Universidad
de La Laguna

Escuela Superior de
Ingeniería y Tecnología

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Sistema de control de accesos seguro

Jairo González Lemus

Dña. Pino Caballero Gil, con N.I.F. 45534310-Z Catedrática de Universidad adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutora.

Dña. Alexandra Rivero García, con N.I.F. 78646309-V Contratada FPI de Universidad adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como cotutora.

C E R T I F I C A N

Que la presente memoria titulada:

"Sistema de control de accesos seguro"

ha sido realizada bajo su dirección por **D. Jairo González Lemus**, con N.I.F. 78616776-Q.

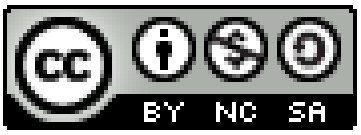
Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 10 de Septiembre de 2019.

Agradecimientos

Después de este largo periodo realizando este proyecto, escribo este apartado de agradecimiento para finalizar mi trabajo fin de grado, lo que significa para mí de cierta forma el final de una etapa de mi vida. Me gustaría agradecer a todas las personas cercanas, como amigos y familiares, que me han motivado a seguir cuando las cosas no salían como se esperaba. Al final, este proyecto no sería posible sin un conocimiento previo de años anteriores en la universidad, no podría hacer mención a un profesor en particular, por eso agradezco de manera generalizada a todos los profesores de la Universidad de La Laguna que me han enseñado algo, y además doy las gracias a mis compañeros de clase por tambien ayudarme a adquirir ciertos conocimientos. Por último agradezco la ayuda prestada por parte de mi tutora y cotutora.

Licencia

Se quiere permitir que se comparta las adaptaciones de la obra mientras se comparta de la misma manera y NO se quiere permitir el usos comercial de la obra.



- **Reconocimiento** — Debe reconocer adecuadamente la autoría, proporcionando un enlace a la licencia e indicar si se han realizado cambios. Puede hacerlo de cualquier manera razonable, pero no de una manera que sugiera que tiene el apoyo del licenciador o lo recibe por el uso que hace.
- **NoComercial** — No puede utilizar el material para una finalidad comercial.
- **CompartirIgual** — Si remezcla, transforma o crea a partir del material, deberá difundir sus contribuciones bajo la misma licencia que el original.

Resumen

El objetivo de este Trabajo de Fin de Grado (TFG) ha sido la implementación de un sistema de control de acceso seguro mediante el uso de dispositivos integrados (una placa Raspberry-Pi, un lector de huellas dactilares y una cámara digital). En el diseño para la gestión y control de acceso a sitios físicos es habitual el uso de aparatos de doble identificación donde intervienen los dispositivos antes mencionados. A lo largo del presente documento se describen los elementos que lo conforman, además, se detallan los métodos y metodologías utilizadas para su elaboración. Entre las principales características desarrolladas podemos destacar: la detección y verificación de la identidad de un usuario mediante el uso del lector de huellas y reconocimiento facial, el registro de la hora y día en la que el usuario es identificado y la visualización de gráficas y tablas con los datos almacenados. Para la gestión de los datos se ha creado una aplicación web que está conectada a una base de datos centralizada que contiene: los datos del registros de control de acceso, los datos de los usuarios y los datos de los aparatos del sistema. La web desarrollada tiene un formato adaptable, es decir, que es visible desde cualquier dispositivo (teléfonos móviles, tablets, ordenadores, etc.) sin importar las dimensiones de la pantalla. Para aportar una mayor seguridad a la hora de la transmisión y guardado de datos, se utiliza el algoritmo criptográfico AES, además de políticas de seguridad, como por ejemplo: solo el usuario root y los administradores, previa identificación, pueden ver los datos. También puede ser instalado tanto en una red doméstica local como en una red empresarial debido a que hace uso de servicios distribuidos, lo que permite una comunicación entre los diferentes dispositivos sin tener que estar todo conectado en la misma sala. Por último, se justifica la elección de los sensores para el presente TFG con la idea principal de reducir los gastos asociados a la tecnología, pretendiendo ofrecer una solución más económica de igual e incluso de más calidad que la que ofrecen actualmente otros sistemas.

Palabras clave: *control de acceso seguro, tecnología, sistema, detección, verificación, identificación.*

Abstract

The objective of this final year project (FYP) has been the implementation of a secure access control system through the use of integrated devices (a Raspberry-Pi board, a fingerprint reader and a digital camera). In the design for the management and control of access to physical sites, it is usual to use double identification devices in which the aforementioned devices are involved. Throughout this document, the elements that make it up are described, as well as the methods and methodologies used to carry it out. Among the main features developed we can highlight: detection and verification of a user's identity through the use of the fingerprint reader and facial recognition, recording of the time and day in which the user is identified and the visualization of graphs and tables with the stored data. For data management, a web application has been created that is connected to a centralized database that contains: the data of the access control registers, the data of the users and the data of the devices of the system. The website developed has a format that responsive, that is, it is visible from any device (mobile phones, tablets, computers, etc.) regardless of the dimensions of the screen. To obtain greater security when transmitting and storing data, the AES cryptographic algorithm is used, in addition to the security policies, such as: only the root user and the administrators, after identification, can see the system data. It can also be installed both in a local home network and in a business network because it has a distributed services service, which allows communication between the different devices without having to be all connected in the same room. Finally, the choice of sensors for the present FYP is justified with the main idea of reducing the expenses associated with the technology, pretending to offer a more economical and quality solution than the current systems.

Keywords: *secure access control, technology, system, detection, verification, identification.*

Índice general

Índice de figuras

Índice de tablas

1. Capítulo 1. Introducción

En este proyecto se ha realizado una implementación de un sistema de control de acceso seguro con doble autenticación. En este capítulo se describen todos los recursos utilizados, tanto hardware como software para construir el sistema, es decir, los dispositivos que se necesitan, los lenguajes utilizados y las herramientas de desarrollo para la construcción.

1.1. Hardware del sistema

1.1.1. Raspberry Pi 3

La Raspberry Pi es un pequeño ordenador de placa reducida de bajo coste, creado por la Fundación Raspberry Pi en Reino Unido. Su lanzamiento al mercado fue el 29 de febrero del 2012, a partir de ahí se han lanzado varios modelos como pueden ser: Raspberry Pi Modelo B, Raspberry Pi Zero, Raspberry Pi 3 Model B (ver Figura 1.1). Con el tiempo se han desarrollado una amplia gama de complementos como pueden ser los sensores que se explican en el punto 1.1.2 y el punto 1.1.3. Su creación fue motivada con el objetivo de animar alumnos en las escuelas de enseñanza de ciencias de la computación a crear proyectos electrónicos [1]. En este proyecto se utiliza para conectar sensores y alojar el software necesario para el funcionamiento del sistema.

Figura 1.1: Placa Raspberry Pi 3.

1.1.2. Lector de huellas

Un lector de huellas o en inglés “fingerprint” [2] es un dispositivo que tiene como misión principal el reconocimiento de huellas dactilares, mediante la lectura y el almacenamiento de las mismas. Antes de la existencia de estos dispositivos, ya se sabía del carácter único de las huella y se utilizaban para cosas como:

- En Babilonia y Persia para autenticar registro en arcilla.
- En 1958 el inglés Sir Williams Herschel [3], principal magistrado del distrito Jungiporr, en la India y un hombre de negocios local firmaron un contrato con la impresión de la mano por parte de este último detrás del contrato.

La necesidad de ciertos sectores por la identificación de personas dio lugar a pensar métodos para lograrlo. En 1883 la primera técnica para resolver este problema la ideó el francés Alphonse Bertillon [4] y consistía en medir diversas partes del cuerpo. Este método fue adoptada por la policía de Francia, pero resultó un fracaso. Por otra parte un estudio realizado por el inglés Francis Galton [5], quien publicó los resultado en su libro “Fingerprints” en 1882, confirmó el carácter invariable de las huellas dactilares. Galton propuso unos cuarenta rasgos para la clasificación de las huellas. Seguidamente Juan Vucetich [6] investigador de la provincia Buenos Aires, después de un análisis de lo que publicó Galton, propuso ciento un rasgos. De esta forma sentó las bases de una identificación personal confiable. A partir de ahí el poder judicial de todos los países empezaron a tomar este método de identificación. Con el paso del tiempo y la llegada de los chips, ordenadores e impresoras este procedimiento que se hacía a mano, se empezó a realizar de otras formas llegando a la invención de este tipo de dispositivos.

En el dispositivo utilizado para la realización de este trabajo de fin de carrera se pueden almacenar unas 1000 huellas, debido a que tiene un buffer de imagen de 72 KB y dos buffers de archivo de 515 bytes. Los usuario pueden realizar cualquier operación mediante instrucciones de memoria. Este dispositivo tiene un gasto de unos 5W, además posee un un led que funciona para señalizar cuando el dispositivo se pone en modo lectura de huella. Este sensor se conecta a otros dispositivos como la Raspberry Pi a través de unas conexiones en serie, cuya codificación es la siguiente (ver Figura 1.2):

- GND: Cable negro , toma de tierra.
- TX: Cable de color azul, envía datos l a ordenador u otro dispositivo.
- RX: Cable verde, toma datos del ordenador u otro dispositivo.
- DNC: Son los cables blancos, pines que por el momento no tienen

función, pero en futuras implementaciones pueden ser utilizados para otros fines.

- VCC: Cable rojo, es el encargado de la alimentación del dispositivo con 5V y 3.3V.

El lector de huellas utilizado en este sistema (ver Figura 1.2) fue creado por un fabricante ajeno a raspberry Pi. Este aparato en concreto se fabrica para conectarse a otros dispositivo de control como por ejemplo: placas arduino, placas raspberry, ordenadores personales, etc. Para ponerlo en funcionamiento se conecta a un conversor de Serie a USB y de ahí mediante cable USB mini macho tipo B a USB de la raspberry-py. Este sensor servirá para el almacenamiento y la identificación de huella digital de los usuarios del sistema.

Figura 1.2: Sensor Dactilar para Raspberry Pi 3.

1.1.3. Cámara digital

La cámara es un dispositivo que su principal funcionamiento es capturar imágenes. En un principio la captura de imágenes se realizaba en un papel especial. Con el paso del tiempo se fueron mejorando las técnicas de captura de imagen hasta la aparición de la primera cámara digital, la cámara Cyclops 1975 creada por Cromemco (compañía de microcomputadoras). En 1988 se crearon los primeros estándares para la imagen digital que son el JPEG y MPEG haciendo posible, de esta forma, la compresión de los archivos de tipo imagen y vídeo para su almacenamiento. Estos dispositivo siguieron evolucionando y llegó en 1995 la primera cámara comercial en utilizar una pantalla de litio que fue la Casio QV-10. En 1996 aparece la CompactFlash de Kodak que logra utilizar tarjetas de memoria para el almacenamiento de las imágenes. En 1997 surge la primera cámara fotográfica con un megapixel para consumidores. Tras el transcurso del tiempo y la evolución de la tecnología este tipo de dispositivo hoy en día han

alcanzado los 32000 megapixeles.

El dispositivo que se ha elegido para el sistema es una cámara de luz visible e infrarroja basada en el sensor Son IM-219 de 8MP (ver Figura 1.3.). Con ella se pueden sacar videos y fotografías, además permite la realización de acciones más complejas como time-lapse y cámara lenta. Permite la creación de diferentes efectos y es compatible con los modos de video 1080p30, 720p60 y VGA-90. La cámara digital para este sistema [7] es creada por los fabricante raspberry Pi. Este dispositivo se conecta al puerto CSI de la Raspberry Pi y funciona en todo los modelos raspberry, además, incluye la biblioteca Picamera escrita en Python para utilizarla. Ha sido elegida debido a que se utiliza bastante en diversos sistemas de seguridad en donde se capturan las imagen para la realización de reconocimientos faciales de los usuarios del sistema.

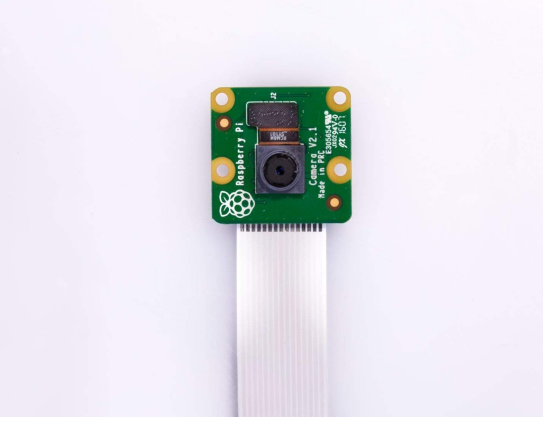


Figura 1.3: Cámara para Raspberry Pi 3.

1.2. Software del sistema

1.2.1. Sistema operativo

El sistema operativo es el encargado de facilitar la interacción entre el usuario, los dispositivos y los demás programas. El sistema seleccionado ha sido sistema operativo Raspbian [8] que es una distribución especial para placas raspberry del sistemas operativo GNU/Linux. Actualmente existen dos versiones compatibles con las Raspberry Pis: Raspbian Pixel (versión completa con entorno gráfico) y Raspbian Lite (versión reducida sin entorno de gráfico, solo modo consola, la cual está específicamente diseñada para la enseñanza). Tiene un modelo de desarrollo de software libre y de código abierto el cual fue creado por Mike Thompson y Peter Green como proyecto independiente. La primera versión de este sistema operativo fue lanzada en el año 2012. Se puede descargar desde la página de dos forma: mediante un asistente (Noobs) o mediante una imagen del sistema operativo. Las razones de haber seleccionado el sistema operativo Raspbian son: el fabricante de Raspberry Pi lo considera estable, está optimizado para

Raspberry, posee una fácil instalación, es software libre y ocupa poco espacio en la memoria principal.

1.2.2. Python

Python es un lenguaje de programación interpretado y multiparadigma, su última versión hasta la fecha es la es Python 3.7.0 [9]. Fue creado por Guido van Rossum y actualmente está siendo mantenido por Python Software Foundation (PSF) que es una organización sin fines de lucro creada el 6 marzo del 2001, esta fundación tienen como propósito fomentar el desarrollo de la comunidad que utiliza Python. La primera versión fue publicada a finales de los ochenta. En el sistema operativo elegido ya viene una versión preinstalada. En caso de querer la versión más actual se ejecuta la siguiente línea de comando **"sudo apt-get install python(Versión)"** por el terminal. Se ha escogido este lenguaje de programación porque posee una licencia de código abierto, denominada Python Software Foundation License, que es compatible con la Licencia pública general de GNU a partir de la versión 2.1.1, además, tiene un amplia gama de librerías y una comunidad muy grande. En este proyecto se utiliza para: registrar huellas en el dispositivo, reconocimiento dactilar, reconocimiento facial y envío de datos a la base de datos.

1.2.3. OpencV

OpenCV [10] es una librería libre de visión artificial y está escrita en C y C++ publicada bajo licencia BSD (Berkeley Software Distribution). Se ha desarrollado de forma multiplataforma para que pueda ser utilizado en cualquier sistema operativo, se podría decir que es un conjunto de implementaciones funcionales que cubren una serie de áreas en el proceso de visión, entre ellas reconocimiento facial. Tiene más de 2500 algoritmos optimizados. Ha sido desarrollada por Intel en junio del año 2000. Se instala en el sistema operativo con el comando siguiente **"sudo apt-get install python-opencv"**. La principal razón para seleccionarla es que es una de las herramientas más utilizadas para proyectos similares en donde se requiere el reconocimiento de objetos.

1.2.4. Node.js

Node.js [11] es una librería de Javascript, muy utilizada para el desarrollo de aplicaciones web. Es de utilización libre y además tienes una gran comunidad muy participativa que colabora activamente para facilitar su utilización. Express es otra librería de node.js que nos ayuda a crear una estructura para desarrollar aplicaciones web. Fue escrito originalmente por Ryan Lienhart Dahl en 2009 y sus posteriores versiones están apadrinadas por la empresa Joyent que contrató a su creador. Para poder hacer que el servidor funcione tenemos que tener instalado en el sistema operativo node y

npm (desde la versión 0.6.3 de Node.js instala automáticamente npm) [12]. Npm [13] es un gestor de paquetes de Node.js que fue creado en 2010 y está escrito en javaScript y desarrollado por Isaac Z. Schlueter. Cabe destacar que está compuesto por: la interfaz de línea de comandos (CLI) y el registro. Con npm y en su sitio web [14] podemos realizar las siguientes cosas: adaptar los paquetes de código a sus aplicaciones, bajar los paquetes necesarios con sus dependencias, ejecutar paquetes sin descargar, compartir código con cualquier usuario de la página, restringir el código a los desarrolladores específicos, formar orgs (organizaciones) para coordinar el mantenimiento del paquete, la codificación y los desarrolladores, formar equipos virtuales mediante orgs, administrar varias versiones, actualizar las aplicaciones fácilmente cuando se actualice el código subyacente, descubrir múltiples maneras para resolver un mismo problema, entre otras cosas.Por último decir que se utiliza en el presente trabajo de fin de grado para crear el servidor web con el propósito de llevar la gestión de la página de la app web del sistema.

1.2.5. Firebase

Es una plataforma que se encuentra en la nube la cual presenta diferentes herramientas para ayudar en el desarrollo de aplicaciones web y móvil, así como para facilitar la conexión de nuestras aplicaciones con una base de datos en tiempo real, de esta forma podemos almacenarlos y tenerlos siempre sincronizados [15]. En el 2016 presentó nuevas funcionalidades como poder hacer autentificación, nuevas formas de almacenamiento, alojamiento, notificaciones, mensajería en la nube. Además se integra con los sistemas javascript. Con esta plataforma se pretende ganar dinero por tener tus datos almacenados y que no se pierdan, ademásse pueden utilizar casi todas sus funcionalidades de manera limitada y de forma gratis, pudiendo ser interesante para aplicaciones de pruebas con poco volumen de datos. Fue desarrollada por James Tamplin y Andrew Lee en 2011 y adquirida por Google en 2014 [16]. Para implementar la base de datos hay que registrarse en firebase, crear un nuevo proyecto y utilizar el código necesario, depende del lenguaje de programación, para realizar la conexión Se emplea esta plataforma para centralizar y facilitar el acceso a los datos, además, la base de datos en la nube simplifica código en el lado del servidor y hace que al desarrollador se despreocupe del mantenimiento.

1.2.6. Bootstrap

Bootstrap es una librería de código abierto creado para la mejora del diseño de sitios y aplicaciones web. En su origen se llamada blueprint. Fue desarrollada por Twitter concretamente por dos de sus desarrollador Mark

Otto y Jacob Thornton. La fecha de lanzamiento fue la 19 de agosto de 2011 [17]. Se incorpora al TFG para que la app web desarrollada pueda verse correctamente en cualquier dispositivo.

1.2.7. Google Charts

Google Charts [18] es una librería de javascript que sirve para construir gráficos para representar datos. se puede utilizar en cualquier proyecto y es de libre uso. Fue creada por Google en el 2007. Hay muchas más librerías que se utilizan para hacer gráficas pero la razón principal de apoyarnos en ella es que es una de las más utilizadas .

1.3. Herramienta de desarrollo para el sistema

1.3.1. Github

Plataforma para el alojamiento de proyectos informáticos [20] que sirve para llevar el control de versiones así como para permitir el desarrollo colaborativo tanto para empresas como para el desarrollo libre. Se utiliza para la gestión de software y es una forma de llevar un registro de quién hizo cada cosa en un determinado proyecto. Hay una gran comunidad y puedes ayudar a gente de todo el mundo a realizar muchas clases de proyecto de tipo digital. También puede servir para ver ejemplos de código de otras personas. Actualmente es de Microsoft y fue creado por Chris Wanstrath, PJ Hyett y Tom Preston-Werner en el 2008 [21]. Para poder utilizar esta herramienta hay que registrase en la plataforma y tener instalado git en el sistema operativo. Se utiliza en el TFG para llevar un registro de las versiones y para que sea más fácil transportar el código de un dispositivo a otro.

1.3.2. SublimeText

Editor de texto que se puede ir configurando y transformando gracias a los plugin. No es un programa de código abierto pero la versión de prueba es de libre utilización. Además de utilizarse con python se puede utilizar para c, c++ o java. Tiene una versión para windows, linux y os x. Está escrito en C++ y Python. Los usuario crean plugins de código abierto. Fue desarrollado por Jon Skinner y Will Bond en enero del 2008. Para poder utilizar la herramienta hay que descargarla desde la página web [22]. Se ha escogido esta herramienta porque tiene una comunidad de desarrolladores grande y se utilizado para trabajar con python.

2. Capítulo 2. Sistemas de Control de Acceso

En este capítulo se hace un análisis de la evolución de los sistemas de control de acceso. Se describen los más utilizados y se comparan sistemas similares al del TFG, detallando las mejoras con respecto a otros sistemas.

2.1. Antecedentes y estado actual

El ser humano con el paso del tiempo y a medida que ha evolucionado ha querido saber quién estaba en un determinado lugar o controlar de alguna manera quien podría pasar a determinados sitios en una instalación. En sus comienzos, los sistemas de control utilizaban personal humano para realizar esta acción. Debido a la globalización, la poca eficiencia y la inseguridad en determinada situaciones, se han creado sistemas electrónicos para llevar a cabo dicho fin. Esta evolución de la tecnología está llegando a tal punto que los propios datos de los registros de estos sistemas pueden generar información para tomar decisiones en una organización. Cada día, las empresas tienden a confiar más en estos sistemas ya que con ellos, debido a la información que se pueden sacar de estos, las empresas consiguen administrar recurso de una manera más óptima. Algunos de los beneficios de los sistemas de control de acceso en locales y espacios restringidos son los siguientes:

- Control de entrada y salida del personal.
- Mayor seguridad y control del público.
- Ahorro en coste de personal vigilante, debido a que es reemplazado por sistemas mecánicos.
- Disminución en tiempo de registro.
- Mejora en la productividad del personal.
- Controlar el acceso a zonas restringidas.

2.2. Otros sistemas de control de accesos

Las opciones a la hora de crear sistemas de control de acceso hoy en día son muy variadas y para ello se pueden utilizan los siguientes dispositivos:

- ***Lectores biométrico de huella:*** Este tipo de sensores [23] obtienen la información de la huella dactilar de una persona que luego puede ser tratada por otros sistemas.

- ***Lectores de iris:*** Este tipo de sensores [24] obtienen la información del ojo de una persona. Esta información podría ser de utilidad en un tratamiento de datos posterior.
- ***Reconocedores faciales:*** Son sistemas compuestos por una cámara de video y generalmente por un programa que hace la detección facial [25]. El programa analiza las características faciales de la persona que se encuentra delante de la cámara y verifica si tiene permiso de acceso.
- ***Lector de tarjetas chip:*** Aparato que lee las tarjetas chip introduciéndolas por un orificio. El chip guardar información que verifica la identidad del usuario [26].
- ***Lector de Tarjetas de radiofrecuencia (RFID):*** Son similares a los lectores de tarjeta chip, pero en este caso el lector detecta la tarjeta mediante aproximación [27].
- ***Código acceso:*** Es un tipo de dispositivo que contiene un teclado por el que usuario introduce una clave y el aparato verifica que sea correcta [28].
- ***Sistemas híbridos:*** Estos sistemas utilizan para identificar al usuario varios de los dispositivos anteriores [29].

Otras opciones utilizadas actualmente para la identificación de las persona son:

- ***La utilización de un teléfono móvil:*** Se envía a través del móvil una contraseña y el dispositivo verifica su identidad [30].
- ***Reconocedor de patrón de venas:*** Máquina de rayos que reconoce el patrón de venas que presenta la mano [31].

Como vemos, existen muchos dispositivos para configurar un sistema de control de acceso, los cuales vienen en distintos formatos y calidades.

2.3. Mejoras implementadas con respectos a otros sistemas

Para ejemplificar los componentes que hay en el mercado y que hacen algo parecido a lo que desarrollado en este TFG, se han consultado diversas páginas principalmente de ventas de este tipo de producto [32] y en youtube [33]. El conocimiento que aportan esas páginas es que este tipo de dispositivo tiene un elevado precio rondando los 250€ o más y esto sin contar su colocación. Una mejora importante derivada de este TFG es poder reducir el coste de este aparato. Además se pretende implementar de manera software el poder ver los datos de entrada y salidas de manera gráfica de una forma cómoda. Otro aspecto que se mejora es la facilidad de integración de nuevos dispositivos a la hora de colocarlos en otras partes de la empresa.

3. Capítulo 3. Análisis de Casos de Uso del Sistema

En este capítulo se muestra un pequeño análisis de los casos de uso del sistema. Para ello se diferencian 3 tipos de actores en el sistema a los que se nombran root, administrador y usuario. Se toma este tipo de jerarquía de usuarios muy similar al sistemas operativo linux debido al sistema que se pretende conseguir, para tener un cierto nivel de control y de seguridad en el dispositivo. La utilización del sistema por parte de estos actores se ve reflejada en los siguientes diagramas de caso de uso.

3.1. Funcionamiento

3.1.1. Usuario Root

Este usuario es el que tiene acceso a toda la app y el que puede hacer todas las configuraciones existentes. Las operaciones que realiza el usuario root son las siguiente (ver Figura 3.1):

1. Añadir nuevo usuario.
2. Eliminar usuario existente.
3. Añadir usuario administrador.
4. Eliminar usuario administrador.
5. Ver gráficas que reflejan los datos de entrada y salida por parte de los usuario “usuarios”.

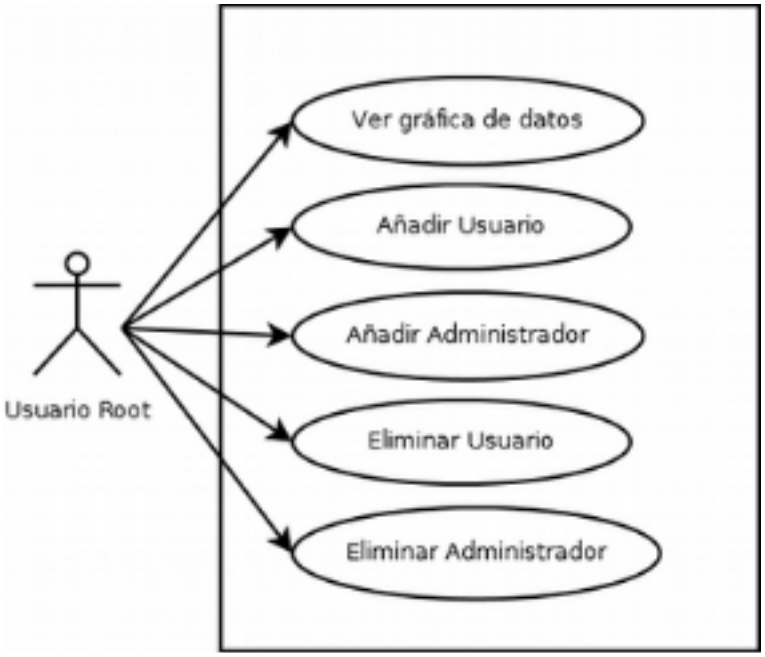


Figura 3.1: Diagrama de caso de uso del usuario root.

3.1.2. Administrador

El usuario administrador tiene limitado el acceso a parte de contenido de web, éste será añadido por el usuario root. Las operaciones que realiza son las siguiente (ver Figura 3.2):

1. Añadir nuevo usuario.
2. Eliminar usuario existente.
3. Ver gráficas refleja los datos de entrada y salida.

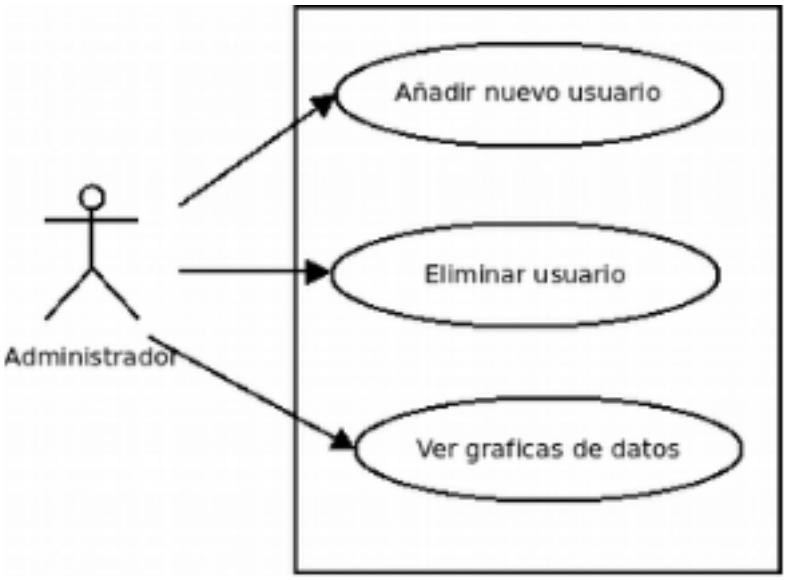


Figura 3.2: Diagrama de caso de uso del usuario administrador.

3.1.3. Usuario

Este usuario “usuario” es introducido por root o por los administradores. Este simplemente interactúa con los sensores del aparato, tanto para que los otros tipos de usuarios introduzcan los datos de este en el sistema como para que el propio aparato lo identifique. Las operaciones que realiza son las siguiente (ver Figura 3.3):

1. Interactuar con la cámara: se realizará cuando la cámara esté activada y el usuario se ponga delante.
2. Interactuar con el sensor de huellas dactilares: cuando el sistema ilumine este sensor, con una luz azul, el usuario posiciona su dedo en la pantalla del sensor.

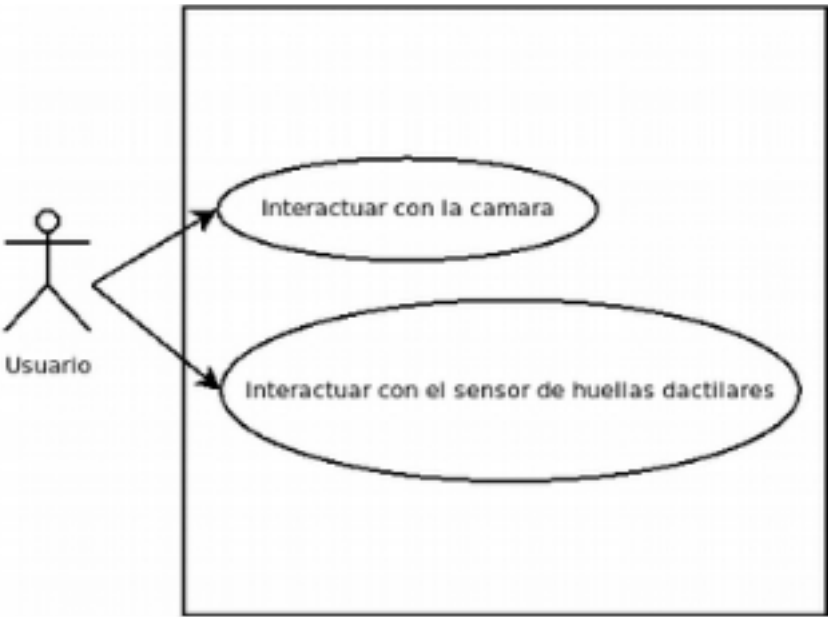


Figura 3.3: Diagrama de caso de uso del usuario.

4. Capítulo 4. Fases y Desarrollo

En este apartado se describe todo lo que tiene que ver con la etapa de desarrollo del sistema, exponiendo cómo se ha realizado cada tarea para lograr una buena implementación del proyecto.

4.1. Servidor web

Para poner en funcionamiento el servidor web, dentro del sistema operativo que utiliza la raspberryPi, se abrirá la shell y se procederá a la instalación de las herramientas necesarias. Para instalar las herramientas se pondrá la siguiente sucesión de comandos:

1. **sudo apt-get install nodejs -y** “Instalar nodejs”.
2. **sudo apt-get install npm** “Instalar npm”.
3. **mkdir paginaWeb** “Creamos un directorio para albergar la aplicación”.
4. **npm install -g express-generator** “Instalar un generador de estructura de Express que crea toda la estructura de directorios básica, con ficheros de demostración. Con el parámetro -g indicamos que sea un comando disponible para todo el sistema en el que lo instalamos”.
5. **express ctrlAccess** “Se le dice al generador que genere la estructura en el directorio ctrlAccess. Así creará una serie de directorios y ficheros básicos para comenzar a trabajar con Express”(ver Figura 4.1).

```
a1u01008l3272:~/workspace/paginaWeb (master) $ express ctrlAccess

  warning: the default view engine will not be jade in future releases
  warning: use `--view=jade' or `--help' for additional options


   create : ctrlAccess/
   create : ctrlAccess/public/
   create : ctrlAccess/public/javascripts/
   create : ctrlAccess/public/images/
   create : ctrlAccess/public/stylesheets/
   create : ctrlAccess/public/stylesheets/style.css
   create : ctrlAccess/routes/
   create : ctrlAccess/routes/index.js
   create : ctrlAccess/routes/users.js
   create : ctrlAccess/views/
   create : ctrlAccess/views/error.jade
   create : ctrlAccess/views/index.jade
   create : ctrlAccess/views/layout.jade
   create : ctrlAccess/app.js
   create : ctrlAccess/package.json
   create : ctrlAccess/bin/
   create : ctrlAccess/bin/www

   change directory:
     $ cd ctrlAccess

   install dependencies:
     $ npm install

   run the app:
     $ DEBUG=ctrlaccess:* npm start
```

Figura 4.1: Estructura creada con el comando express ctrlAccess.

6. **npm init** “Este comando lo ponemos para configurar de nuevo el fichero package.json donde se le indicarán las dependencias y algunas propiedades del proyecto(versiones, nombres , comando,..).

Nota: Con todos los puntos anteriores ya estará configurado el servidor básico pero para completar instalaremos un para de utilidades para desarrollar, que son herramientas para hacer testing. Mocha y Chai. el parámetro --save-dev logramos que, además de instalarlas, queden las dependencias reflejadas en las dependencias de desarrollo.

7. **npm install chai --save-dev** “Instalar chai”.
8. **npm install mocha --save-dev** “Instalar moca”.
9. **mkdir test** “Creamos una carpeta donde realizaremos los test”. Así, cuando escribamos **npm test**, se ejecutará el comando “mocha”, que buscará los test en el directorio “test”. Añadir el comando para hacer test al package.json, justo al lado del comando start en el apartado de scripts. Es decir, quedaría algo así: "scripts": { "start": "node ./bin/www", "test": "mocha" }.
10. Para comenzar a rodar el servidor web si es la primera vez hay que instalar las dependencias que faltasen con el comando **“npm i”** y seguidamente inicializamos el servidor con **“npm start”.**
11. Para poder ver la página en el navegador poner la siguiente dirección http://0.0.0.0:3000/ o http://localhost:3000/

4.2. Página web

Como ya se ha mencionado, el sistema está gestionado por una app web. Por lo que se ha creado y diseñado diversas páginas web. Ya sabiendo lo que hace y lo que quiere que se haga en el sistema, se implementará de manera visual. Como hemos visto en el capítulo 3 los usuarios que utilizan este dispositivo son de tres tipos, por lo que se tendrá que diferenciar la web en tres partes. Se creará entonces de forma visual una parte para el usuario root, otra para los administradores, y por último para los usuarios. En una primera fase se realiza unos dibujos a mano como diseño preliminar para ver las posibles visualizaciones de las páginas. A continuación se ha tomado la decisión de que las páginas tengan un diseño responsivo o en inglés “responsive”, es decir, que se pueda ver en cualquier tipo de dispositivo. Después de un pequeño estudio buscando información por internet [34] nos hemos decantado por utilizar bootstraps, ya que según es unos de los framework más utilizado en el mercado para hacer la web responsiva. Para el diseño de las página, se tiene en cuenta ejemplos diseñados de otras páginas con bootstrap los cuales se han adaptado al proyecto. Se ha

intentado que los usuario les sea fácil reconocer lo que puedan o no hacer. La primera página (ver Figura 4.2), el index, que se presentará al usuario estará compuesta por dos botones. El que pone **“administrator”** llevará a la parte del usuario root y administrador y el otro botón **“get in”** entrara a la parte donde se pondrá en funcionamiento el reconocimiento del usuario.

Figura 4.2: Página index del sistema.

Cada usuario realizará cosas distintas ya analizadas en el capítulo 3 lo que conlleva páginas y menús distinto:

- Para el **usuario root**, antes de ir a su página de menú tendrá una página de login (ver Figura 4.3), después de la verificación con correo y contraseña, que por defecto será **“root@gmail.com”** y la contraseña “**123456**”, accede a la página donde se visualizará un menú en la parte superior en el que se verán las siguiente opciones (ver Figura 4.4):

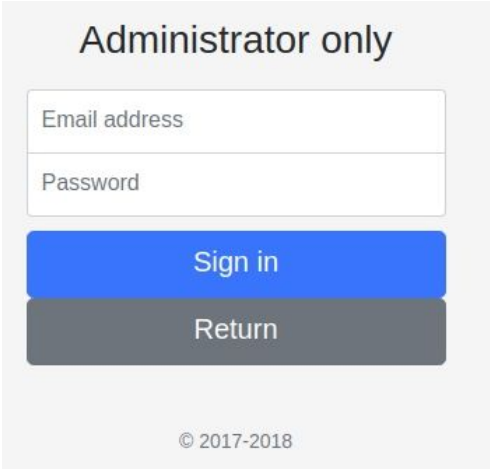


Figura 4.3: Página de login del sistema.

.

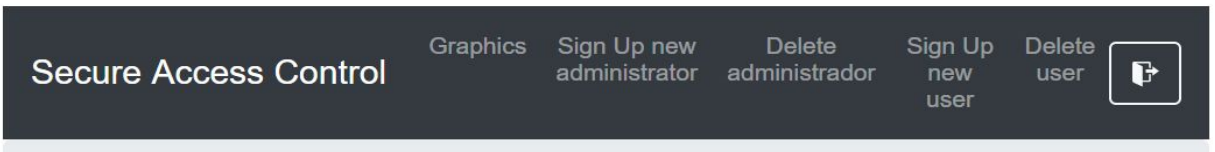


Figura 4.4: Menú del usuario root del sistema.

  - **Añadir nuevo administrador:** Esta pestaña redirecciona a una

página donde se introduce todo los datos necesario para registrar un administrador.
    - **Añadir nuevo usuario:** Esta pestaña redirecciona a una página donde se introduce todo los datos necesario para registrar un usuario.
    - **Eliminar administrador:** Esta pestaña redirecciona a una página donde se pide el identificador de usuario administrador que se desea eliminar. Además, se visualizará una tabla con todos los administradores existente.
    - **Eliminar usuario:** Esta pestaña redirecciona a una página donde se pide el identificador de usuario tipo usuario que se desea eliminar. Además, se visualizará una tabla con todos los usuario de tipo usuario y cierta información de ellos.
    - **Gráficas:** Esta pestaña redirecciona a una página donde se pondrá seleccionar varios tipos de gráficas y tablas, que representarán los datos producido por la identificación de los usuarios realizada por el sistema.
- Para el usuario **administrador** ante de ir a su menú también tendrá que pasar por una página de login (ver Figura 4.3), después de la verificación con correo y contraseña accede a la página donde se visualizará un menú en la parte superior en el que se verán las siguiente opciones (ver Figura 4.5):

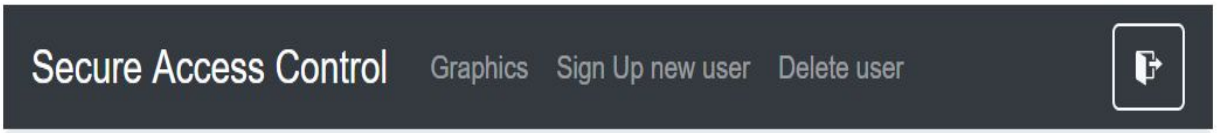


Figura 4.5: Menú del usuario administrador del sistema.

    - **Eliminar usuario:** Esta pestaña redirecciona a una página donde se pide el identificador de usuario tipo usuario que se desea eliminar. Además, se visualizará una tabla con todos los usuario de tipo usuario y cierta información de ellos.
    - **Añadir nuevo usuario:** Esta pestaña redirecciona a una página donde se introduce todo los datos necesario para registrar un usuario.
    - **Gráficas:** Esta pestaña redirecciona a una página donde se pondrá seleccionar varios tipos de gráficas y tablas, que representarán los datos producido por la identificación de los usuarios realizada por el sistema.
- En la página del **usuario tipo usuario** (ver Figura 4.6) muestra lo que visualiza la cámara en tiempo real, el sistema pondrá en funcionamiento el programa de doble identificación. En la propia imagen, que se está visualizando por la página, se observará

información sobre si se identifica o no al usuario que está enfrente de la cámara.

Figura 4.6: Página para activar el proceso de identificación.

4.3. Reconocimiento facial

A continuación se explica cómo se debe configurar el dispositivo y se realiza una breve descripción del funcionamiento del dispositivo reconocedor de rostros y de la librería escrita en python utilizada para que el dispositivo funcione correctamente.

4.3.1. Dispositivo

El dispositivo a utilizar es la cámara pi ya descrita con más detalle en el punto 1.1.3 y donde también se menciona el módulo Picamera de python que utilizaremos para la configuración y la captura de imágenes. Con ella se capturará las imágenes que posteriormente se analizaran para realizar el reconocimiento facial. Después de haber visto cómo trabajar con la cam y el módulo de python Picamera [35], ver los requerimiento del sistema y hacer un breve análisis para la elección de un formato de imagen [36]. Se ha llegado a la siguientes conclusiones para que el sistema sea lo más óptimo: la resolución de imagen que se capturara será 640X480, para que la imagen tenga un efecto espejo se pondrá a true la siguiente variable hflip y para unificar un formato se a escogido .jpg (ver Figura 47.).

```
with picamera.PiCamera(resolution='640x480', framerate=30) as camera:
    output = StreamingOutput(camera=camera)
    camera.hflip=True
    camera.annotate_text = "Not detected"
    camera.annotate_background = picamera.Color('red')
    camera.start_recording(output, format='mjpeg',quality=30)
    try:
       address = ('127.0.0.1', 8000)
       server = StreamingServer(address, StreamingHandler)
       server.serve_forever()
    finally:
          camera.stop_recording()
```

Figura 4.7: Líneas de código de configuración de la cámara.

4.3.2. OpenCV funcionamiento

Esta biblioteca ya descrita en el punto 1.2.3 tiene multitud de funciones para el tratamiento de imágenes. Para este TFG una de las tareas es la identificación facial y para ello se necesita realizar tanto la detección como el reconocimiento del usuario. Después de ver algunas formas de resolver estos problemas y los datos de sus resultados se concluye que no hay una solución 100% efectiva para el 100% de los casos. Además, las mejores soluciones necesitan una gran capacidad de cómputo por lo que en ninguno de los casos y debido a que se trabajara con la raspberry pi no sería lógico incorporarlas ya que tiene un limitado procesamiento. A continuación se enumeran algunos de los algoritmos de detección y de reconocimiento más relevantes que se pueden implementar o implementados por esta biblioteca:

- Face Detection using Haar Cascades. Detección.
- Scale Invariant Feature TRansform(SIFT)(1999). Detección.
- Speed Up Robust Features(SURF)(2006). Detección.
- Eigenface(1991). Reconocimiento.
- Local Binary Patterns Histograms (LBPH)(1996)[37]. Reconocimiento.
- Fisherface(1997). Reconocimiento.

Mencionar que de cada algoritmo hay mejoras implementadas pero estas mejoras provocan un aumento de coste computacional. Para concluir y después de un pequeño análisis y por motivo de limitado procesamiento se decide implementar en el sistema los siguiente algoritmo:

- Face Detection using Haar Cascades. Detección.
- Local Binary Patterns Histograms (LBPH) (1996)[37] Reconocimiento.

4.3.3. Reconocimiento facial

Para la **detección** se ha elegido Face Detection using haarCascades ya que su incorporación es muy fácil y simplemente se necesitan dos

funciones y un archivo de configuración .xml. Esta parte está muy bien documentada en su página [38]. Tras la experiencia de haber trabajado con ella puedo decir que realiza la una detección positiva en la mayoría de los casos pero depende como se configure puede dar mucho falsos positivos en las imágenes. En este caso en concreto se configura el programa para que solo pueda detectar un rostro y ha cierta distancia de la cámara. El sistema de detección, para simplificar, se describe de la siguiente forma: se partirá la imagen en múltiples trocito y cada una pasará por los clasificados mientras no sea descartado, cada clasificado buscar ciertas características y se irán deshaciendo de esos trozos hasta que al final si hay alguna imagen con un rostro la encuentre. Este sistema puede detectar más de un rostro a la vez.

Para la selección del **reconocedor** fue un poco más complejo. Al final después de sopesar los pros y contras, el clasificador seleccionado ha sido LBPHFaceRecognizer que por lo que se puede leer en foros es utilizado en sistemas similares. Para resumir lo que hace el algoritmo LBPH o en español histogramas de patrones binarios locales se describe de la siguiente manera:

1. Para poder utilizar esto la imagen a reconocer tiene que estar en escala de grises. Así los valores de cada píxel estará compuesto por un solo elemento comprendido en ‘0 a 255.
2. Se construirá una nueva imagen a partir de esa en escala de grises:
    a. Se particiona la imagen en una cuadrícula
    b. De cada trocito de la cuadrícula: Se mirara cada pixel haciendo una media de los valores vecinos del pixeles. Ese valor medio pondrá a 1 o 0 el píxel analizado.
    c. Lo anterior se repetirá en cada pixel.
    d. Se generará una imagen binaria de toda la imagen.
    e. Se construirá un histograma por cada trocito de la cuadrícula.
3. Se generará un histograma de histogramas con los valores de los trocitos de las cuadrículas.
4. Se compara el histograma resultante con los de las imágenes almacenadas.
5. De esa comparación se obtendrá un valor que se llamaremos presión, mientras más cercanas al 0 más iguales serán las imágenes comparadas.
6. Si precisión estará por debajo de un valor se dirá que se ha detectado la imagen y se reconocerá a la persona.

Por último, para no empezar desde cero y ver que se hace el reconocimiento correctamente hemos cogido imágenes de rostros que se pueden utilizar para pruebas de reconocimiento de la base de datos [39].

4.3.4. Implementación en el sistema

En este punto se explican con detalle los problemas del cómo, cuándo y dónde se realiza el almacenamiento y el reconocimiento de los rostros.

4.3.4.1. Realizar el almacenamiento del rostro

El usuario que podrá realizar esto es root y también el administrador. Al dirigirse a la página de newUser se pondrá a funcionar el programa que almacena las imágenes. Los pasos para hacer el almacenamiento de imágenes en el sistema son los siguiente:

- Coger la captura de la cámara.
- Pre-Procesado de imagen.
- Detección del rostro en la imagen pre-procesada.
- Recortar y redimensionar el rostro de la imagen procesada.
- Se irán almacenamiento las capturas de imágenes del rostro en una carpeta temporal.
- Apretar botón **“Takes photos”** Seleccionar 20 imágenes y guarda en una nueva ubicación temporal. Se podrá borrarlas, y seleccionar toda las imágenes con los boten **“Delete Select ”**, **“Select All”** respectivamente.
- Si ya estamos conformes con las 20 imágenes se pisa el botón **“Add Image”** eso hará que las imágenes se guarden en una carpeta asociadas al un nombre. Para minimizar errores ese nombre será único en sistema e identificar al usuario a reconocer. En este caso en concreto estará formado por el nombre principal y la parte primera del correo esto hará que no se produzcan errores de duplicidad a la hora de reconocimiento facial.

Mientras más imagen diferente del rostro del individuo se almacenan el reconocimiento será más efectivo. A partir de 3 imágenes el sistema reconocerá al individuo pero en muy pocos casos. En otros proyectos las pruebas se realizan almacenando 10 imágenes del rostro por persona y dan buen resultado. En este caso se toma la decisión de almacenar 20 imágenes faciales del usuario para su posterior reconocimiento.

4.3.4.2. Realizar reconocimiento facial

El reconocimiento empezará a funcionar en la página getIn y se realizará los siguiente pasos:

- Coger la captura de la cámara.
- Pre-Procesado de imagen (convertir imagen gris y se regula el contraste).
- Entrenar el sistema con el algoritmo a utilizar y las imágenes de

la base de datos.
- Detección del rostro en la imagen preprocesada.
- Compara mediante el algoritmo con las imágenes guardadas.
- Implementación en el Sistema.

4.4. Reconocimiento de huellas

A continuación se realiza una breve descripción del funcionamiento del dispositivo reconocedor de huellas y de la librería escrita en python utilizada para que el dispositivo funcione correctamente.

4.4.1. Dispositivo

El dispositivo de reconocimiento dactilar es con el que se realiza la identificación de la huella del usuario. En un primer momento cuando acercamos el dedo a la pantalla este saca una imagen digital. Esta imagen es procesada y después es comparada con todas las que previamente se han introducido, en caso de que exista una coincidencia habrá reconocido la huella. En este caso concreto y ya mencionado en el punto 1.1.2 el aparato tiene almacenamiento necesario para introducir 1000 huellas digitales y dos búferes con el que hace operaciones. Las operación que puede realizar con el dispositivo entre otras son: almacenamiento de la huella de imagen digital, ver el vector resultante del cálculo de la huella, borra la huella que está en una posición dada, comparar dos huellas que tenemos en el buffer, etc. Destacar también y muy importante que todas las opciones anteriores se realizan a través de instrucciones, lo que quiere decir que no tenemos ningún tipo de control de cómo se harán las operaciones, por ejemplo: el algoritmo para sacar el vector característico de la huella, cantidad de datos a leer, qué rasgos reconocer de la huella, como interpretar la imagen de la huella, la forma como borramos los datos, como leer las imágenes, etc. En resumen, el dispositivo es como una caja negra al que se la hace peticiones y devuelve un resultado o se realizan funciones dentro de él.

4.4.2. Reconocimiento dactilar

El sistema de reconocimiento dactilar [40] está basado en el procesamiento de una imagen digital. Debido a que cada fabricante implementa su propia forma de reconocer la huella y no la detallan por motivos de seguridad, se explica de forma superficial cómo se realiza el reconocimiento. Previamente, con el sensor, es tomada una imagen del dedo de la persona a identificar y almacenada en el. Para la identificación se pondrá el dedo antes almacenado en el sensor y el propio dispositivo pasará el algoritmos de filtrado propio, en este paso se detecta ciertas peculiaridad de la huella como por ejemplo los valles y las crestas (ver Figura 4.8) entre

otras, después de esto, se pasa un algoritmo que calcula la distancia que hay entre ellas y se genera un vector de información de la huella. Cada vez que el dispositivo saca una imagen o hacemos que saque una imagen se genera un vector característico. Como es muy poco probables que dos imágenes sean iguales o generen un vector de huella igual se tendrá en cuenta un dato más en la comparación, la variable precisión. Esta variable lo que dice que tan iguales son los dos vectores característicos a comparar, si alcanza cierto valor dará como reconocida la huella.

Figura 4.8:Imagen de Características de la huella dactilar. Extraído de [40]

4.4.3. Módulo Pyfingerprint

Ya visto como se hace de manera teórica el reconocimiento dactilar hay que implementar esa funcionalidad y para ello se utiliza de la librería de python pyfingerprint [41]. Este módulo está bien documenta y es fácil emplearlo. Se instala en linux utilizando el siguiente comando: **pip install pyfingerprint**. La primera parte que se realiza siempre es inicializar el sensor y verificar que todo está correctamente funcionando (ver Figura 4.9.):

```
try:
    f = PyFingerprint('/dev/ttyUSB0', 57600, 0xFFFFFFFF, 0x00000000)

    if ( f.verifyPassword() == False ):
        raise ValueError('The given fingerprint sensor password is wrong!')
except Exception as e:
    print('The fingerprint sensor could not be initialized!')
    print('Exception message: ' + str(e))
    exit(1)
```

Figura 4.9: Código de inicialización utilizando pytingerprint.

Seguidamente si queremos hacer una lectura del dedos se realiza con el siguiente código (ver Figura 4.10).

```
## Wait that finger is read
while ( f.readImage() == False ):
    pass
```

Figura 4.10: Código esperando lectura de huella con pytingerprint.

Además, se puede realizar: una búsqueda de las imágenes que estén almacenadas, almacenar una nueva imagen, descargar una imagen por fuera del dispositivo, comparar dos huellas, etc. Cada cosa que se puede hacer con el dispositivo tienes una función.

4.4.4. Implementación en el sistema

En este punto se explican los problemas del cómo, cuándo y dónde se realiza el almacenamiento y el reconocimiento de la huella dactilar.

4.4.4.1. Realizar el almacenamiento dactilar

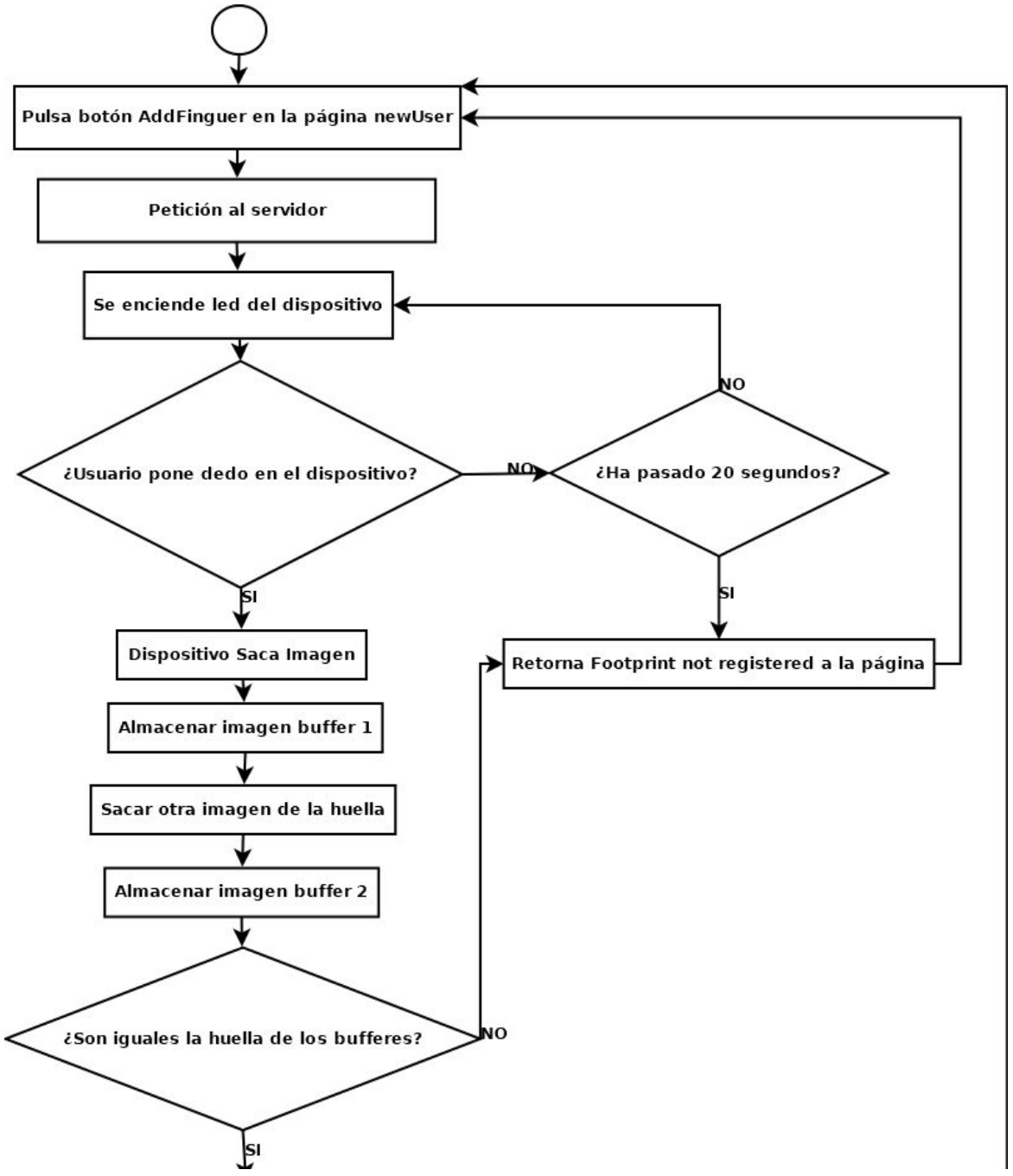


Figura 4.11: Parte 1 Diagrama de flujo del almacenamiento de la huella dactilar.

Tanto el usuario root como el administrador podrá realizar esta tarea. Los dos podrán introducir la huella dactilar del nuevo usuario cuando lo registren. Esta operación se realizará en el formulario de nuevo usuario en la parte donde hay un botón **"Add Footprint"** que se pulsara y hará que se ponga en funcionamiento el programa para almacenar la huella. Al pisar el botón se accionara un evento el cual se recogerá con javascript y hará una petición POST al servidor, en el lado del servidor cuando recibe esta petición lo que hará es llamar al código escrito en python que hace la inserción dentro del dispositivo de la huella dactilar (ver Figura 4.11). Para finalizar el proceso (ver Figura 4.12), se sacará el vector característico de la huella almacenada, ya que este dato es muy importante le daremos algo de seguridad codificando y para esto utilizando el algoritmo AES. Para realizar lo anterior lo implementaremos con el módulo de python Crypto [42] ese cálculo dará como resultado un string el cual se ingresará en la base de datos y se retorna a la página el mensaje de All right.

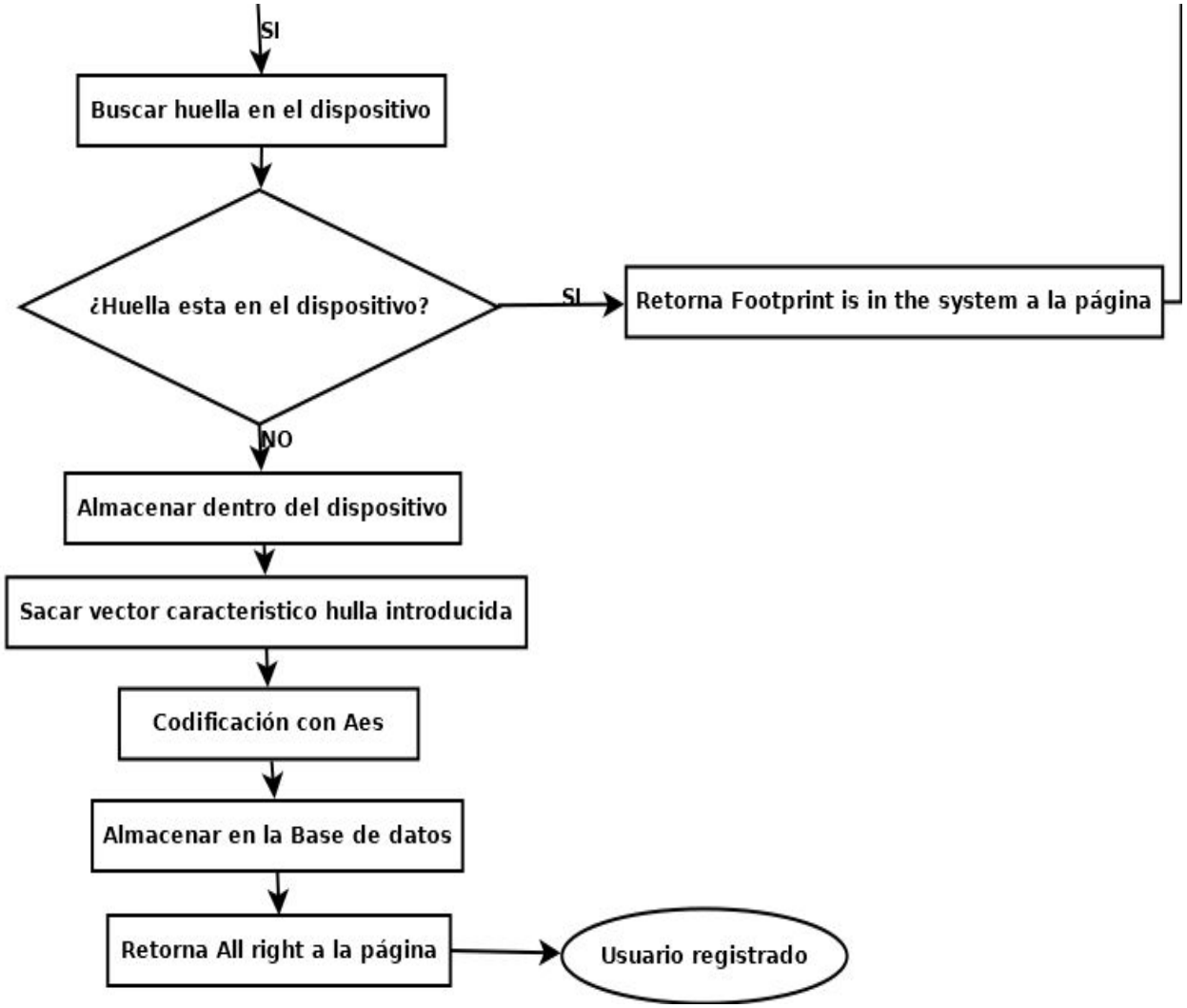


Figura 4.12: Parte 2 Diagrama de flujo del almacenamiento de la huella dactilar.

4.4.4.2. Realizar reconocimiento dactilar

Para que el sistema realice esta función y sobre todo para que se haga la doble identificación se tendrá que previamente haber reconocido fácilmente al usuario. Al final del reconocimiento facial se obtendrá un nombre. Seguidamente se buscará en la base de datos los vectores de huellas que corresponden con ese nombre y la máquina en la que esté el usuario registrado (ver Figura 4.13).

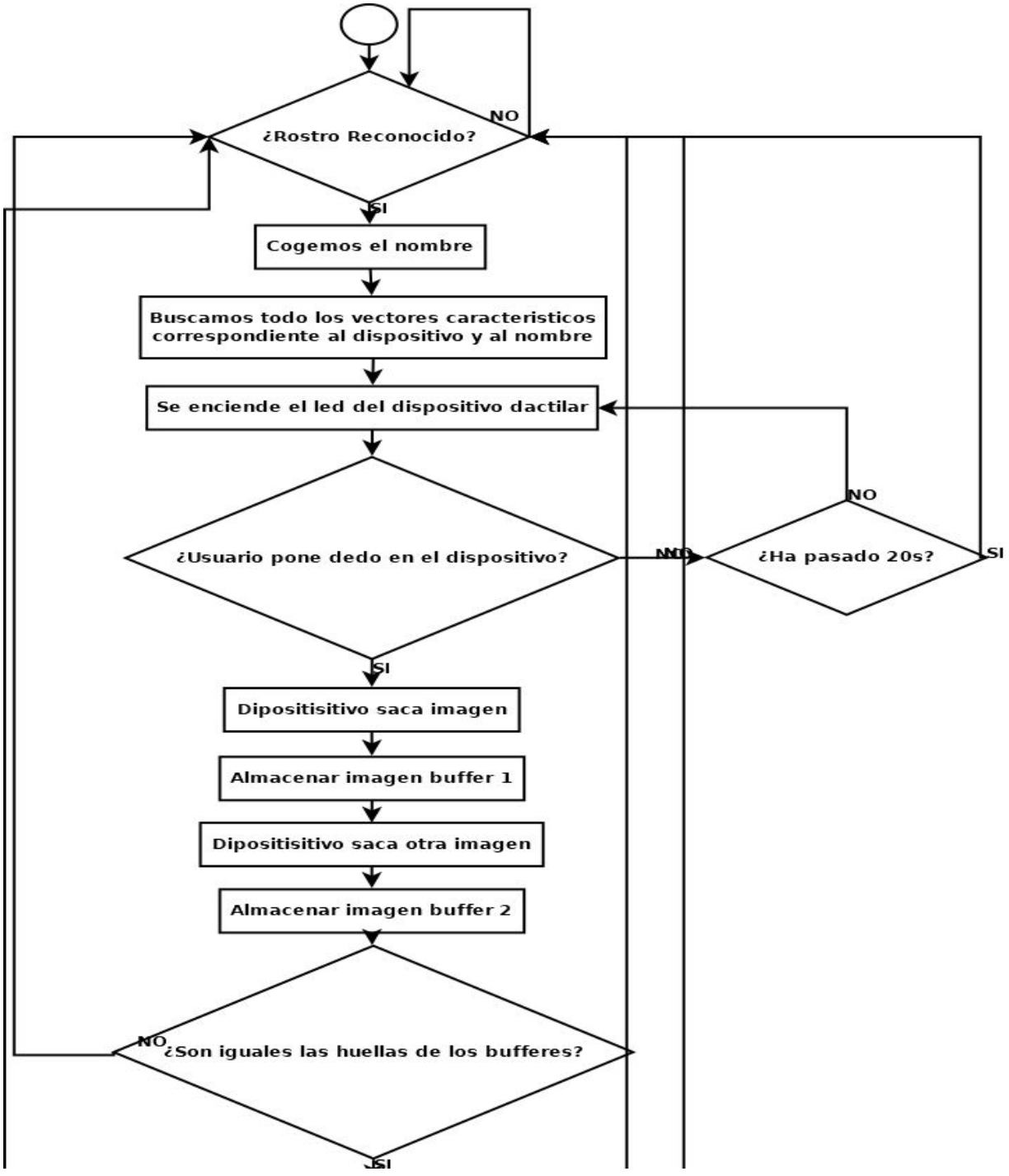


Figura 4.13: Parte 1 Diagrama de flujo reconocimiento dactilar

Después de verificar que la huella esté dentro del dispositivo dactilar (ver Figura 4.14) se verificará la correspondencia con alguna de los vectores de huellas bajado de la base de datos, en el caso de que coincidan se dará validada la doble identificación y se registrara al usuario como una entra en el registro. Para poder comparar estos vector previamente se tendría que desencriptar con AES.

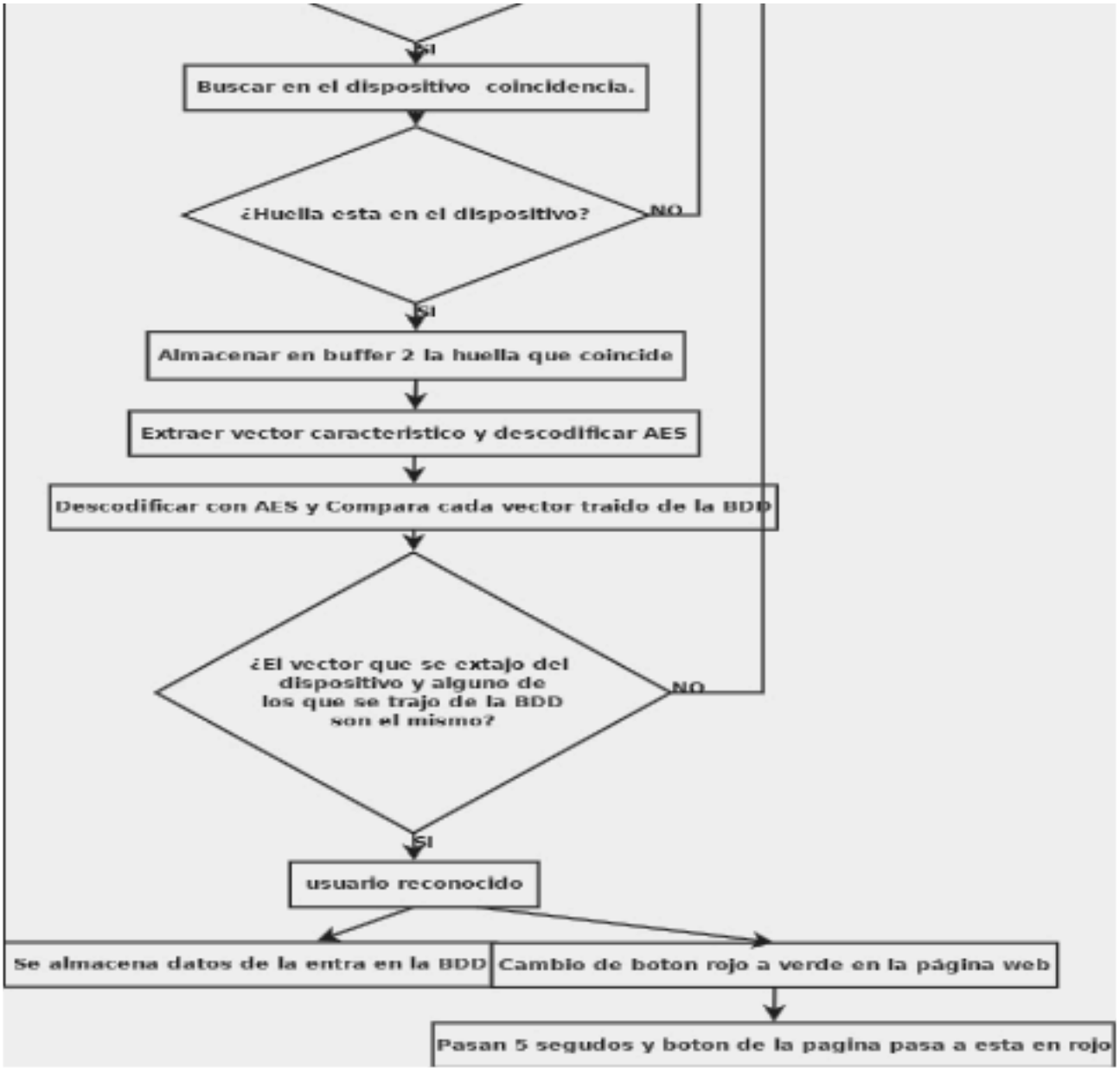


Figura 4.14: Parte 2 Diagrama de flujo reconocimiento dactilar

4.5. Implementación de la doble identificación

En este punto se hará una descripción de cómo se ha realizado la doble identificación. Definimos así lo que nuestro sistema es la doble identificación: Relación que hay entre la detección facial y dactilar, es decir el

usuario que se detecte fácilmente tendrá que tener una huella asociada. Para que exista esa relación tiene que estar expresada en el resultado que darán al final de los dos tipos de sistema de reconocimiento. Tanto el reconocimiento facial como el dactilar tendrá que dar valores únicos para cada usuario y esos valores hacerlos corresponder con el usuario a identificar. El valor único de correspondencia en el reconocimiento facial será el nombre del reconocimiento del usuario que será un dato único en el sistema. Por otro parte la identificación dactilar dará como resultado un vector característico único que está almacenado en el dispositivo dactilar. Los datos resultante tendrán que estar previamente almacenados con el nombre único de usuario en la bases de datos. Al final se logrará relacionar entre nombre del reconocimiento del rostro, el vector almacenado en el dispositivo y el nombre del usuario correspondiente. Para que la doble identificación sea posible primero tienes que existir una identificación fácil, después una identificación dactilar y seguidamente ver si hay una relación entre las dos identificaciones. Si todo es positivo se ha identificado al usuario en el sistema.

4.6. Base de datos

En este apartado se intenta explicar el desarrollo de la base de datos (a partir de ahora BDD) para sistema a implementar. Aquí se intenta plasmar todo lo que tenga que ver con la BDD.

4.6.1. Análisis y desarrollo

Seguramente cada empresa tendrá sus motivó para guardar ciertos datos en esta base de datos. Para simplificar, sólo se registran los atributos que se considera básico para el funcionamiento de sistema. También se tiene en cuenta las características de los sensores y de la plataforma de la base de datos. Para implementar todo esto y para centralizar los datos se elige firebase la plataforma descrita en el punto 1.2.8.

Por motivo de facilitar la implementación, seguridad y aprovechar uno de los servicios de la plataforma se realizar el inicio de sesión de los usuario administradores y root con la opción de método correo/contraseña (ver Figura 4.15). Activamos este método, a partir de ahí, se genera una BDD automática y estará localizada en la parte de firebase authentication.

Figura 4.15: Página authentication de ferabase.

Se ha separado la base de datos en tres colecciones a las que están seguidamente descritas:

1. **usuario:** En este colección (ver Figura 4.16) se almacenan todo los datos referido al usuario protagonista y los datos para hacer la doble identificación. Estos documentos tienen los siguientes campos:
    a. Campos de datos personales: emailId, firstName,lastName, otherInfo, worPosition.
    b. Campo identificador facial: nameFile.
    c. Campo identificar la huella: Este campo m_div que será un grupo de documento que tendrá un indices con el número mac del dispositivo del que se hace el registro y el vector característicos almacenado en el dispositivo dactilar.

Figura 4.16: Ejemplo de documento usuario.

2. **passVerification:** Cada usuario de tipo usuario cada vez que sea identificado generará un documento que subirá a la colección passVerification (ver Figura 4.17). Se introduce un documento con estos datos el firstName, emailId, mac de destino, y el día y

la hora de la identificado expresados en distinto formatos.

Figura 4.17: Ejemplo de documento passVerification.

3. **divece**: este documento solo tendrá el número mac del dispositivo que se registra en el sistema y un name para tener de referencia (ver Figura 4.18).

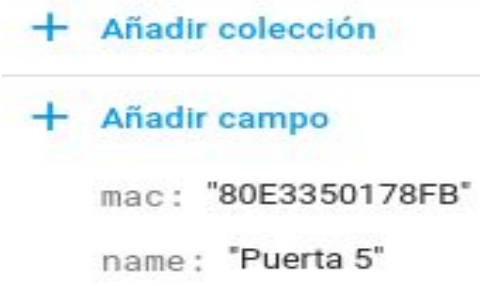


Figura 4.18: Ejemplo de documento device

4.7. Estadísticas de los usuarios

Este apartado pretende expresar lo que representan las gráficas del sistemas. Para probar que todo el sistema en general y ver que tiene un buen funcionamiento se hizo una carga de dato en la plataforma firebase. Para realizar esta prueba se ha creado un clase en python que genera la cantidad exacta de documentos aleatorio para que con ellos se pueda probar y verificar las gráficas del sistema.

4.7.1. Visualizar las gráficas

En este apartado se hizo un pequeño estudio de cómo se hacer las gráficas en las páginas web. En un primera momento por lo que puede apreciar la realización de gráficos en html se utiliza la etiqueta canvas y gracias a alguna función se puede pintar dentro como si fuera un lienzo. Esto

es un trabajo complicado ya que se tiene que tener en consideración parámetros como: donde irá colocado la figura, si se quiere ver con relleno, si una línea va de tal punto, estar calculando dimensiones, etc. Como en este caso lo que se pretende es expresar información contenida en la base de datos de forma visual, se tendría que hacer a través de varias gráficas. Debido a este laborioso y complicado trabajo se opta por una solución más cómoda la utilizar una librería para hacer graficas. Después de mirar algunos artículo [48] se toma la decisión de usar la librería google chart.

4.7.2. Ejemplo del funcionamiento de las gráficas

- En este primer punto se mostrará una lista con todos lo usuarios que han pasado por todo los dispositivo en el mismo día o por el dispositivo que seleccionara (ver Figura 4.19),como nota aclaratoria todo el proceso de selección de datos y conteos está realizado en javascript.

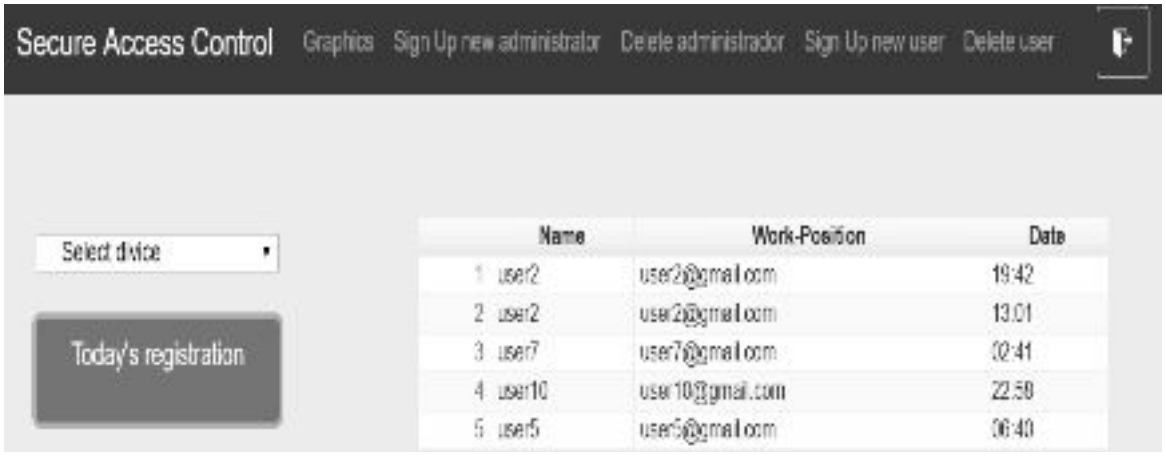


Figura 4.19: Registro del día.

- En estas gráfica se llevará un conteo y se mostrará la cantidad de gente que ha pasado esta semana cada día hasta el momento de ahora(ver Figura 4.20) o el conteo de la cantidad de gente que ha pasado por el dispositivo esta semana cada día hasta el momento de ahora .

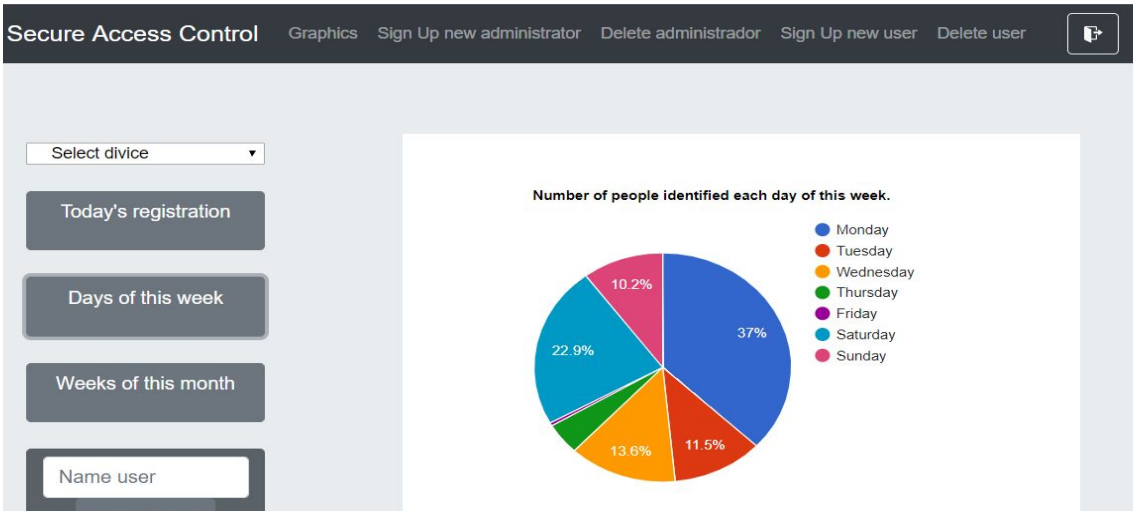


Figura 4.20: Gráfica del registro de la semana en la que estoy.

- La siguiente gráfica (ver figura 4.21) expresa la cantidad de personas identificadas por semana del mes de la consulta para un dispositivo en concreto o para todo el sistema. Definiendo la semana como los días comprendido desde lunes a domingo. Para realizar esta selección y conteo de los datos se utiliza javascript.

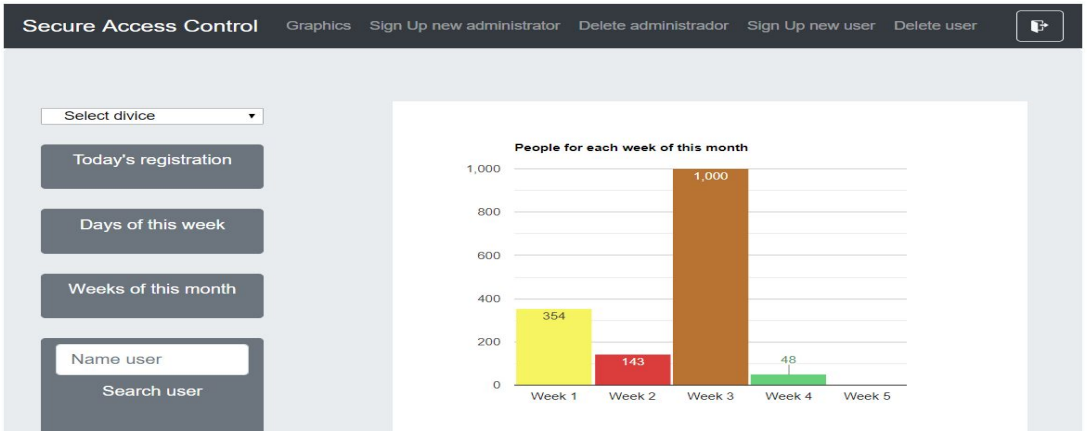


Figura 4.21: Gráfica de cantidad de registro por semana del mes actual.

- En la última pestaña se listan los registros de un usuario por todos los dispositivos o por un dispositivo en concreto (ver Figura 4.22).

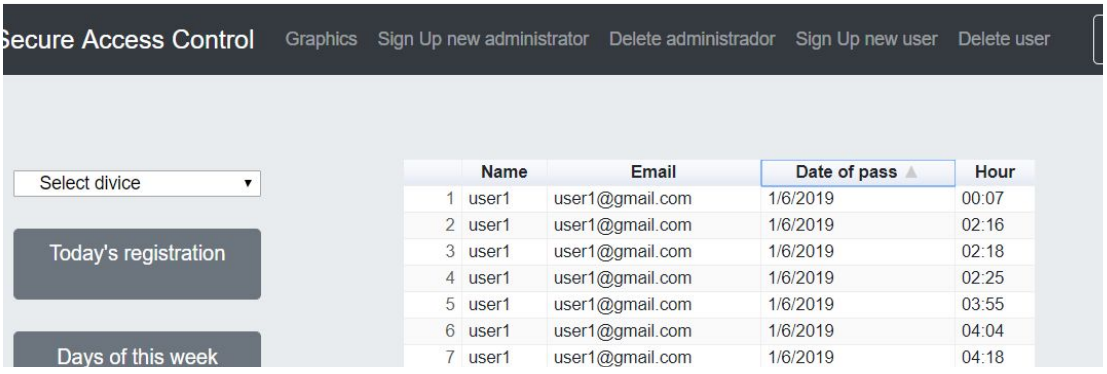


Figura 4.22: Buscar usuario en el sistema o en un dispositivo.

5. Capítulo 5. Seguridad Implementada en el Sistema

En este capítulo se expondrán todo lo relacionadas con la seguridad que se a introducido al sistema o que por otros motivos también se puede considerar parte de la seguridad para el sistema.

5.1. Reglas de seguridad en firebase

Se exponen las reglas puestas en la plataforma de firebase para que solos los usuarios registrados pueden acceder a los datos del sistema para leer y escribir(ver Figura 5.1). Como se observa, las reglas se separan por servicio y por colección como vemos el servicio cloud.firestore hay tres colecciones las cuales no se podrá acceder a estas ni para lees ni para escribir sin previamente estar autentificado en la plataforma.

```
service cloud.firestore {
  match /databases/{database}/documents {
    match /passVerification/{document=**} {
          allow read: if request.auth.uid != null;
          allow write: if request.auth.uid != null;

    }
    match /users/{document=**} {
          allow read: if request.auth.uid != null;
          allow write: if request.auth.uid != null;
    }
    match /device/{document=**} {
          allow read: if request.auth.uid != null;
          allow write: if request.auth.uid != null;
    }

  }
```

Figura 5.1: Reglas firebase.

5.2. Reglas de seguridad de lógica de funcionamiento del sistema

Las reglas de seguridad aplicadas a este sistema son:

- Jerarquía de usuario en el que hay un usuario que controla a los demás y es el único que puede ingresar los usuarios administradores, el llevara en sí la gestión de todo.
- El sensor dactilar y la cámara solo se puede acceder desde un dispositivo, por lo que si se intenta un acceso simultáneo será cancelada la operación.
- Por razones de seguridad el acceso al dispositivo es limitado solo el

usuario root y al grupo dialout “marcar hacia afuera” (ver Figura 5.2) . Para poder acceder al dispositivo, se creará un usuario y se agregara al grupo dialout.

```
ls -l /dev/ttyUSB0
crw-rw---- 1 root dialout 188, 0 mar  1 16:25 /dev/ttyUSB0
```

Figura 5.2: Comprobación de la existencia del grupo dialout para los dispositivos.

- Tanto el reconocimiento como el registro solo se podrá hacer desde el dispositivo.
- Para poder registrar un nuevo usuario solo es posible si se hace directamente desde el aparato, esto se realiza mediante la comprobación ip del aparato, si no es el propio aparato no podrá enviar la página newUser y dará error de acceso a la página.
- Por la característica del sistema el usuario tipo usuario tendrá que registrarse en cada dispositivo por el que se quiera que se identifique.
- Incorporación de cifrado con AES el vector característico del dedo.

5.3. Seguridad en la parte del servidor

Se expone a continuación los puntos relacionados con la seguridad en la parte del servidor:

- Se verá que usuario es el que está navegando y se enviaran o no ciertas páginas. Se comprobará si es root o otro usuario mediante una Cookie y previa identificación del usuario en el sistema. Esto solo es valido para los usuario administrador y root (ver Figura 5.3).

```
if (req.cookies.email != 'root@gmail.com') {
  res.render('menuAdm', { title: titleApp, iam: 'root' });
} else {
  res.render('menuAdm', { title: titleApp, iam: 'other' });
}
```

Figura 5.3: Comprobación para enviar página según si es root o no.

- El servidor no es aconsejable arrancar como root ya que todo los usuario que se conectasen al sistemas se convertirían en superusuario, haciendo que todo fuese accesible, por lo anterior tendremos que crear un usuario y ponerlo en el grupo dialout introduciendo el siguiente comando **sudo usermod -a -G dialout ServidorWeb** además poner el grupo a los archivo **chgrp dialotu archivo.py.**

6. Capítulo 6. Dificultades, Líneas Futuras y Conclusiones

Este capítulo enumera las dificultades que se han presentado en el desarrollo del proyecto. También se discutirán los problemas que pueden mejorarse en este sistema, tanto en la parte de hardware como de software y, en mi opinión, faltaría para que el sistema sea comercial y competitivo. Por último, se describen las conclusiones a las que se ha llegado después de finalizar el proyecto.

6.1. Dificultades

- Aprender el funcionamiento de los dispositivos. El hecho de trabajar con dispositivos separada y generar interacción entre ellos implica una cierta dificultad.
- Aprender como poder realizar streaming.
- Aprender a generar y eliminar procesos en el sistema operativo.
- Aprender a trabajar con hilos.
- Aprenda a trabajar con node y socket.io de javascript
- Aprender trabajar con firebase.
- Aprender sobre como hace el reconocimiento facial.
- Aprenda sobre el sistema de procesamiento de imágenes.
- Tener que aprender un nuevo lenguaje de programación, (python).
- Tener que optimizar mucho ciertas tareas o realizarlas de una manera en específica para poder hacer que el sistema funcione correctamente debido a la limitación de procesamiento por parte de la raspberry.

6.2. Líneas futuras

- Realizar la aplicación con la nueva versión de angula reduciendo al cándida de archivo y de código.
- Mejora estética del sistema. Mejorar aspecto exterior hardware del sistema.
- Añadir nuevos tipos de gráficas.
- Se puede hacer que el usuario seleccione el tipo de gráfica y los datos a representar.
- Hacer que el usuario root pueda configurar el sistema, por ejemplo: seleccionar tipos de algoritmo a utilizar, añadir logos al

sistema, cambiar colores de la app, formatos de imagen, opciones de visualización, etc.

- Añadir los datos que se deseen pedir a los usuarios del sistema.
- Agregue otra funcionalidad que podría llamarse ubicación, para encontrar al usuario en el sistema que indica dónde se registró por última vez.
- Creación de una base de datos que pertenezca al sistema sin que dependencia a firebase.
- Optimizar el sistema para el ingreso de un nuevo dispositivo en el sistema.
- Mejorar la experiencia de usuario haciendo que el sistema detecte la falta de espacio para no poder ingresar más usuario.

6.3. Conclusiones

Para concluir, los objetivos de este TFG se han alcanzado, que son en resumen la realización de la doble autentificación del usuario, el registro previamente del usuario en el sistema y la posibilidad de visualizar los datos. Se ha intentado hacer un sistema de control de acceso que todo los datos estén centralizado. Todo los datos de las gráficas se podrán ver desde cualquier lugar de la empresa y por solo personal encargado. También se ha simplificado el proceso de registro del usuario pero es mejorable. Mencionar también que la falta de conocimiento y prácticas en algunos aspectos que se ha desarrollado en sistema han hecho que se haya tenido que dedicar mucho tiempo en la adquisición del conocimiento necesario para su realización. Destacar que en el campo del reconocimiento facial queda mucho por hacer y aprender, no hay sistemas que den como reconocido un usuario con un valor de 100% de seguridad y uno de los déficit más llamativo es la variable tiempo en la persona debido a la forma en la que se reconoce a la persona el hecho de envejecer hace que cualquier sistema identificador puede dejar de reconocer a las personas de ahí que estos sistemas tengan que cada cierto tiempo actualizarse.

7. Chapter 7. Difficulties, future work and conclusions

This chapter lists the difficulties they have presented in developing the project. They will also discuss the problems that can be improved in this system, both in the hardware and software part and, in my opinion, would be missing for the system to be commercial and competitive. Finally, the conclusions reached after completing the project are described.

7.1. Difficulties

- Learn the operation of the devices. The fact of working with separate devices and somehow generating interaction between them implies some difficulty.
- Learn to stream videos.
- Learn to generate and eliminate processes in the operating system.
- Learn to work with threads.
- Learn to work with node and socket.io of javascript
- Learn to work with firebase.
- Learn about how it does facial recognition.
- Learn about the image processing system.
- Having to learn a new programming language (python).
- Having to update many tasks or perform tasks in a specific way to be able to make the system work properly due to the limitation of processing by the raspberry.

7.2. Future Work

- Make the application with the new version of Angula reducing the file number and code.
- Aesthetic improvement of the system. Improve external appearance system hardware.
- Add new types of graphics.
- You can have the user select the type of graph and the data to be

represented.

- Make the root user able to configure the system, for example: select algorithm types to use, add logos to the system, change app colors, image formats, display option, etc.
- Add the data that you want to ask the users of the system.
- Add other functionality that could be called location, to find the user in the system that indicates where they last registered.
- Creation of a database belonging to the system without dependence on firebase.
- Optimize the system to introduce a new device into the system.
- Improve the user experience by having the system detect the lack of space so that no more user can enter.

7.3. Conclusions

To conclude, the objectives of this TFG have been achieved, which are in summary the realization of the double authentication of the user, the previous registration of the user in the system and the possibility of visualizing the data. An attempt has been made to make an access control system that all data is centralized. All graphics data can be viewed from anywhere in the company and only by the staff in charge. The user registration process has also been simplified but it can be improved. The lack of knowledge and practice in some aspects that has been developed in the system has meant that much time has been devoted to acquiring the knowledge necessary for its realization. Note that in the field of facial recognition there is much to do and learn, there is no 100% effective system and one of the most striking deficits is the time variable in the person, the face changes over time and this can cause the system to not recognize and it is necessary to update the system from time to time.

8. Capítulo 8. Presupuesto

8.1. Recurso hardware

Descripción	Importe €
Raspberry	40 €
Cámara	25 €
Sensor dactilar	12 €
Tarjeta micro sd 16	5 €
Precio total	82 €

Tabla 9.1: Presupuesto para parte hardware.

8.2. Recursos humanos

Descripción	Precio por hora	Horas invertidas	Total
Análisis y diseño software.	20€	80h	1600€
Implementación software.	10€	160h	1600€
Total		240h	3200€

Tabla 9.2: Presupuesto de recursos humanos.

8.3. Presupuesto final

Tipo	Importe €
Recurso Hardware.	82
Recurso Humanos.	3200
IGIC 7%	229.74
Total	**3511 €**

Tabla 9.3: Presupuesto final.

9. Bibliografía

[1] Raspberry Pi[online]. Wikipedia. Fecha de consulta: 14/04/2018.
https://es.wikipedia.org/wiki/Raspberry_Pi
[2] Sensor dactilar [online]. Wikipedia. Fecha de consulta: 14/04/2018.
https://es.wikipedia.org/wiki/Sensor_de_huella_digita
[3] Sir Guillermo Herche [online]. Wikipedia. Fecha de consulta: 11/06/2019.
https://es.wikipedia.org/wiki/William_Herschel
[4] Alphonse Bertillon [online]. Wikipedia. Fecha de consulta: 11/06/2019.
https://es.wikipedia.org/wiki/Alphonse_Bertillon
[5] Francis Galton [online]. Wikipedia. Fecha de consulta: 11/06/2019.
https://es.wikipedia.org/wiki/Francis_Galton
[6] Juan Vucetich [online]. Wikipedia. Fecha de consulta: 11/06/2019.
https://es.wikipedia.org/wiki/Juan_Vucetich
[7] Características técnicas de la cámara raspberry pi [online]. Página oficial de raspberry-pi. Fecha de consulta: 11/06/2019. https://www.raspberrypi.org/blog/new-8-megapixel-camera-board-sale-25/
[8] Página oficial de Raspberry [online]. Bajar sistema operativo raspbian. Fecha de consulta: 14/04/2018.
https://www.raspberrypi.org/downloads.
[9] Página oficial de Python [online]. Python downloads . Fecha de consulta: 14/04/2018.
https://www.python.org/downloads/
[10] Página oficial OpenCV [online]. Opencv. Fecha de consulta: 11/06/2019. https://opencv.org/
[11] Node.js [online]. Wikipedia. Fecha de consulta: 20/07/2018. https://es.wikipedia.org/wiki/Node.js
[12] Página oficial de nodejs [online]. Nodejs downloads. Fecha de consulta: 20/07/2018.
https://nodejs.org/en/download/
[13] Npm [online]. Wikipedia. Fecha de consulta: 20/07/2018. https://es.wikipedia.org/wiki/Npm
[14] Página oficial de npm [online]. Npm. Fecha de consulta: 20/07/2018. https://www.npmjs.com/
[15] Firebase [online]. Wikipedia. Fecha de consulta: 22/07/2018. https://es.wikipedia.org/wiki/Firebase
[16] Página oficial de Firebase [online]. Firebase. Fecha de consulta: 22/07/2018.
https://firebase.google.com/
[17] Página oficial de Bootstrap [online]. Como se utiliza Bootstrap. Fecha de consulta: 3/08/2018.
http://getbootstrap.com/
[18] Página oficial de Google [online]. Como se utiliza Google Charts. Fecha de consulta: 3/08/2018.
https://developers.google.com/chart/interactive/docs/gallery
[19] Página de descarga DIA [online]. Aplicación DIA. Fecha de consulta: 3/08/2018.
https://wiki.gnome.org/Apps/Dia
[20] Página oficial GitHub.GitHub[online]. Fecha de consulta: 3/08/2018. https://github.com/
[21] GitHub[online]. Wikipedia. Fecha de consulta: 3/08/2018. https://en.wikipedia.org/wiki/GitHub
[22] Página oficial Sublime-Text. Descarga de Sublime-Text. Fecha de consulta: 3/08/2018.
https://www.sublimetext.com/3
[23] Página comercial[online]. Sensor dactilar. Fecha de consulta: 3/06/2018.
https://www.adafruit.com/product/751
[24] Manual en pdf. Lector de reconocimiento ocular. Fecha de consulta: 3/06/2018.
http://www.drsecurity.net/productos/CA/IMG/BIOMETRICOSIRIS/IRS-ICAM7111.pdf
[25] Página Genbeta[online]. Software de reconocimiento facial. Fecha de consulta: 3/06/2018.
https://goo.gl/1Bgxis
[26] Página venta de productos[online]. Lector de tarjetas. Fecha de consulta: 3/06/2018.
https://goo.gl/9URwJk
[27] Página venta de productos[online]. Lector de proximidad de tarjeta. Fecha de consulta: 3/06/2018.
https://goo.gl/sEQdM5

[28] Manual en pdf. Teclado numérico para identificación de contraseña. Fecha de consulta: 3/06/2018. https://goo.gl/dusrZ3

[29] Manual en pdf. Sensor mixto. Fecha de consulta: 3/06/2018. http://www.drsecurity.net/productos/CA/IMG/BIOMETRICOSFACIALES/ZK-iFACE302.pdf

[30] Página Youtube[online]. Video comercial de tarjeta de proximidad. Fecha de consulta: 3/06/2018. https://youtu.be/z5IIX0iODyY

[31] Página Youtube[online]. Identificador con las venas de la mano. Fecha de consulta: 3/06/2018. https://www.youtube.com/watch?v=rBcdQ3nr1qE

[32] Página venta de productos[online]. Control de Presencia biométrico facial. Fecha de consulta: 3/06/2018. https://goo.gl/Kb4mvq

[33] Página Youtube[online]. Control de Presencia biométrico facial. Fecha de consulta: 3/06/2018. https://goo.gl/7cuVCg

[34] Blog de diseño web. Mejor Frameworks CSS. Fecha de consulta: 3/06/2018. https://www.silocreativo.com/mejores-frameworks-css/

[35] Documentación módulo python Picamera. [online]. Página picamera. Fecha de consulta: 11/06/2019. https://picamera.readthedocs.io/en/release-1.13/

[36] Formatos [online]. Página xataka. Fecha de consulta: 11/06/2019. https://www.xataka.com/basics/asi-es-webp-el-formato-que-pesa-un-64-menos-que-un-gif

[37] Algoritmo LBPH[online]. Página compartirá conocimiento. Fecha de consulta: 18/06/2019. https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b

[38] Página oficial de python. Librería de python para reconocimiento facial. Fecha de consulta: 3/07/2018. https://pypi.org/project/opencv-python

[39] Base de datos de imágenes de rostros[online]. Cambridge University Computer Laboratory . Fecha de consulta: 25/06/2019. https://www.cl.cam.ac.uk/research/dtg/attarchive/facesataglance.html

[40] Artículo en pdf de reconocimiento como se reconoce una la huella. Journal of Applied Research and Technology 7 . Fecha de consulta: 3/07/2018. http://www.scielo.org.mx/pdf/jart/v10n5/v10n5a11.pdf

[41] Imagen de un blog sobre huellas forenses. Fecha de consulta: 3/07/2018. http://ellegadoenlascenadeldelito.blogspot.com/2014/05/puntos-caracteristicos-de-las-huellas.html

[42] Página oficial de python. Librería de python para reconocimiento dactilar. Fecha de consulta: 3/07/2018. https://pypi.org/project/pyfingerprint/