLK = 27
13 segCode = [0x6f,0x7f,0x07,0x7d,0x6d,0x66,0x4f,0x5b,0x06,0x3f]
14 GPIO.setup(verde, GPIO.OUT)
15 GPIO.output(verde, False)
16 GPIO.setup(rojo, GPIO.OUT)
17 GPIO.output(rojo, True)
18 GPIO.setup(SDI, GPIO.OUT, initial=GPIO.LOW)
19 GPIO.setup(RCLK, GPIO.OUT, initial=GPIO.LOW)
20 GPIO.setup(SRCLK, GPIO.OUT, initial=GPIO.LOW)
21 reader = SimpleMFRC522()
22 def cuentaAtras(dat):
23     for bit in range(0, 8):
24         GPIO.output(SDI, 0x80 & (dat << bit))
25         GPIO.output(SRCLK, GPIO.HIGH)
26         time.sleep(0.001)
27         GPIO.output(SRCLK, GPIO.LOW)
28     GPIO.output(RCLK, GPIO.HIGH)
29     time.sleep(0.001)
30     GPIO.output(RCLK, GPIO.LOW)
31 try:
32     while True:
33         id, text = reader.read()
34         if(id>=1):
35             f = open ('/home/pi/Desktop/lecturaid.txt', 'w')
36             f.write(str(id))
37             f.close()
38             GPIO.output(rojo, False)
39             GPIO.output(verde, True)
40             for code in segCode:
41                 cuentaAtras(code)
42                 time.sleep(2)
43             GPIO.output(rojo, True)
44             GPIO.output(verde, False)
45             f = open ('/home/pi/Desktop/lecturaid.txt', 'w')
46             f.write(" ")
47             f.close()
48             os.system('clear')
49 finally:
50     GPIO.cleanup()
```



## ANEXO III - Biometría

Antes de profundizar en el código, debemos asegurarnos de que disponemos de una Webcam, ya que usaremos la identificación facial a través de ella. El objetivo es crear un control de acceso por el cual una cámara nos detectará la cara, y si estamos guardados en su base de datos, nos reconozca y nos deje pasar, mientras que si no lo estamos, nos detecte como desconocido y nos deniegue el acceso. Nuestro programa, constará de 3 subprogramas python que crearemos desde la consola con el siguiente comando:

```
touch registro.py guardado.py identificar.py
```

y posteriormente editaremos escribiendo el comando **nano** seguido del programa en cuestión.

El primero de ellos y sin el que el programa no tendría sentido, será el que nos registre en la base de datos para que cuando pasemos por delante de la cámara, nos reconozca.

Lo primero que haremos para ello será instalar el openCV (Open Computer Vision), una biblioteca de uso libre y que es apta para prácticamente cualquier sistema operativo tales como GNU/Linux, Mac OS X, Windows, Android y en nuestro caso, Raspberry Pi.

El primer paso es asegurarnos de que no tenemos instalada una versión desactualizada. Además, hay cuatro posibles paquetes para instalar, y solo debemos tener uno de ellos para evitar posibles problemas. Para asegurarnos de que todo esté correctamente, en nuestro terminal, haremos un:

```
pip uninstall
```

de todos ellos e instalaremos únicamente el deseado. A continuación, ejecutaremos el comando:

```
pip install opencv-contrib-python
```

el cual nos servirá para los módulos principales y contrib, pero también es válida una versión más sencilla:

```
pip install opencv-python-headless
```

Por último, les dejamos un script que tarda bastante en ejecutarse, sobretodo si no tenemos nuestro sistema actualizado, pero permitirá al opencv funcionar correctamente en versiones mas nuevas de python como python3.

Para ello, lo primero que haremos será crear un archivo **.sh** desde la consola:

```
touch opencvfinal.sh
```

A seguido lo abriremos:

```
nano opencvfinal.sh
```

y copiaremos el siguiente texto:

```
OPENCV_VERSION='4.2.0'
OPENCV_CONTRIB='NO'
sudo apt-get -y update
sudo apt-get install -y build-essential cmake
sudo apt-get install -y qt5-default libvtk6-dev
sudo apt-get install -y zlib1g-dev libjpeg-dev libwebp-dev libpng-dev libtiff5-dev libjasper-dev \
libopenexr-dev libgdal-dev
sudo apt-get install -y libdc1394-22-dev libavcodec-dev libavformat-dev libswscale-dev \
libtheora-dev libvorbis-dev libxvidcore-dev libx264-dev yasm \
libopencore-amrnb-dev libopencore-amrwb-dev libv4l-dev libxine2-dev
sudo apt-get install -y libtbb-dev libeigen3-dev
sudo apt-get install -y python-dev python-tk pylint python-numpy \
python3-dev python3-tk pylint3 python3-numpy flake8
sudo apt-get install -y ant default-jdk
sudo apt-get install -y doxygen unzip wget
wget https://github.com/opencv/opencv/archive/${OPENCV_VERSION}.zip
unzip ${OPENCV_VERSION}.zip && rm ${OPENCV_VERSION}.zip
mv opencv-${OPENCV_VERSION} OpenCV
if [ $OPENCV_CONTRIB = 'YES' ]; then
    wget https://github.com/opencv/opencv_contrib/archive/${OPENCV_VERSION}.zip
    unzip ${OPENCV_VERSION}.zip && rm ${OPENCV_VERSION}.zip
    mv opencv_contrib-${OPENCV_VERSION} opencv_contrib
    mv opencv_contrib OpenCV
fi
cd OpenCV && mkdir build && cd build
if [ $OPENCV_CONTRIB = 'NO' ]; then
    cmake -DWITH_QT=ON -DWITH_OPENGL=ON -DFORCE_VTK=ON -DWITH_TBB=ON -DWITH_GDAL=ON \
-DWITH_XINE=ON -DENABLE_PRECOMPILED_HEADERS=OFF ..
fi
if [ $OPENCV_CONTRIB = 'YES' ]; then
    cmake -DWITH_QT=ON -DWITH_OPENGL=ON -DFORCE_VTK=ON -DWITH_TBB=ON -DWITH_GDAL=ON \
-DWITH_XINE=ON -DENABLE_PRECOMPILED_HEADERS=OFF \
-DOPENCV_EXTRA_MODULES_PATH=./opencv_contrib/modules ..
fi
make -j8
sudo make install
sudo ldconfig
```

Para guardar, haremos “ctrl-o” y para cerrar “ctrl-x”. Finalmente, ejecutamos el programa con el siguiente comando:

```
bash opencvfinal.sh
```



## Registro.py

Una vez completados los requisitos anteriores, podemos empezar a editar nuestro primer programa python añadiendo la librería de opencv con la línea:

```
import cv2
```

además de la librería:

```
import os
```

A continuación iniciamos la interacción con el usuario; las siguientes líneas, serán para que el usuario decida a que persona desea registrar, además de un pequeño menú que le permitirá cambiarle el nombre a la persona si ya hay uno que posea ese mismo nombre, o abandonar el programa en caso de cambiar de idea.

```
print("¿A quien desea introducir?\n")
nombre = input()
continuar = "2"
while
os.path.isdir('/home/pi/Downloads/face_recognitionOpenCv2-master/images/'+n
ombre):
    print("Ese nombre ya existe, elija el numero de una de las siguientes
opciones:")
    print("=====")
    print("|1. Escribir un nombre distinto.      |")
    print("|2. Continuar con este mismo nombre. |")
    print("|3. Abandonar el registro.          |")
    print("=====")
    continuar = input()
    if continuar != "1":
        break
    else:
        nombre = input("Escriba el nuevo nombre:\n")
if continuar == "2":
os.mkdir("/home/pi/Downloads/face_recognitionOpenCv2-master/images/"+nombre
)
```

Finalmente el programa nos crea una carpeta en el directorio de imágenes del proyecto, en el que se guardaran las fotos que la cámara nos tome para compararnos en un futuro.

Con el openCV ya incluido en nuestro programa, lo que crearemos será una variable que nos permita abrir la camara de nuestra máquina :

```
camara = cv2.VideoCapture(0)
```

De normal, la capturadora de video va a llevar el ID 0, a no ser que estemos utilizando una cámara externa, en cuyo caso puede variar. La siguiente línea es la ruta del archivo que nos

ayuda a detectar los rostros. En nuestro caso, utilizamos una ruta relativa a este archivo, pues lo tenemos guardado en una carpeta junto al programa python en uso, pero perfectamente se puede usar una ruta absoluta al mismo:

```
rutaCascade = "Cascades/haarcascade_frontalface_default.xml"
```

La siguiente variable que crearemos será para llamar al clasificador de la biblioteca cv2, al cual le enviaremos la ruta del archivo que vamos a utilizar para identificar las caras:

```
caraCascade = cv2.CascadeClassifier(rutaCascade)
```

Por último, crearemos una variable contador que nos indique el número de imágenes que se van a crear en la carpeta del set de datos.

```
contador = 0
```

Una vez creadas estas variables, podemos pasar al programa en sí. Lo primero que haremos será crear un bucle infinito usando el:

```
while (True):
```

que más tarde detendremos según los resultados obtenidos. A continuación definiremos una variable para la ventana que nos abrirá la cámara y la pondrá en funcionamiento.

```
_, borde = camara.read()
```

Ahora haremos que, pese a que nosotros nos veamos a color en la cámara, las fotos que nos tome sean en una escala de grises ya que posteriormente necesitaremos que así sea.

```
gris = cv2.cvtColor(borde, cv2.COLOR_BGR2GRAY)
```

Por último abriremos el clasificador de openCV `caraCascade` y el método `detectMultiScale` en una nueva variable `cara`. Este método solo detecta imágenes en la escala de grises, y es por eso por lo que definimos previamente nuestra variable: `gris`

```
cara = caraCascade.detectMultiScale(gris, 1.5, 5)
```

A continuación, creamos un `for` con el recuadro que nuestro rostro va a ocupar y la propia variable rostro, y sumamos 1 al contador `contador`:

```
for(x,y,w,h) in cara:
```

```
    cv2.rectangle(borde, (x,y), (x+w, y+h), (255,0,0), 4)
```

```
    contador = contador + 1
```

Ahora abriremos nuestra carpeta de imágenes y dentro guardaremos la recién tomada, con un nombre predefinido (en nuestro caso "A") añadiéndole al nombre la variable count, haciendo que nuestras imagenes no se sobrescriban y permitiéndonos así llevar la cuenta

de las que llevamos y finalmente poniendo el formato de imagen “.jpg”.

```
cv2.imwrite("images/"+nombre+"/"+nombre+""+str(contador)+".jpg",
gris[y:y+h, x:x+w])
cv2.imshow("Creando Dataset", borde)
```

Una vez hecho el `for`, añadiremos dos condiciones al bucle `while` que habíamos abierto. La primera de ellas será que si pulsamos la letra `c`, la cámara deje de tomarnos fotos de la cara, y se quede solamente con los registros que ya lleve. La otra opción será para que si alcanzamos el número de 400 fotos, se detenga automáticamente, pues con 400 fotos debería de ser suficiente para detectarlos el rostro.

```
if cv2.waitKey(1) & 0xFF == ord('c'):
    break
elif contador >= 400:
    break
```

Finalmente cerramos todas las ventanas de la cámara y finalizamos el programa.

```
camara.release()
cv2.destroyAllWindows()
print('Programa finalizado.')
```

A continuación les dejamos una foto de como debería de quedar el archivo python:

```
import cv2
import os

print("¿A quien desea introducir?\n")
nombre = input()
continuar = "2"
while os.path.isdir('/home/pi/Downloads/face_recognitionOpenCv2-master/images/'+nombre):
    print("Ese nombre ya existe, elija el numero de una de las siguientes opciones:")
    print("=====")
    print("[1. Escribir un nombre distinto. |")
    print("[2. Continuar con este mismo nombre. |")
    print("[3. Abandonar el registro. |")
    print("=====")
    continuar = input()
    if continuar != "1":
        break
    else:
        nombre = input("Escriba el nuevo nombre:\n")
if continuar == "2":
    os.mkdir("/home/pi/Downloads/face_recognitionOpenCv2-master/images/"+nombre)
    camara = cv2.VideoCapture(0)

    rutaCascade = "Cascades/haarcascade_frontalface_default.xml"
    caraCascade = cv2.CascadeClassifier(rutaCascade)

    contador = 0
    while(True):
        _, borde = camara.read()

        gris = cv2.cvtColor(borde, cv2.COLOR_BGR2GRAY)

        cara = caraCascade.detectMultiScale(gris, 1.5, 5)

        for(x,y,w,h) in cara:
            cv2.rectangle(borde, (x,y), (x+w, y+h), (255,0,0), 4)
            contador = contador + 1

            cv2.imwrite("images/"+nombre+"/"+nombre+""+str(contador)+".jpg", gris[y:y+h, x:x+w])
            cv2.imshow("Creando Dataset", borde)

            if cv2.waitKey(1) & 0xFF == ord('c'):
                break

            elif contador >= 400:
                break

        camara.release()
        cv2.destroyAllWindows()
    print('Programa finalizado.')
```

## Guardado.py

Una vez completado este programa, pasamos al segundo, el cual usa otras bibliotecas de python a parte de la de openCV, pero esta vez no nos detendremos a explicarlas tan detalladamente.

```
import cv2
import pickle
import os
import numpy as np
from PIL import Image
```

A continuación volvemos a crear las variables `rutaCascade` y `caraCascade` como hicimos en el programa anterior (el reconocedor de rostros y el clasificador).

```
rutaCascada = "Cascades/haarcascade_frontalface_alt2.xml"
caraCascade = cv2.CascadeClassifier(rutaCascada)
```

A continuación, utilizando la biblioteca de cv2, crearemos una variable que utiliza un método de reconocimiento automático de rostros.

```
reconocer = cv2.face.LBPHFaceRecognizer_create()
```

A seguido, definimos las rutas que el programa debe de utilizar para definir las etiquetas, además de las variables de las mismas:

```
dir_base = os.path.dirname(os.path.abspath(__file__))
dir_fotos = os.path.join(dir_base, "images")
etiquetasY = []
entrenamientoX = []
id_etiquetas = {}
id_contador = 0
```

Estas etiquetas serán las que aparezcan sobre nuestra cabeza cuando nos detecte el programa, ya sea porque nos ha reconocido y ponga nuestro nombre, o en su defecto, porque no nos ha reconocido y ponga la etiqueta de "Desconocido". Para que el programa nos detecte, deberemos abrir los archivos ".png" y ".jpg" que tengamos guardados en nuestra carpeta y con los cuales nos comparará en tiempo real. Estos los abriremos en un tamaño determinado que nosotros mismos definiremos. Para ello, utilizaremos el siguiente código:

```
for root, dirs, archivos in os.walk(dir_fotos):
    for archivo in archivos:
        if archivo.endswith("png") or archivo.endswith("jpg"):
            rutaFoto = os.path.join(root, archivo)
            etiqueta = os.path.basename(root).replace(" ", "-").lower()
```

```

#Para crear las etiquetas
if not etiqueta in id_etiquetas:
    id_etiquetas[etiqueta] = id_contador
    id_contador = id_contador + 1
id_etiquetas_entero = id_etiquetas[etiqueta]
p_foto = Image.open(rutaFoto).convert("L")
tam = (550,550)
fotoFinal = p_foto.resize(tam, Image.ANTIALIAS)
array_foto = np.array(p_foto,"uint8")
caras = caraCascade.detectMultiScale(array_foto, 1.5, 5)
for (x,y,w,h) in caras:
    roi = array_foto[y:y+h, x:x+w]
    entrenamientoX.append(roi)
    etiquetasY.append(id_etiquetas_entero)

```

Por último abrimos el label.pickle para guardar cada etiqueta con su id y un archivo .yml que nos haga de base de datos donde almacenar toda la información.

```

with open("labels.pickle",'wb') as f:
    pickle.dump(id_etiquetas, f)
reconocer.train(entrenamientoX, np.array(etiquetasY))
reconocer.save("entrenamiento.yml")

```

Cabe recordar que cada vez que actualicemos a una persona en el primer programa de python, ejecutemos este mismo programa para que nos actualice la base de datos. A continuación les dejamos una foto de como debería de quedar el archivo python:

```

import cv2
import pickle
import os
import numpy as np
from PIL import Image

#variables
rutaCascada = "Cascades/haarcascade_frontalface_alt2.xml"
caraCascade = cv2.CascadeClassifier(rutaCascada)
reconocer = cv2.face.LBPHFaceRecognizer_create()#reconocer con cv2
dir_base = os.path.dirname(os.path.abspath(__file__))
dir_fotos = os.path.join(dir_base,"images")
etiquetasY = []
entrenamientoX = []
id_etiquetas = {}
id_contador = 0

for root, dirs, archivos in os.walk(dir_fotos):

    for archivo in archivos:

        if archivo.endswith("png") or archivo.endswith("jpg"):
            rutaFoto = os.path.join(root,archivo)
            etiqueta = os.path.basename(root).replace(" ", "-").lower()

            #Para crear las etiquetas
            if not etiqueta in id_etiquetas:

                id_etiquetas[etiqueta] = id_contador
                id_contador = id_contador + 1

            id_etiquetas_entero = id_etiquetas[etiqueta]
            p_foto = Image.open(rutaFoto).convert("L")
            tam = (550,550)
            fotoFinal = p_foto.resize(tam, Image.ANTIALIAS)
            array_foto = np.array(p_foto,"uint8")
            caras = caraCascade.detectMultiScale(array_foto, 1.5, 5)

            for (x,y,w,h) in caras:

                roi = array_foto[y:y+h, x:x+w]
                entrenamientoX.append(roi)
                etiquetasY.append(id_etiquetas_entero)

with open("labels.pickle",'wb') as f:
    pickle.dump(id_etiquetas, f)

reconocer.train(entrenamientoX, np.array(etiquetasY))
reconocer.save("entrenamiento.yml")

```

## Identificar.py

Finalmente, pasamos a nuestro tercer programa, el cual se encargará de detectar distintas partes de nuestro rostro para así encontrarnos en la base de datos si es que estamos. Una vez más, importaremos las librerías de `cv2` y `pickle` y crearemos las variables `rutaCascade` y `caraCascade`:

```

import cv2
import pickle
cascPath = "Cascades/haarcascade_frontalface_alt2.xml"
faceCascade = cv2.CascadeClassifier(cascPath)

```

Para mayor precisión en nuestro programa, incluiremos un detector de ojos y uno de la boca, lo cual nos proporciona de nuevo la biblioteca de openCV:

```

ojosCascade = cv2.CascadeClassifier("Cascades/haarcascade_eye.xml")
bocaCascade = cv2.CascadeClassifier("Cascades/haarcascade_smile.xml")

```

De nuevo, introducimos el método de reconocimiento facial automático como ya hicimos en nuestro primer programa:

```

reconocer = cv2.face.LBPHFaceRecognizer_create()

```

```
reconocer.read("entrenamiento.yml")
```

a continuación inicializamos las etiquetas, y abrimos el archivo label.pickle en modo de lectura donde cargaremos esas etiquetas:

```
etiquetas_a = {"name_persona" : 1 }  
with open("labels.pickle", 'rb') as f:  
    etiquetas_b = pickle.load(f)  
    etiquetas_a = { v:k for k,v in etiquetas_b.items() }
```

Ahora abriremos la cámara para hacer la comparación entre la persona que esté situada frente a la cámara, y las imágenes que tenemos guardadas.

```
camara = cv2.VideoCapture(0)
```

Ahora creamos una variable llamada similitudes que más adelante explicaremos su utilidad.

```
similitudes = 0
```

A continuación creamos un bucle:

```
while True:
```

y como hacíamos en el primer programa, tendremos que abrir la cámara, que nos cree el marco alrededor de la cara y que en segundo plano, nos la reproduzca en una escala de grises para así poderla comparar con las imágenes ya guardadas.

```
# Para capturar el recuadro  
ret, recuadro = camara.read()  
gris = cv2.cvtColor(recuadro, cv2.COLOR_BGR2GRAY)  
caras = caraCascade.detectMultiScale(gris, 1.5, 5)  
for (x, y, w, h) in caras:  
    roi_gris = gris[y:y+h, x:x+w]  
    roi_color = recuadro[y:y+h, x:x+w]
```

También haremos que si encuentra una coincidencia de la cara entre unos parámetros, que nos dé un resultado. Lo primero es que las coincidencias sean estén por encima de un mínimo, ya que lo que nos interesa es que al menos detecte que es una cara, y no que simplemente por ver una mesa nos de "Desconocido". Además, si la imagen no tiene un mínimo de 50% de coincidencia, aparecerá el id "Desconocido", mientras que si es superior, nos pondrá el id de la persona correspondiente. Acompañándolo, incluimos algunas líneas para que en la interfaz gráfica sea más amena, como por ejemplo el grosor del rectángulo que nos detecta o lo que nos ocupa en la pantalla. Como se puede apreciar en el código, hemos añadido un lanzador a una página web (en nuestro caso local, pero que podría ser cualquier otra). Este lanzador se activa cuando la cámara identifica a un usuario registrado previamente. Como medida preventiva, hemos requerido que el usuario tenga que ser

detectado durante un breve espacio de tiempo (gracias a la variable `similitudes` que funciona como contador dentro del bucle `while`), de tal manera que si la cámara no detecta a nadie, o detecta a alguien desconocido, no ejecute el lanzador.

```
id_, simil_conf = reconocer.predict(roi_gris)
    if simil_conf >= 4 and simil_conf < 85:
        font = cv2.FONT_HERSHEY_SIMPLEX
        name = etiquetas_a[id_]
        if simil_conf > 50:
            name = "Desconocido"
        if name != "Desconocido" and name != "":
            similitudes += 1
            sleep(0.1)
            if similitudes > 4:
                webbrowser.open_new("http://192.168.1.44/pagina/registro_entrar.html")
                similitudes = 0
            color = (255,255,255)
            ancho = 2
            cv2.putText(recuadro, name, (x,y), font, 1, color, ancho, cv2.LINE_AA)
            img_item = "my-image.png"
            cv2.imwrite(img_item, roi_gris)
            cv2.rectangle(recuadro, (x, y), (x+w, y+h), (0, 255, 0), 2)
            caract = bocaCascade.detectMultiScale(roi_gris)
            for (ex,ey,ew,eh) in caract:
                cv2.rectangle(roi_color, (ex, ey), (ex+ew, ey+eh), (0, 255, 0), 2)
            reprod_recuadro = cv2.resize(recuadro, (1200, 650), interpolation = cv2.INTER_CUBIC)
            cv2.imshow('Detectando caras', reprod_recuadro)
```

Finalmente, establecemos que al pulsar la letra “c” se finalice el bucle infinito, de forma que a continuación se cierren tanto el programa, como las ventanas que este nos había abierto.

```
    if cv2.waitKey(1) & 0xFF == ord('c'):
        break
camara.release()
cv2.destroyAllWindows()
```

A continuación les dejamos una foto de como debería de quedar el archivo python:



```

import webbrowser
import time
from time import sleep
import cv2
import pickle

rutaCascade = "Cascades/haarcascade_frontalface_alt2.xml"
caraCascade = cv2.CascadeClassifier(rutaCascade)
ojosCascade = cv2.CascadeClassifier("Cascades/haarcascade_eye.xml")
bocaCascade = cv2.CascadeClassifier("Cascades/haarcascade_smile.xml")
reconocer = cv2.face.LBPHFaceRecognizer_create()
reconocer.read("entrenamiento.yml")
etiquetas_a = {"name_persona" : 1 }

with open("labels.pickle",'rb') as f:
    etiquetas_b = pickle.load(f)
    etiquetas_a = { v:k for k,v in etiquetas_b.items()}

camara = cv2.VideoCapture(0)

similitudes = 0

while True:
    # Para capturar el recuadro
    ret, recuadro = camara.read()
    gris = cv2.cvtColor(recuadro, cv2.COLOR_BGR2GRAY)
    caras = caraCascade.detectMultiScale(gris, 1.5, 5)

    for (x, y, w, h) in caras:
        roi_gris = gris[y:y+h, x:x+w]
        roi_color = recuadro[y:y+h, x:x+w]
        id_, simil_conf = reconocer.predict(roi_gris)

        if simil_conf >= 4 and simil_conf < 85:
            font = cv2.FONT_HERSHEY_SIMPLEX
            name = etiquetas_a[id_]

            if simil_conf > 50:
                name = "Desconocido"
            if name != "Desconocido" and name != "":
                similitudes += 1
                sleep(0.1)
                if similitudes > 4:
                    webbrowser.open_new("http://192.168.1.44/pagina/registro_entrar.html")
                    similitudes = 0
                color = (255,255,255)
                ancho = 2
                cv2.putText(recuadro, name, (x,y), font, 1, color, ancho, cv2.LINE_AA)

            img_item = "my-image.png"
            cv2.imwrite(img_item, roi_gris)
            cv2.rectangle(recuadro, (x, y), (x+w, y+h), (0, 255, 0), 2)
            caract = bocaCascade.detectMultiScale(roi_gris)

            for(ex,ey,ew,eh) in caract:
                cv2.rectangle(roi_color, (ex, ey), (ex+ew, ey+eh), (0, 255, 0), 2)

        reprod_recuadro = cv2.resize(recuadro, (1200, 650), interpolation = cv2.INTER_CUBIC)
        cv2.imshow('Detectando caras', reprod_recuadro)

        if cv2.waitKey(1) & 0xFF == ord('c'):
            break

camara.release()
cv2.destroyAllWindows()

```

## ANEXO IV - Contraseña numérica

Para poder implementar una contraseña hemos utilizado un par de botones y un visualizador de siete segmentos. Importamos en nuestro programa python las siguientes librerías:

```

import RPi.GPIO as GPIO
import time

```

y, en nuestro caso, establecemos el modo de lectura de pines de la RaspBerry en BCM.

```
GPIO.setmode(GPIO.BCM)
```

Definimos las variables de los botones además del SDI RCLK y SRCLK del visualizador y los disponemos en modo salida.

```
BtnA=12
BtnB=16
SDI = 6
RCLK = 26
SRCLK = 13
GPIO.setup(BtnA, GPIO.OUT)
GPIO.setup(BtnB, GPIO.OUT)
GPIO.setup(SDI, GPIO.OUT, initial=GPIO.LOW)
GPIO.setup(RCLK, GPIO.OUT, initial=GPIO.LOW)
GPIO.setup(SRCLK, GPIO.OUT, initial=GPIO.LOW)
```

Definimos en un array los números del 0 al 9 para el modelo de lectura del visualizador.

```
segCode = [0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f,0x80]
```

Ahora definimos una función a la que llamaremos más tarde. Esta función se encarga de recorrer el array de números que definimos previamente.

```
def jose(dat):
    for bit in range(0, 8):
        GPIO.output(SDI, 0x80 & (dat << bit))
        GPIO.output(SRCLK, GPIO.HIGH)
        time.sleep(0.001)
        GPIO.output(SRCLK, GPIO.LOW)
        GPIO.output(RCLK, GPIO.HIGH)
        time.sleep(0.001)
        GPIO.output(RCLK, GPIO.LOW)
```

Definimos una nueva función en la que creamos una variable que nos hará de contador. A seguido, creamos un bucle en el que si pulsamos el primer botón, nos suma uno al contador. También definimos que, si sobrepasamos el número 9 en el contador, este se restablezca en 0. De esta manera, si nos hemos pasado de número, solo tendremos que dar la vuelta de nuevo. Para evitar fallos en el sistema, se establece que no se puede añadir un número al contador hasta que no se haya soltado el botón, de forma que si dejamos pulsado el contador, este no seguirá añadiendo números. Para saber en qué número nos encontramos exactamente, cada vez que se suma uno al contador, se llama a la función anterior, enviando el valor del contador para que recorra el array y nos reproduzca el

número en el visualizador de siete segmentos. El bucle termina cuando, tras establecer el contador en el número que queremos, pulsamos el segundo botón, devolviendo el valor actual del contador a la variable que llame a la función.

```
def numeroClave():
    contador=0
    while True:
        if GPIO.input(BtnA)==True:
            contador +=1
            if contador > 9:
                contador=0
            print(contador)
            jose(segCode[contador])
            while GPIO.input(BtnA)==True:
                if GPIO.input(BtnA)==False:
                    break
            if GPIO.input(BtnB)==True:
                return contador
```

Por último, creamos un bucle while en el que se llama cuatro veces a la función anterior, devolviendo cada vez el valor que seleccionemos en esta a cada una de las cuatro variables que la llamen.

```
while True:
    jose(segCode[0])
    if GPIO.input(BtnB)==False:
        contadorA=numeroClave()
        print("Su numero a es ", contadorA)
        while True:
            jose(segCode[0])
            if GPIO.input(BtnB)==False:
                contadorB=numeroClave()
                print("Su numero b es ", contadorB)
                while True:
                    jose(segCode[0])
                    if GPIO.input(BtnB)==False:
                        contadorC=numeroClave()
                        print("Su numero c es ", contadorC)
                        while True:
                            jose(segCode[0])
                            if GPIO.input(BtnB)==False:
                                contadorD=numeroClave()
                                print("Su numero d es ", contadorD)
                                jose(segCode[10])
                                break
                            break
                        break
```

```
break
```

```
break
```

Finalmente, se muestra la contraseña juntando todos los números y posteriormente, se cambia el tipo de dato INT a CHAR para poder utilizarlo como una cadena de caracteres.

```
print("Su contraseña es", contadorA, contadorB, contadorC, contadorD)
contraseña=str(contadorA)+str(contadorB)+str(contadorC)+str(contadorD)
print(contraseña)
```

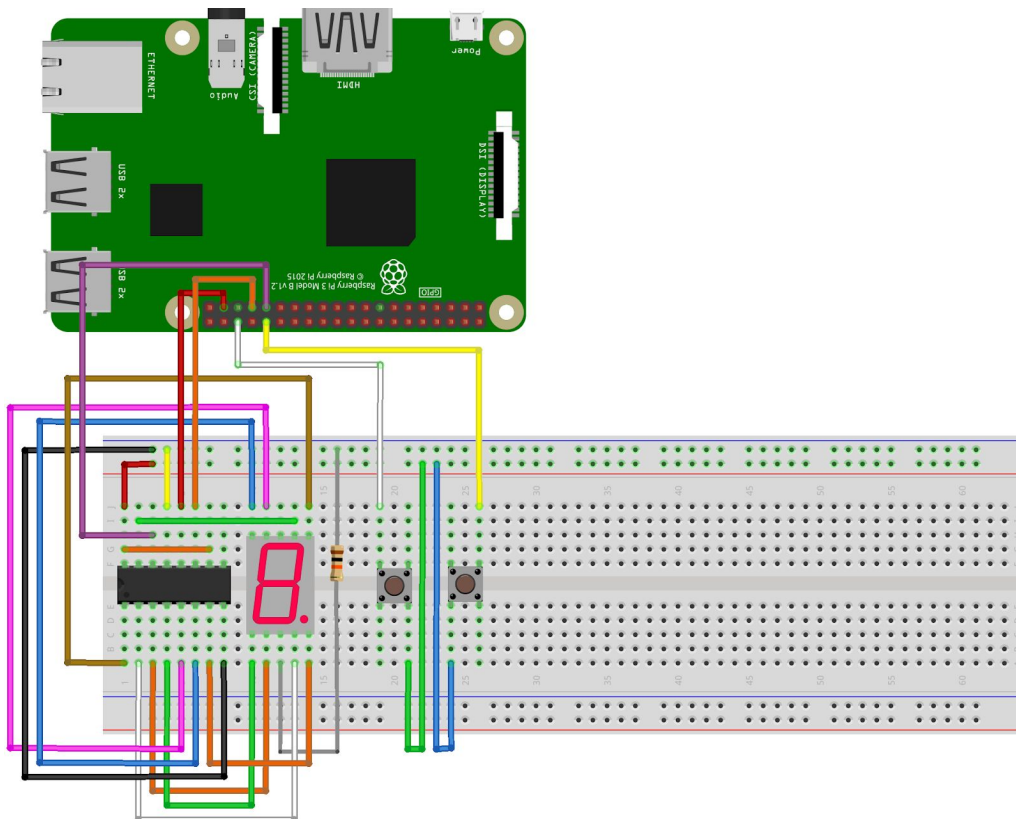
A continuación se presenta una foto de como queda el programa:

```
1 import RPi.GPIO as GPIO
2 import time
3 import pymysql
4 GPIO.setwarnings(False)
5 GPIO.setmode(GPIO.BCM)
6
7 BtnA=12
8 BtnB=16
9
10 SDI = 6
11 RCLK = 26
12 SRCLK = 13
13 segCode = [0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f,0x80]
14
15 GPIO.setup(BtnA, GPIO.OUT)
16 GPIO.setup(BtnB, GPIO.OUT)
17 GPIO.setup(SDI, GPIO.OUT, initial=GPIO.LOW)
18 GPIO.setup(RCLK, GPIO.OUT, initial=GPIO.LOW)
19 GPIO.setup(SRCLK, GPIO.OUT, initial=GPIO.LOW)
20
21 def jose(dat):
22     for bit in range(0, 8):
23         GPIO.output(SDI, 0x80 & (dat << bit))
24         GPIO.output(SRCLK, GPIO.HIGH)
25         time.sleep(0.001)
26         GPIO.output(SRCLK, GPIO.LOW)
27     GPIO.output(RCLK, GPIO.HIGH)
28     time.sleep(0.001)
29     GPIO.output(RCLK, GPIO.LOW)
30
31 def numeroClave():
32     contador=0
33     while True:
34         if GPIO.input(BtnA)==True:
35             contador +=1
36             if contador > 9:
37                 contador=0
38                 print(contador)
```

```

39     jose(segCode[contador])
40     while GPIO.input(BtnA)==True:
41         if GPIO.input(BtnA)==False:
42             break
43     if GPIO.input(BtnB)==True:
44         return contador
45
46
47 while True:
48     jose(segCode[0])
49     if GPIO.input(BtnB)==False:
50         contadorA=numeroClave()
51         print("Su numero a es ", contadorA)
52         while True:
53             jose(segCode[0])
54             if GPIO.input(BtnB)==False:
55                 contadorB=numeroClave()
56                 print("Su numero b es ", contadorB)
57                 while True:
58                     jose(segCode[0])
59                     if GPIO.input(BtnB)==False:
60                         contadorC=numeroClave()
61                         print("Su numero c es ", contadorC)
62                         while True:
63                             jose(segCode[0])
64                             if GPIO.input(BtnB)==False:
65                                 contadorD=numeroClave()
66                                 print("Su numero d es ", contadorD)
67                                 jose(segCode[10])
68                                 break
69                             break
70                         break
71                 break
72         break
73
74 print("Su contraseña es", contadorA, contadorB, contadorC, contadorD)
75 contraseña=str(contadorA)+str(contadorB)+str(contadorC)+str(contadorD)
76 print(contraseña)

```



## ANEXO V - Autentificador en tres pasos

“Autentificador.py” es el programa encargado de unificar los tres programas previos basados en la autenticación. Además, es el programa final que corre nuestras cerraduras y por ende, el más importante junto con “lector.py”. “Autentificador.py” se inicializa automáticamente nada más encender la RaspBerry gracias a la línea:

```
@/usr/bin/python3 /home/pi/Desktop/face_recognitionOpenCv2-master/autenticador.py
```

en el archivo:

```
sudo nano /etc/xdg/lxsession/LXDE-pi/autostart
```

Lo primero que hacemos en el archivo, es incluir las librerías que hemos utilizado en cada uno de los programas:

```
import RPi.GPIO as GPIO
import pymysql
from mfrc522 import SimpleMFRC522
import os
import time
from time import sleep
import webbrowser
import cv2
import pickle
```

Definimos todas las variables (tanto las de los leds, como las del servomotor, las los botones, las del visualizador de 7 segmentos, las del lector RFID y las de la biometría):

```
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(7,GPIO.OUT)
p = GPIO.PWM(7,50)
p.start(7.5)
verde=27
rojo=17
pulsado = 2
sin_pulsar = 22

GPIO.setup(verde, GPIO.OUT)
GPIO.setup(rojo, GPIO.OUT)
GPIO.setup(pulsado, GPIO.OUT)
GPIO.setup(sin_pulsar, GPIO.OUT)

GPIO.setup(2, GPIO.IN)
GPIO.setup(22, GPIO.OUT)
GPIO.output(verde, False)
```

```
GPIO.output(rojo, True)
```

```
reader = SimpleMFRC522()
```

```
rutaCascade = "Cascades/haarcascade_frontalface_alt2.xml"
```

```
caraCascade = cv2.CascadeClassifier(rutaCascade)
```

```
ojosCascade = cv2.CascadeClassifier("Cascades/haarcascade_eye.xml")
```

```
bocaCascade = cv2.CascadeClassifier("Cascades/haarcascade_smile.xml")
```

```
reconocer = cv2.face.LBPHFaceRecognizer_create()
```

```
reconocer.read("entrenamiento.yml")
```

```
etiquetas_a = {"name_persona" : 1 }
```

```
with open("labels.pickle", 'rb') as f:
```

```
    etiquetas_b = pickle.load(f)
```

```
    etiquetas_a = { v:k for k,v in etiquetas_b.items() }
```

```
web_cam = cv2.VideoCapture(0)
```

Creamos una función que se encarga del parpadeo del led rojo, apagandolo y encendiendolo una sola vez:

```
def parpadeoRojo():
```

```
    sleep(0.25)
```

```
    GPIO.output(rojo, False)
```

```
    sleep(0.25)
```

```
    GPIO.output(rojo, True)
```

Creamos la función que se activa una vez superamos los tres niveles de autenticación. Esta, al igual que el resto del programa, posee varios "print" que nos sirven de referente cuando activamos el programa desde la consola de comandos, pero a la hora de su implementación los print no serán visibles, así que en esta explicación los obviaremos. Una vez empieza esta función, lo primero que hace es activar el servomotor que abre la cerradura a la vez que cambia la luz led verde en lugar de la roja, para indicarnos que podemos entrar:

```
def correctPassword(a):
```

```
    p.ChangeDutyCycle(3)
```

```
    GPIO.output(verde, True)
```

```
    GPIO.output(rojo, False)
```

A seguido, comienza un bucle while que deja el programa en standby hasta que cerramos de nuevo la puerta, pulsando así el botón que hace de tope.

```
while True:
```

```
    if GPIO.input(pulsado):
```

```
        GPIO.output(sin_pulsar, False)
```

```
    else:
```

```
GPIO.output(sin_pulsar, True)
GPIO.output(sin_pulsar, GPIO.LOW)
```

Esto acciona de nuevo el servomotor que devuelve la cerradura a su estado original, de forma que la puerta queda bloqueada de nuevo. Una vez completado el cierre, la luz verde se vuelve a apagar dando paso a la luz roja y finalizando la función.

```
p.ChangeDutyCycle(7.5)
sleep(1)
break
GPIO.output(verde, False)
GPIO.output(rojo, True)
os.system('clear') #sirve para limpiar la pantalla del terminal
```

La siguiente función con la que nos encontramos es la que se activa si fallamos en alguno de los procesos de autenticación. Esta función llamará repetidamente a la función del parpadeo rojo para notificar de forma visual que no tenemos el acceso permitido.

```
def wrongPassword():
    while(pr<12):
        parpadeoRojo()
        pr+=1
```

En la siguiente función, nos encontramos con la primera prueba de autenticación. En ella, conectamos con la base de datos de “usuarios” y comparamos el ID que se nos envía a la función, con todos los IDs de la tabla de “admitidos” uno por uno, no sin antes haber creado un contador. Si hay alguna coincidencia entre el ID y los registrados en la tabla, se le suma uno al contador, de forma que cuando se terminan de consultar todos los IDs, si el contador es mayor de cero significa que ha habido alguna coincidencia y por ende, que estamos registrados y podemos continuar con el proceso de autenticación. Para ello, si el contador es mayor de cero, se devuelve un uno a la variable que llama a la función, y cero si el valor del contador es cero.

```
def mandarId(a):
    print (a)
    contador=0;
    conexion = pymysql.connect(host='192.168.1.47', user='ja', password='rootroot', database='usuarios')
    cursor = conexion.cursor()

    todo="select * from admitidos;"

    cursor.execute(todo)
    filas=cursor.fetchall()
    for fila in filas:
        if(a==fila[1]):
            contador+=1
```



```

conexion.commit()
conexion.close()
if(contador>0):
    return 1
else:
    return 0

```

A seguido se encuentra la función de la contraseña. En ella se declaran las variables de los botones y del visualizador de siete segmentos junto con el array de números interpretables:

```

def botonesPwd():
    BtnA=12
    BtnB=16
    SDI    = 6
    RCLK   = 26
    SRCLK  = 13
    segCode = [0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f,0x80]
    GPIO.setup(BtnA, GPIO.OUT)
    GPIO.setup(BtnB, GPIO.OUT)
    GPIO.setup(SDI, GPIO.OUT, initial=GPIO.LOW)
    GPIO.setup(RCLK, GPIO.OUT, initial=GPIO.LOW)
    GPIO.setup(SRCLK, GPIO.OUT, initial=GPIO.LOW)

```

Dentro de esta función, como veíamos en el [ANEXO IV](#), definimos la función que recorre el array de números:

```

def numArray(dat):
    for bit in range(0, 8):
        GPIO.output(SDI, 0x80 & (dat << bit))
        GPIO.output(SRCLK, GPIO.HIGH)
        time.sleep(0.001)
        GPIO.output(SRCLK, GPIO.LOW)
        GPIO.output(RCLK, GPIO.HIGH)
        time.sleep(0.001)
        GPIO.output(RCLK, GPIO.LOW)

```

además de la función para establecer el número que queremos marcar, la cual incrementa el valor cada vez que pulsamos el botón uno y nos lo muestra en el visualizador de siete segmentos, volviendo al cero cada vez que superamos el número nueve. Una vez que estamos conformes con el dígito marcado, pulsamos el segundo botón, y esto devuelve el número a la función que lo engloba, finalizando esta misma.

```

def numeroClave():
    contador=0
    while True:
        if GPIO.input(BtnA)==True:
            contador +=1

```

```

        if contador > 9:
            contador=0
            numArray(segCode[contador])
            while GPIO.input(BtnA)==True:
                if GPIO.input(BtnA)==False:
                    break
            if GPIO.input(BtnB)==True:
                return contador

```

Sin salir de la función perteneciente a la segunda autenticación, creamos un bucle while en el que cuatro variables distintas llamaran a la función anterior, para obtener cada una de las cuatro cifras que conforman la contraseña.

```

while True:
    numArray(segCode[0])
    if GPIO.input(BtnB)==False:
        contadorA=numeroClave()
        while True:
            numArray(segCode[0])
            if GPIO.input(BtnB)==False:
                contadorB=numeroClave()
                while True:
                    numArray(segCode[0])
                    if GPIO.input(BtnB)==False:
                        contadorC=numeroClave()
                        while True:
                            numArray(segCode[0])
                            if GPIO.input(BtnB)==False:
                                contadorD=numeroClave()
                                numArray(segCode[10])
                                break
                        break
                    break
                break
            break
        break

```

Esta contraseña es transformada de formato INT a formato CHAR para poderla comparar como cadena de caracteres.

```

contraseña=str(contadorA)+str(contadorB)+str(contadorC)+str(contadorD)

```

La función continúa abriendo la base de datos de “autenticaciontres” y leyendo la tabla con la contraseña de la puerta actual fila por fila, aunque en este caso, cada puerta solo consta de una fila pues solo tiene una contraseña. Si la contraseña coincide, se devuelve a la variable que solicitara esta función un uno, mientras que si no coincide, devuelve un cero.

```

conexion = pymysql.connect(host='192.168.1.47',user='ja',password='rootroot',database='autenticaciontres')
cursor = conexion.cursor()

```

```
todo="select * from puertas;"
```

```
cursor.execute(todo)
filas=cursor.fetchall()
contador=0
for fila in filas:
    if(contraseña==fila[0]):
        contador+=1
```

```
conexion.commit()
```

```
conexion.close()
if(contador>0):
    return 1
else:
    return 0
```

La función finaliza y pasamos a la siguiente, encargada de la autenticación biométrica (Esta parte del programa se encuentra explicada en detalle en el [ANEXO III](#)). La función comienza declarando dos variables:

```
def bioMetria():
    similitudes = 0
    nosimilitudes=0
```

En ese momento la cámara se enciende y empieza a buscar coincidencias. Para evitar errores con objetos inanimados, se establece que haya un mínimo de similitud superior al 4%. Si las similitudes son superiores al 50%, la cámara nos identifica como “Desconocidos” mientras que si es superior al 90% nos pone nuestro nombre sobre la cabeza. Para evitar posibles fallos, la cámara aguanta un rato si el resultado es “Desconocido” ya que tenemos que coincidir al menos 10 veces durante 0.02 segundos como tal para que nos rechace. Esta es una medida de precaución para evitar que nada más nos enfoque la cámara y nos pille un mal ángulo, nos deniegue la entrada. A su vez, la cámara tiene que detectarse durante un breve espacio de tiempo como la persona que somos, para evitar de igual manera, fallos con personas muy similares.

Si el resultado es “Desconocido”, la función devuelve cero a la variable que la llama, mientras que si nos reconoce correctamente, devuelve uno. En ambos casos, la función finaliza.

```
while True:
    ret, recuadro = web_cam.read()
    gris = cv2.cvtColor(recuadro, cv2.COLOR_BGR2GRAY)
    caras = caraCascade.detectMultiScale(gris, 1.5, 5)
    for (x, y, w, h) in caras:
        roi_gris = gris[y:y+h, x:x+w]
```

```

        roi_color = recuadro[y:y+h, x:x+w]
        id_, simil_conf = reconocer.predict(roi_gris)

        if simil_conf >= 4 and simil_conf < 85:
            font = cv2.FONT_HERSHEY_SIMPLEX
            name = etiquetas_a[id_]
            if simil_conf > 70:
                name = "Desconocido"
                nosimilitudes += 1
                sleep(0.02)
                if nosimilitudes > 10:
                    cv2.destroyAllWindows()
                    return 0
            if name != "Desconocido" and name != "":
                similitudes += 1
                sleep(0.02)
                if similitudes > 10:
                    cv2.destroyAllWindows()
                    return 1
            color = (255,255,255)
            ancho = 2
            cv2.putText(recuadro, name, (x,y), font, 1, color, ancho, cv2.LINE_AA)
            img_item = "my-image.png"
            cv2.imwrite(img_item, roi_gris)
            cv2.rectangle(recuadro, (x, y), (x+w, y+h), (0, 255, 0), 2)
            caract = bocaCascade.detectMultiScale(roi_gris)
            for(ex,ey,ew,eh) in caract:
                cv2.rectangle(roi_color, (ex, ey), (ex+ew, ey+eh), (0, 255, 0), 2)
            reprod_recuadro = cv2.resize(recuadro, (1200, 650), interpolation = cv2.INTER_CUBIC)
            cv2.imshow('Detectando caras', reprod_recuadro)
            if cv2.waitKey(1) & 0xFF == ord('q'):
                break

```

La última función del programa corresponde a la notificación de denegados en la tabla de "historial\_denegados" de la base de datos "usuarios". La función recibe el ID y lo ingresa en la tabla junto con la fecha y hora de ese preciso instante.

```

def notificaDenegados(a):
    print(a)
    conexion = pymysql.connect(host='192.168.1.47', user='ja', password='rootroot', database='usuarios')
    cursor = conexion.cursor()
    tododos="insert into historial_x (numero_de_usuario, fecha_hora) values(%s, now());"
    cursor.execute(tododos, (a))
    conexion.commit()
    conexion.close()

```

```
os.system('clear')
```

Para concluir con el programa, tenemos la parte que activa las funciones anteriores dependiendo del camino que tomemos. Empieza con el bucle infinito en el que se buscan señales de radiofrecuencia.

```
try:
    while True:
        id, text = reader.read()
        a=id
```

Una vez encuentra una, activa la función referente al primer paso de autenticación.

```
    if(a>=1):
        autuno=mandarId(a)
        print("Primera autenticacion: ", autuno)
```

Si recibe como resultado un uno, es que se ha realizado con éxito, y pasa a la siguiente autenticación.

```
        if(autuno==1):
            autdos=bioMetria()
            print("Segunda autenticacion: ", autdos)
```

Una vez más, si el resultado vuelve a ser un uno, se sigue el proceso de autenticación.

```
            if(autdos==1):
                auttres=botonesPwd()
                print("Tercera autenticacion: ", auttres)
```

Si vuelve a ser positivo, recibimos otro uno como respuesta, y el programa conecta con las tablas que guardan tanto el historial de la puerta en cuestión, como el historial conjunto, e introducir nuestro ID en ellas junto con la fecha y hora de este momento.

```
                if(auttres==1):
                    conexion =
pymysql.connect(host='192.168.1.47', user='ja', password='rootroot',
database='usuarios')
```

```
                    cursor = conexion.cursor()
                    tododos="insert into historial
(numero_de_usuario, fecha_hora) values(%s, now());"
```

```
                    cursor.execute(tododos, (a))
                    tododos="insert into historial_p
(numero_de_usuario, fecha_hora) values(%s, now());"
```

```
cursor.execute ( todos , ( a ) )
```

Una vez hecho esto, el programa llama a la función que nos abre la puerta y nos deja pasar.

```
correctPassword ( a )  
conexion.commit ()  
conexion.close ()
```

Finalmente se define que si en cualquiera de los anteriores pasos de autenticación hemos sido rechazados, que se recurra a la función que advierte del error en el intento de entrada y a la que notifica tal intento en la tabla de "historial\_denegados".

```
else:  
    wrongPassword ()  
    notificaDenegados ( a )  
else:  
    wrongPassword ()  
    notificaDenegados ( a )
```

```
else:  
    wrongPassword ()  
    notificaDenegados ( a )
```

```
else:  
    print ( "mal" )
```

```
finally:  
    GPIO.cleanup ()
```

A continuación dejamos las fotos del código en cuestión, con un pequeño añadido visual a base de "print"s para su aplicación desde la consola de comandos:

```

1  #!/usr/bin/python
2  import RPi.GPIO as GPIO
3  import pymysql
4  from mfrc522 import SimpleMFRC522
5  import os
6  import time
7  from time import sleep
8  import webbrowser
9  import cv2
10 import pickle
11
12 GPIO.setmode(GPIO.BCM)
13 GPIO.setwarnings(False)
14 GPIO.setup(7,GPIO.OUT)
15 p = GPIO.PWM(7,50)
16 p.start(7.5)
17 verde=27
18 rojo=17
19 pulsado = 2
20 sin_pulsar = 22
21
22 GPIO.setup(verde, GPIO.OUT)
23 GPIO.setup(rojo, GPIO.OUT)
24 GPIO.setup(pulsado, GPIO.OUT)
25 GPIO.setup(sin_pulsar, GPIO.OUT)
26
27 GPIO.setup(2, GPIO.IN)
28 GPIO.setup(22, GPIO.OUT)
29 GPIO.output(verde, False)
30 GPIO.output(rojo, True)
31
32 reader = SimpleMFRC522()
33
34 rutaCascade = "Cascades/haarcascade_frontalface_alt2.xml"
35 caraCascade = cv2.CascadeClassifier(rutaCascade)
36 ojosCascade = cv2.CascadeClassifier("Cascades/haarcascade_eye.xml")
37 bocaCascade = cv2.CascadeClassifier("Cascades/haarcascade_smile.xml")
38 reconocer = cv2.face.LBPHFaceRecognizer_create()
39 reconocer.read("entrenamiento.yml")
40 etiquetas_a = {"name_persona" : 1 }
41
42 with open("labels.pickle",'rb') as f:
43     etiquetas_b = pickle.load(f)
44     etiquetas_a = { v:k for k,v in etiquetas_b.items()}
45
46 web_cam = cv2.VideoCapture(0)
47
48 def parpadeoRojo():
49     sleep(0.25)
50     GPIO.output(rojo, False)
51     sleep(0.25)
52     GPIO.output(rojo, True)
53
54 def correctPassword(a):
55     print ("Contraseña aceptada")
56     sleep(1)
57     print ()
58     print ("*****", "*" * len(text.replace(" ", "")), sep="")
59     print ("* Buenos dias ", text.replace(" ", ""), " *", sep="")
60     print ("*****", "*" * len(text.replace(" ", "")), sep="")
61     sleep(1)
62     print ()
63     sleep(1)
64     print ("Abriendo cerradura")
65     print ()
66     sleep(1)
67     p.ChangeDutyCycle(3)
68     print("Cerradura abierta.")a
69     GPIO.output(verde, True)
70     GPIO.output(rojo, False)
71     print ()
72     sleep(1)
73
74 while True:
75     if GPIO.input(pulsado):
76         GPIO.output(sin_pulsar, False)
77     else:
78         GPIO.output(sin_pulsar, True)
79         GPIO.output(sin_pulsar, GPIO.LOW)
80         p.ChangeDutyCycle(7.5)
81         sleep(1)
82         break
83     print("Hasta pronto")
84     GPIO.output(verde, False)
85     GPIO.output(rojo, True)
86     print ()
87     sleep(2)
88     os.system('clear')

```

```

89
90 def wrongPassword():
91     parpadeoRojo()
92     GPIO.output(rojo, False)
93     print ("#####")
94     print ("# Usuario erróneo #")
95     print ("#####")
96     parpadeoRojo()
97     print ("|||||")
98     parpadeoRojo()
99     print ("|||||")
100    parpadeoRojo()
101    print ("|||||")
102    parpadeoRojo()
103    print ("VVVVVVVVVVVVVVVV")
104    parpadeoRojo()
105    print ("Contraseña Denegada")
106    pr=0
107    while(pr<12):
108        parpadeoRojo()
109        pr+=1
110    GPIO.output(rojo, True)
111    os.system('clear')
112
113 def mandarId(a):
114     print (a)
115     contador=0;
116     conexion = pymysql.connect(host='192.168.1.47', user='ja', password='rootroot', database='usuarios')
117     cursor = conexion.cursor()
118
119     todo="select * from admitidos;"
120
121     cursor.execute(todo)
122     filas=cursor.fetchall()
123     for fila in filas:
124         if(a==fila[1]):
125
126             print(fila[1])
127             contador+=1
128
129     print(contador)
130
131     conexion.commit()
132
133     conexion.close()
134     if(contador>0):
135         return 1
136     else:
137         return 0
138
139 def botonesPwd():
140
141     BtnA=12
142     BtnB=16
143
144     SDI = 6
145     RCLK = 26
146     SRCLK = 13
147     segCode = [0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f,0x80]
148
149     GPIO.setup(BtnA, GPIO.OUT)
150     GPIO.setup(BtnB, GPIO.OUT)
151     GPIO.setup(SDI, GPIO.OUT, initial=GPIO.LOW)
152     GPIO.setup(RCLK, GPIO.OUT, initial=GPIO.LOW)
153     GPIO.setup(SRCLK, GPIO.OUT, initial=GPIO.LOW)
154
155     def numArray(dat):
156         for bit in range(0, 8):
157             GPIO.output(SDI, 0x80 & (dat << bit))
158             GPIO.output(SRCLK, GPIO.HIGH)
159             time.sleep(0.001)
160             GPIO.output(SRCLK, GPIO.LOW)
161             GPIO.output(RCLK, GPIO.HIGH)
162             time.sleep(0.001)
163             GPIO.output(RCLK, GPIO.LOW)
164
165     def numeroClave():
166         contador=0
167         while True:
168             if GPIO.input(BtnA)==True:
169                 contador +=1
170                 if contador > 9:
171                     contador=0
172                 print(contador)
173                 numArray(segCode[contador])
174                 while GPIO.input(BtnA)==True:
175                     if GPIO.input(BtnA)==False:

```



```

176         break
177     if GPIO.input(BtnB)==True:
178         return contador
179
180 while True:
181     numArray(segCode[0])
182     if GPIO.input(BtnB)==False:
183         contadorA=numeroClave()
184         print("Su numero a es ", contadorA)
185         while True:
186             numArray(segCode[0])
187             if GPIO.input(BtnB)==False:
188                 contadorB=numeroClave()
189                 print("Su numero b es ", contadorB)
190                 while True:
191                     numArray(segCode[0])
192                     if GPIO.input(BtnB)==False:
193                         contadorC=numeroClave()
194                         print("Su numero c es ", contadorC)
195                         while True:
196                             numArray(segCode[0])
197                             if GPIO.input(BtnB)==False:
198                                 contadorD=numeroClave()
199                                 print("Su numero d es ", contadorD)
200                                 numArray(segCode[10])
201                                 break
202                             break
203                         break
204                     break
205                 break
206         print("Su contraseña es", contadorA, contadorB, contadorC, contadorD)
207         contraseña=str(contadorA)+str(contadorB)+str(contadorC)+str(contadorD)
208         print(contraseña)
209         conexion = pymysql.connect(host='192.168.1.47', user='ja', password='rootroot', database='autenticaciontres')
210         cursor = conexion.cursor()
211
212         todo="select * from puertas;"
213
214         cursor.execute(todo)
215         filas=cursor.fetchall()
216         contador=0
217         for fila in filas:
218             if(contraseña==fila[0]):
219                 print(contraseña)
220                 contador+=1
221
222         print(contador)
223
224         conexion.commit()
225
226         conexion.close()
227         if(contador>0):
228             return 1
229         else:
230             return 0
231
232 def bioMetria():
233     similitudes = 0
234     nosimilitudes=0
235     while True:
236         ret, recuadro = web_cam.read()
237         gris = cv2.cvtColor(recuadro, cv2.COLOR_BGR2GRAY)
238         caras = caraCascade.detectMultiScale(gris, 1.5, 5)
239
240         for (x, y, w, h) in caras:
241             roi_gris = gris[y:y+h, x:x+w]
242             roi_color = recuadro[y:y+h, x:x+w]
243             id_, simil_conf = reconocer.predict(roi_gris)
244
245             if simil_conf >= 4 and simil_conf < 85:
246                 font = cv2.FONT_HERSHEY_SIMPLEX
247                 name = etiquetas_a[id_]
248
249                 if simil_conf > 50:
250                     name = "Desconocido"
251                     nosimilitudes += 1
252                     sleep(0.02)
253                     if nosimilitudes > 10:
254                         cv2.destroyAllWindows()
255                         return 0
256                 if name != "Desconocido" and name != "":
257                     similitudes += 1
258                     sleep(0.02)
259                     if similitudes > 10:
260                         cv2.destroyAllWindows()
261                         return 1
262                 color = (255,255,255)

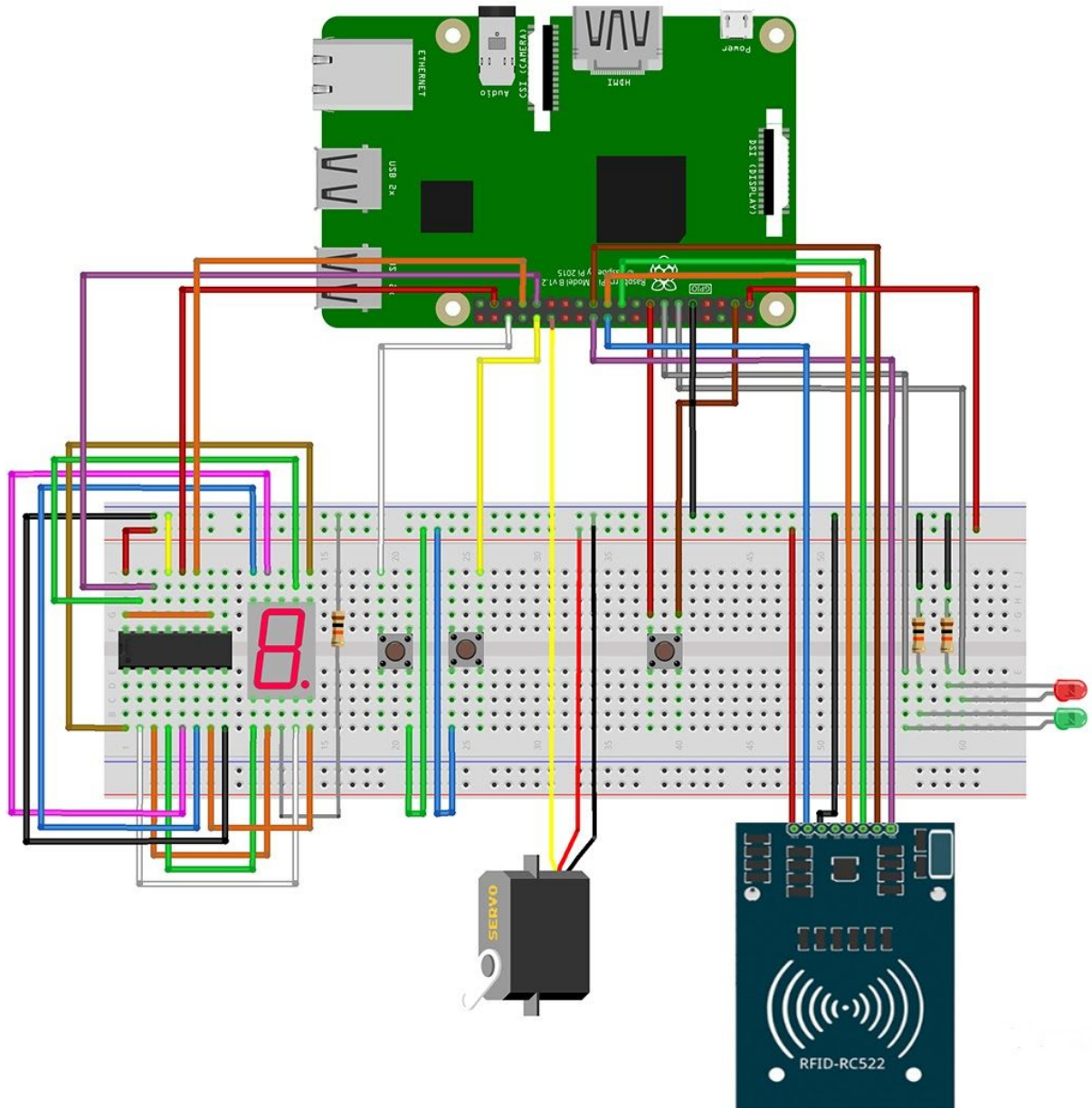
```

```

263         ancho = 2
264         cv2.putText(recuadro, name, (x,y), font, 1, color, ancho, cv2.LINE_AA)
265
266     img_item = "my-image.png"
267     cv2.imwrite(img_item, roi_gris)
268     cv2.rectangle(recuadro, (x, y), (x+w, y+h), (0, 255, 0), 2)
269     caract = bocaCascade.detectMultiScale(roi_gris)
270
271     for(ex,ey,ew,eh) in caract:
272         cv2.rectangle(roi_color, (ex, ey), (ex+ew, ey+eh), (0, 255, 0), 2)
273
274     reprod_recuadro = cv2.resize(recuadro, (1200, 650), interpolation = cv2.INTER_CUBIC)
275     cv2.imshow('Detectando caras', reprod_recuadro)
276
277     if cv2.waitKey(1) & 0xFF == ord('q'):
278         break
279
280 def notificaDenegados(a):
281     print(a)
282     conexion = pymysql.connect(host='192.168.1.47', user='ja', password='rootroot', database='usuarios')
283     cursor = conexion.cursor()
284     todos="insert into historial_denegados (numero_de_usuario, fecha_hora) values(%, now());"
285     cursor.execute(todos, (a))
286     conexion.commit()
287     conexion.close()
288
289 os.system('clear')
290
291 try:
292     while True:
293         id, text = reader.read()
294         a=id
295         print(a)
296         if(a>=1):
297             autuno=mandarId(a)
298             print("Primera autentificacion: ", autuno)
299             if(autuno==1):
300                 autdos=bioMetria()
301                 print("Segunda autentificacion: ", autdos)
302                 if(autdos==1):
303                     auttres=botonesPwd()
304                     print("Tercera autentificacion: ", auttres)
305                 cursor.execute(todos, (a))
306                 conexion.commit()
307                 conexion.close()
308
309 os.system('clear')
310
311 try:
312     while True:
313         id, text = reader.read()
314         a=id
315         print(a)
316         if(a>=1):
317             autuno=mandarId(a)
318             print("Primera autentificacion: ", autuno)
319             if(autuno==1):
320                 autdos=bioMetria()
321                 print("Segunda autentificacion: ", autdos)
322                 if(autdos==1):
323                     auttres=botonesPwd()
324                     print("Tercera autentificacion: ", auttres)
325                     if(auttres==1):
326                         conexion = pymysql.connect(host='192.168.1.47', user='ja', password='rootroot', database='usuarios')
327                         cursor = conexion.cursor()
328                         todos="insert into historial (numero_de_usuario, fecha_hora) values(%, now());"
329                         cursor.execute(todos, (a))
330                         todos="insert into historial_p (numero_de_usuario, fecha_hora) values(%, now());"
331                         cursor.execute(todos, (a))
332                         correctPassword(a)
333                         conexion.commit()
334                         conexion.close()
335                     else:
336                         wrongPassword()
337                         notificaDenegados(a)
338                 else:
339                     wrongPassword()
340                     notificaDenegados(a)
341             else:
342                 wrongPassword()
343                 notificaDenegados(a)
344         else:
345             print ("mal")
346
347 finally:
348     GPIO.cleanup()

```

Por ultimo, la foto del ensamblado del mismo:



## ANEXO VI - Suricata

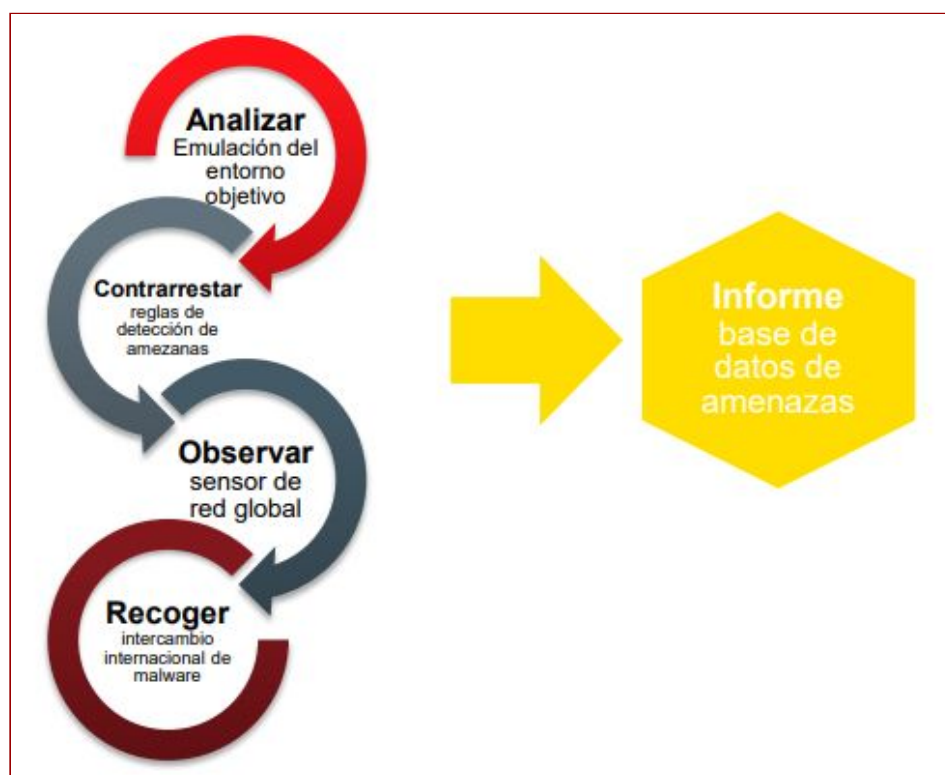
### Análisis de la herramienta Suricata

Un IDS es un sistema de detección de intrusos que nos permite incrementar la seguridad de nuestras redes, se encarga de analizar el tráfico detectando los comportamientos extraños y las actividades sospechosas o dañinas.

Suricata es un motor de red de alto rendimiento IDS (también puede actuar como IPS). Esta es una aplicación de código abierto multiplataforma desarrollado por Open Information Security Foundation (OISF), una fundación sin ánimo de lucro. Suricata es propiedad de OISF.

Está basado en un conjunto de reglas que se desarrollan para controlar el tráfico de red y aportar así alertas al administrador cuando se detecten eventos sospechosos.

### Fases en la gestión de amenazas



### Características

- Identifica los principales protocolos de red, por lo que reconoce en todo momento el tráfico generado en el sistema.
  - Controla los archivos que viajan por la red identificando sus diferentes formatos y verificando que no han sido manipulados.
  - Es multihilo, es decir, permite una configuración que distribuye los procesos en los

diferentes núcleos, que desemboca a un mejor rendimiento.

- Controla las peticiones DNS, se monitorizan y almacenan en los logs establecidos.
- Incluye un soporte de secuencias de comandos LUA para ayudar a localizar amenazas complejas.
- Emplea un sistemas de reglas compatible con Snort, haciendo así más fácil la migración de un sistemas a otro; Con formatos de entrada y salida estándar como YAML y JSON, las integraciones con herramientas como SIEM, Splunk, Kibana y otras bases de datos se realizan sin esfuerzo.

## Estructura de las reglas

- Acción, determina qué sucede cuando un paquete es detectado por la regla.

Variedad de opciones:

- pass
- drop
- reject
- alert

- Cabecera, define el origen y destino de la comunicación donde se producirá una acción.

Estructura:

- Protocolo
- Red de origen
- Puerto de origen
- Dirección
- Red de destino
- Puerto de destino

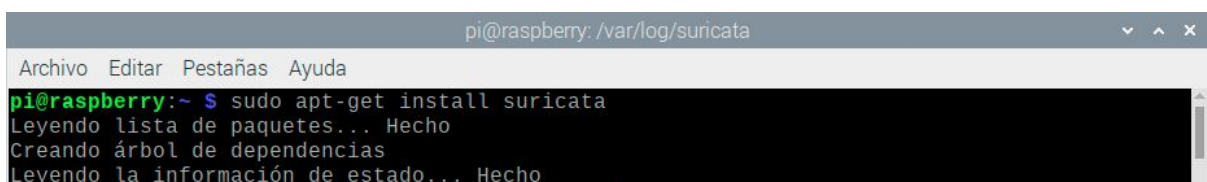
- Las opciones de la regla, son los detalles de la regla, se definen entre paréntesis y cada una de estas irá separada por un punto y coma (;), de cada opción se puede especificar la clave y su valor separados por dos puntos (:).

## Instalación

Antes de proceder a la instalación, debemos tener nuestro sistemas actualizado (los paquetes y versiones), utilizaremos el comando:

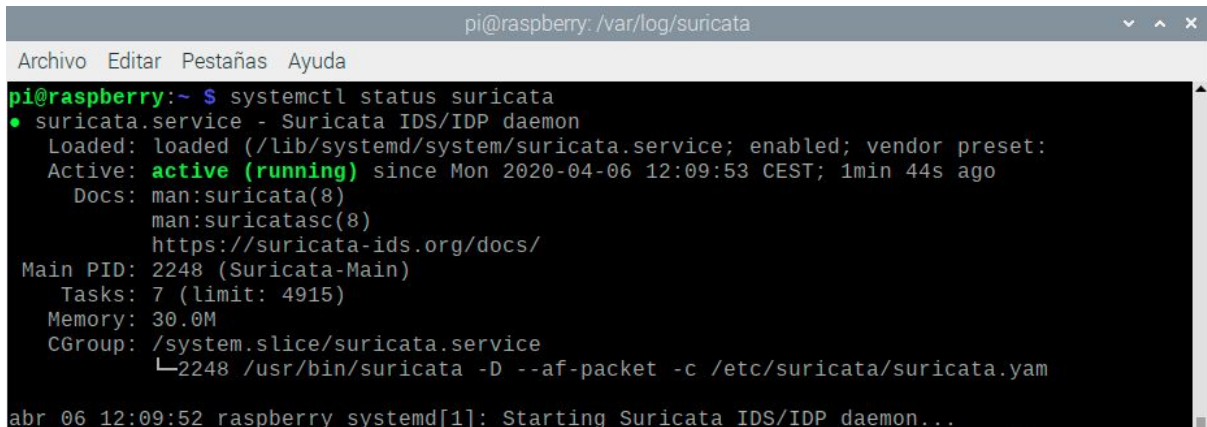
```
sudo apt-get update && sudo apt-get upgrade
```

1. Instalamos la herramienta con el comando `sudo apt-get install suricata`



```
pi@raspberrypi: /var/log/suricata
Archivo Editar Pestañas Ayuda
pi@raspberrypi:~$ sudo apt-get install suricata
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
```

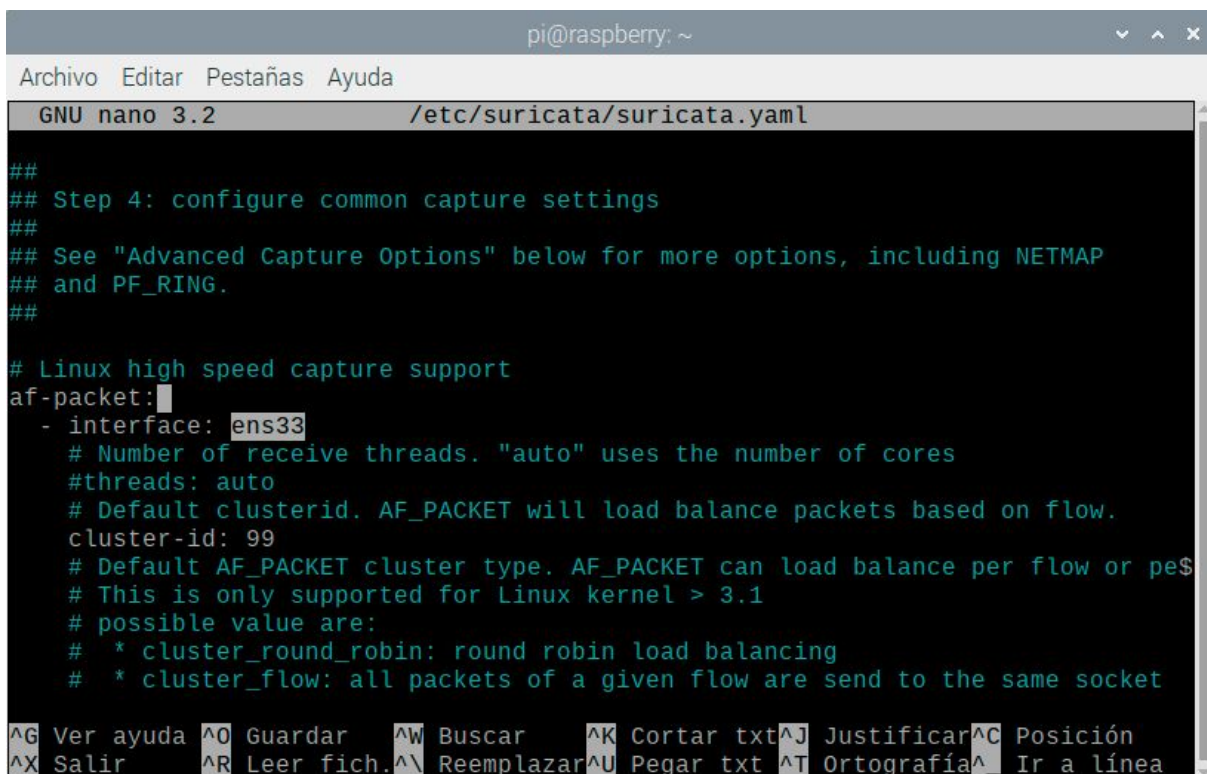
2. Comprobamos que el servicio funciona correctamente y se está ejecutando con el comando `systemctl status suricata`



```
pi@raspberrypi: /var/log/suricata
Archivo Editar Pestañas Ayuda
pi@raspberrypi:~$ systemctl status suricata
● suricata.service - Suricata IDS/IDP daemon
   Loaded: loaded (/lib/systemd/system/suricata.service; enabled; vendor preset:
   Active: active (running) since Mon 2020-04-06 12:09:53 CEST; 1min 44s ago
     Docs: man:suricata(8)
           man:suricatasc(8)
           https://suricata-ids.org/docs/
   Main PID: 2248 (Suricata-Main)
     Tasks: 7 (limit: 4915)
    Memory: 30.0M
    CGroup: /system.slice/suricata.service
            └─2248 /usr/bin/suricata -D --af-packet -c /etc/suricata/suricata.yam
abr 06 12:09:52 raspberrypi systemd[1]: Starting Suricata IDS/IDP daemon...
```

Si hubiera algún error, podemos encontrarlo en el fichero `/var/log/suricata/suricata.log`

3. Dentro del fichero de configuración `/etc/suricata/suricata.yaml`, debemos de cambiar la interfaz de red física predeterminada a `ens33`, esto se debe a que Suricata utilizar el modo de trabajo af-packet sobre la interfaz eth0, y en las últimas versiones de Debian esta interfaz no existe.



```
pi@raspberrypi: ~
Archivo Editar Pestañas Ayuda
GNU nano 3.2 /etc/suricata/suricata.yaml
##
## Step 4: configure common capture settings
##
## See "Advanced Capture Options" below for more options, including NETMAP
## and PF_RING.
##
# Linux high speed capture support
af-packet:
- interface: ens33
  # Number of receive threads. "auto" uses the number of cores
  #threads: auto
  # Default clusterid. AF_PACKET will load balance packets based on flow.
  cluster-id: 99
  # Default AF_PACKET cluster type. AF_PACKET can load balance per flow or peS
  # This is only supported for Linux kernel > 3.1
  # possible value are:
  # * cluster_round_robin: round robin load balancing
  # * cluster_flow: all packets of a given flow are send to the same socket
^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar txt ^J Justificar ^C Posición
^X Salir ^R Leer fich. ^\ Reemplazar ^U Pegar txt ^T Ortografía ^_ Ir a línea
```

4. Actualizamos las reglas mediante el comando `sudo suricata-oinkmaster-updater`, descargando así las últimas reglas de repositorio publicadas por Emerging Threats.

```
pi@raspberrypi: /var/log/suricata
Archivo Editar Pestañas Ayuda
pi@raspberrypi:~$ sudo suricata-oinkmaster-updater
Loading /etc/suricata/suricata-oinkmaster.conf
Downloading file from https://rules.emergingthreats.net/open/suricata-4.0.0/emerging.rules.tar.gz... done.
Archive successfully downloaded, unpacking... done.
Setting up rules structures... done.
Processing downloaded rules... disablesid 0, enablesid 0, modifiesid 0, localsid 0, total rules 28894
Setting up rules structures... done.
Comparing new files to the old ones... done.
Checking flowbits dependencies... no problems found.
Updating local rules files... done.

[***] Results from Oinkmaster started 20200406 12:13:24 [***]

[*] Rules modifications: [*]
    None.

[*] Non-rule line modifications: [*]
    None.

[+] Added files (consider updating your snort.conf to include them if needed): [+]

    -> botcc.portgrouped.rules
```

5. Creamos el fichero de reglas. En nuestro caso lo hemos llamado “**custom.rules**” y lo guardaremos en la siguiente ruta `/etc/suricata/rules/`

```
pi@raspberrypi: ~
Archivo Editar Pestañas Ayuda
pi@raspberrypi:~$ sudo nano /etc/suricata/rules/custom.rules
```

6. Agregaremos el fichero creado en el paso anterior “**custom.rules**” al fichero de configuración de Suricata `/etc/suricata/suricata.yaml` para que pueda aplicarlas sobre el tráfico. Debemos colocarlas a continuación de las reglas que actualizamos anteriormente.

```
pi@raspberrypi: ~
GNU nano 3.2 /etc/suricata/suricata.yaml
# - decoder-events.rules # available in suricata sources under rules dir
# - stream-events.rules # available in suricata sources under rules dir
- http-events.rules # available in suricata sources under rules dir
- smtp-events.rules # available in suricata sources under rules dir
- dns-events.rules # available in suricata sources under rules dir
- tls-events.rules # available in suricata sources under rules dir
# - modbus-events.rules # available in suricata sources under rules dir
# - app-layer-events.rules # available in suricata sources under rules dir
# - dnp3-events.rules # available in suricata sources under rules dir
- custom.rules
# - ntp-events.rules # available in suricata sources under rules dir
# - ipsec-events.rules # available in suricata sources under rules dir
# - kerberos-events.rules # available in suricata sources under rules dir

##
## Auxiliary configuration files.
##
classification-file: /etc/suricata/classification.config

^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar txt ^J Justificar ^C Posición M-U Deshacer
^X Salir ^R Leer fich. ^\ Reemplazar ^U Pegar txt ^T Ortografía ^_ Ir a línea M-E Rehacer
```

7. Para aplicar dichos cambios, hacemos un reload al servicio con el comando `sudo systemctl reload suricata` y comprobamos que funciona correctamente con el comando `sudo systemctl status suricata`

```
pi@raspberrypi: /var/log/suricata
Archivo Editar Pestañas Ayuda
pi@raspberrypi:~$ sudo systemctl reload suricata
pi@raspberrypi:~$ sudo systemctl status suricata
● suricata.service - Suricata IDS/IDP daemon
   Loaded: loaded (/lib/systemd/system/suricata.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2020-04-06 12:09:53 CEST; 7min ago
     Docs: man:suricata(8)
           man:suricatasc(8)
           https://suricata-ids.org/docs/
   Process: 6142 ExecReload=/usr/bin/suricatasc -c reload-rules (code=exited, status=0/SUCCESS)
   Process: 6229 ExecReload=/bin/kill -HUP $MAINPID (code=exited, status=0/SUCCESS)
  Main PID: 2248 (Suricata-Main)
    Tasks: 7 (limit: 4915)
   Memory: 261.8M
   CGroup: /system.slice/suricata.service
           └─2248 /usr/bin/suricata -D --af-packet -c /etc/suricata/suricata.yaml --pidfile /var/run/suricata.pid
```



## 8. Demostración.

```
pi@raspberrypi /var/log/suricata
Archivo Editar Pestañas Ayuda
pi@raspberrypi:~$ tail -8 fast.log
04/06/2020-12:20:00.524342 [**] [1:0:0] ICMP protocolo de control de mensajes de internet DETECTADO
[**] [Classification: (null)] [Priority: 3] {IPv6-ICMP} fe80:0000:0000:0000:0478:b1c6:6f95:3305:143
-> ff02:0000:0000:0000:0000:0000:0016:0
04/06/2020-12:22:40.883907 [**] [1:0:0] ICMP protocolo de control de mensajes de internet DETECTADO
[**] [Classification: (null)] [Priority: 3] {IPv6-ICMP} fe80:0000:0000:0000:0478:b1c6:6f95:3305:143
-> ff02:0000:0000:0000:0000:0000:0016:0
04/06/2020-12:23:00.669485 [**] [1:0:0] ICMP protocolo de control de mensajes de internet DETECTADO
[**] [Classification: (null)] [Priority: 3] {IPv6-ICMP} fe80:0000:0000:0000:aebc:32ff:fea2:c8bd:143
-> ff02:0000:0000:0000:0000:0000:0016:0
04/06/2020-12:24:00.962129 [**] [1:0:0] ICMP protocolo de control de mensajes de internet DETECTADO
[**] [Classification: (null)] [Priority: 3] {IPv6-ICMP} fe80:0000:0000:0000:aebc:32ff:fea2:c8bd:143
-> ff02:0000:0000:0000:0000:0000:0016:0
04/06/2020-12:24:41.102931 [**] [1:0:0] ICMP protocolo de control de mensajes de internet DETECTADO
[**] [Classification: (null)] [Priority: 3] {IPv6-ICMP} fe80:0000:0000:0000:aebc:32ff:fea2:c8bd:143
-> ff02:0000:0000:0000:0000:0000:0016:0
04/06/2020-12:26:18.179773 [**] [1:0:0] ICMP protocolo de control de mensajes de internet DETECTADO
[**] [Classification: (null)] [Priority: 3] {IPv6-ICMP} fe80:0000:0000:0000:0478:b1c6:6f95:3305:143
-> ff02:0000:0000:0000:0000:0000:0016:0
pi@raspberrypi:~$ tail -8 fast.log
04/06/2020-12:30:17.593189 [**] [1:0:0] ICMP protocolo de control de mensajes de internet DETECTADO
[**] [Classification: (null)] [Priority: 3] {IPv6-ICMP} fe80:0000:0000:0000:0478:b1c6:6f95:3305:143
-> ff02:0000:0000:0000:0000:0000:0016:0
04/06/2020-12:35:11.689376 [**] [1:0:0] ICMP protocolo de control de mensajes de internet DETECTADO
[**] [Classification: (null)] [Priority: 3] {IPv6-ICMP} fe80:0000:0000:0000:0478:b1c6:6f95:3305:143
-> ff02:0000:0000:0000:0000:0000:0016:0
pi@raspberrypi:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=54 time=7.25 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=54 time=6.74 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=54 time=6.73 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=54 time=7.02 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=54 time=7.35 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=54 time=6.88 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=54 time=22.5 ms
64 bytes from 8.8.8.8: icmp_seq=8 ttl=54 time=7.90 ms
64 bytes from 8.8.8.8: icmp_seq=9 ttl=54 time=7.23 ms
64 bytes from 8.8.8.8: icmp_seq=10 ttl=54 time=6.84 ms
^X64 bytes from 8.8.8.8: icmp_seq=11 ttl=54 time=6.83 ms
^C
--- 8.8.8.8 ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 27ms
rtt min/avg/max/mdev = 6.727/8.482/22.547/4.461 ms
pi@raspberrypi:~$ ping 192.168.1.39
PING 192.168.1.39 (192.168.1.39) 56(84) bytes of data.
64 bytes from 192.168.1.39: icmp_seq=1 ttl=64 time=1264 ms
64 bytes from 192.168.1.39: icmp_seq=2 ttl=64 time=248 ms
64 bytes from 192.168.1.39: icmp_seq=3 ttl=64 time=979 ms
64 bytes from 192.168.1.39: icmp_seq=4 ttl=64 time=5.70 ms
64 bytes from 192.168.1.39: icmp_seq=5 ttl=64 time=6.19 ms
64 bytes from 192.168.1.39: icmp_seq=6 ttl=64 time=958 ms
64 bytes from 192.168.1.39: icmp_seq=7 ttl=64 time=5.88 ms
^C
--- 192.168.1.39 ping statistics ---
8 packets transmitted, 7 received, 12.5% packet loss, time 31ms
rtt min/avg/max/mdev = 5.701/495.456/1264.433/509.830 ms, pipe
2
pi@raspberrypi:~$
```