Digitalización de Elecciones Electorales mediante Tecnología IoT (Digital Polls s.l.)

AUTORES: **Ernesto Colás Lanuza, David Solà Solé, Bernat Mir Masnou**

FECHA: 4 de Octubre del 2018

Resumen

Estamos viviendo unas épocas donde la transformación digital está a la orden del dia. Sin embargo, hay ciertas instituciones que se niegan a dar el siguiente paso y siguen utilizando procedimientos antiguos. Un claro ejemplo de una institución que sigue realizando sus procesos de forma arcaica es la administración pública y más concretamente en el ámbito de las votaciones.

Actualmente, cuando se va a votar, en la mesa donde se deposita el voto hay tres personas: el presidente de la mesa y dos vocales. La función principal de estas personas es presidir el acto de votación, controlar el desarrollo y realizar el recuento y escrutinio.

El presente trabajo pretende aportar una solución tecnológica viable al proceso de autentificación de las personas durante el ejercicio de unas elecciones, integrando un proceso de transformación digital que permite una reducción de colas y disminuye el error producido por el factor humano.

Este documento se compone de una introducción al estado del arte relacionado con las votaciones, la especificación de la solución y cómo se ha diseñado. Se explicarán los componentes tanto hardware como software que se han utilizado para conseguir la solución y los resultados obtenidos. Finalmente se presentará un modelo de negocio para llevar a cabo este proyecto y las conclusiones, que incluyen las líneas futuras y las contribuciones individuales de cada componente del grupo.

Abstract

We are living in an era where digital transformation is the order of the day. However, there are certain institutions that do not want to do with the next step. An example of an institution that follows the processes of the archaic form is the public administration and more specifically in the field of polling.

Currently, when voting, at the table where the vote is deposited there are three people: the president of the table and two members. The main function of these people is to preside over the voting act, to control the development and to carry out the work and the scrutiny.

The present work aims to provide a viable solution to the process of authentication of people during the exercise of elections, integrating a process of digital transformation that allows a reduction of queues and the response of the error produced by the human factor.

This document is composed of an introduction to the state of the art related to voting, the specification of the solution and how it was designed. The components will be explained both in the hardware and in the software that has been used to obtain the solution and the results. Finally, we present a business model to carry out this project and the final conclusions.

ÍNDICE

Introducción 5
¿Què es el “Internet de las cosas”? 5
Objetivos 6

Situación actual 7
Voto físico y voto electrónico 7
Elementos clave en unas votaciones 7

Especificación de la solución 8

Diseño de la solución 10

Componentes de la solución 13
Hardware 13
PN532 13
Tarjeta NFC 14
Arduino 14
Shield Ethernet 16
Sensor de huella dactilar 18
Raspberry Pi 18
Software 20
Arduino IDE 20
Librerías 21

Pruebas de la solución y resultados 23

Modelo de negocio 24
Modelo de negocio canvas 24
Plan de Marketing 28
Análisis D.A.F.O. 28
Valor diferencial 28
Política de Producto. 28
Política de Servicio y Atención al cliente 29
Política de Precios. 30
Estrategia de publicidad y promoción 30
Profesionales 30
Usuarios finales (Gobiernos de estado, territorio o ciudad) 31
Estrategia de comunicación y relaciones públicas 32
Plan de Acción de Marketing 33
Plan de Establecimiento. 33

Plan de Ventas 34
Estrategia de Ventas. 34
El personal de Ventas. 35
Herramientas de venta 36
Barreras de venta 36
Aspectos Legales y Societarios 37
La sociedad. 37
Estatutos y acuerdos de los socios 37
Junta general 37
Administradores 38
Derechos de los socios 39
Obligaciones legales. 39
Licencias y derechos 40
Plan económico-financiero 40
Confidencialidad del documento 40
Proyecciones y previsiones 40
Hipótesis 41
Descripción económica 42
Cuenta de pérdidas y ganancias 46
Cuenta de Tesorería 48
Balance de situación 51
Punto de equilibrio 53

Conclusiones 54

Mejoras futuras 55

Bibliografía 56

Anexo 57
Código fuente 57
PN532 57
Sensor huella dactilar 83
Servidor local 83
iniDB.py 83
utilsSQL.py 85
gestorServer.py 86
Display 88
gestorDisplay.py 88
Conexiones hardware 91

1. Introducción

1.1. ¿Què es el “Internet de las cosas”?

A día de hoy, el “Internet de las cosas” (de ahora en adelante IoT, del inglés *Internet of Things*) se está empezando a posicionar como una de las tendencias con más expectativas de impacto de cara al futuro, tanto en el ámbito profesional como fuera de él. Es un concepto que no solamente tiene el potencial de afectar cómo vivimos sino también cómo trabajamos. En definitiva, consiste en la interconexión digital de objetos cotidianos con Internet.

En 2009 un profesor del MIT, Kevin Ashton, usó por primera vez la expresión “Internet of Things” de forma pública en el RFID journal y, desde aquel entonces, el crecimiento y la expectación alrededor del término empezó a aumentar. Sin embargo, él mismo comentó que esta expresión ya fue de uso corriente en círculos internos de investigación desde 1999.

Una “cosa” dentro del IoT puede ser un teléfono celular, una máquina de café, unos auriculares, una lampara, un enchufe, una lavadora y con casi cualquier cosa imaginable. Este concepto también se aplica en componentes de máquinas, como por ejemplo el sistema de frenado de un coche o una fresadora. Se estima que para 2020 habrán aproximadamente 50.1 billones de dispositivos conectados frente a los 34.8 billones que tenemos a día de hoy. El IoT es una red gigante de “cosas” conectadas (las cuales también incluye a las personas). Las relaciones serán gente-gente, gente-cosas y cosas-cosas, siendo este último el que tenga mayor protagonismo.

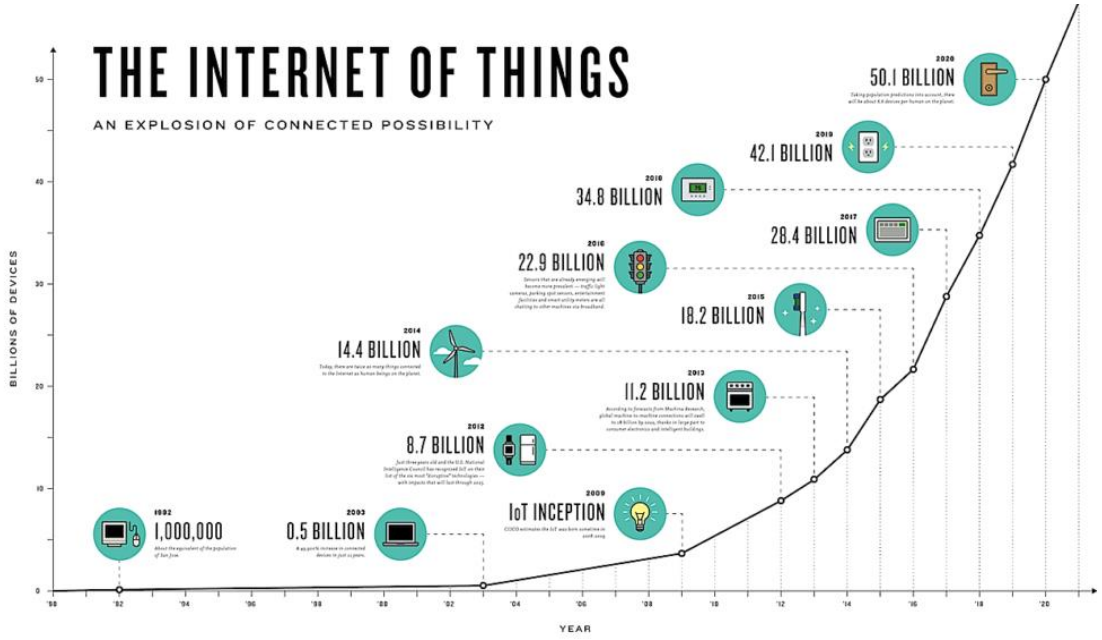


Fig. 1-I: Tendencia del IoT a lo largo de los años [1]

1.2. Objetivos

Este trabajo de fin de Máster tiene como objetivo final la automatización del proceso de autentificación (o verificación) del votante durante el ejercicio de unas votaciones.

Concretamente, gracias a la solución aportada por este proyecto, nuestro propósito consiste en la reducción, al máximo posible, del tiempo de espera en las colas y el factor de error humano durante el proceso de verificación del votante. Una vez finalizadas las votaciones. Se pueden poner en común todos los datos recopilados a lo largo de la jornada para sacar información relevante de cara a la mejora de la gestión de los recursos por parte de los ayuntamientos (decidir el número de mesas por cada colegio, observar las tendencias de la gente a la hora de ir a votar para realizar un mejor balanceo de la carga en horas punta, etc).

A nivel personal, nos parece muy interesante y provechoso tener un contacto más cercano con las tecnologías que rodean al mundo del Internet de las Cosas.

2. Situación actual

En 2012 el huracán Sandy golpeó duramente la costa Este de Estados Unidos y se predijo que podría llegaría a afectar durante el día del proceso de elecciones a la presidencia del país. Con lluvias torrenciales, fuertes vientos e inundaciones, los funcionarios electorales se encontraron con grandes retos a resolver. Entre ellos se encontraban los cortes de energía, inundaciones y tormentas de nieve que podrían llegar a impedir las votaciones en múltiples Estados.

A consecuencia de esto, se permitió de forma excepcional, el uso del voto electrónico, aunque de forma improvisada. Se puso a disposición la facilidad de descargar una papeleta, en formato digital, para ser rellenada y posteriormente enviada vía correo electrónico o fax.

Esta solución presenta innumerables problemas de seguridad por lo que es conveniente tener en cuenta por qué a día de doy, la gran mayoria de paises siguen aún con el modelo de voto físico.

2.1. Voto físico y voto electrónico

El voto físico existe desde hace centenares de años, y durante todo este tiempo, se ha probado contra él cualquier método imaginable de fraude electoral y a día de hoy, aún sigue defendiéndose. Su principal ventaja reside en que los ataques hacia este sistema de voto no escalan bien. Se requiere de mucho esfuerzo, mucha gente y únicamente se necesita de una persona sacar a la luz el fraude.

Con el voto electrónico, se puede realizar el ataque con una persona y el esfuerzo invertido en manipular un voto es el mismo al necesario para manipular un millón. Este ataque se puede realizar sin estar siquiera físicamente en el país donde se celebran las elecciones que se quieren manipular.

2.2. Elementos clave en unas votaciones

Existen dos partes fundamentales en unas elecciones. La primera es el anonimato. Si se llega a identificar de algún modo un voto, ya sea mediante una firma, un nombre o cualquier cosa donde se pueda comprobar cómo has votado, el voto es descartado de manera que no se pueda identificar a la persona para ser forzada o sobornada a votar de una cierta manera. El voto es guardado dentro de un sobre sellado donde es abierto delante de todo el personal electoral que participa en el proceso de contabilización de los resultados. De este modo, tienes la certeza de que tu voto ha sido tenido en cuenta aún sabiendo que no volverás a verlo.

La segunda parte consiste en la confianza. Nunca se puede confiar en una única persona, a ser posible, tampoco en un grupo reducido de dos o tres personas. La gente puede estar bajo los efectos del soborno, amenazada o pueden cometer algún error humano.

3. Especificación de la solución

Como se ha comentado en apartados anteriores, este proyecto pretende agilizar el sistema de votaciones instaurado actualmente y reducir los costes asociados. Así pues, el principal objetivo es facilitar tanto a los ciudadanos con derecho a voto como el personal presente en la mesa electoral el ejercicio de la votación de una forma sencilla y rápida.

Los principales componentes de la solución son:

- Lector RFID
- Sensor de huella dactilar
- Microcontroladores para la gestión de los componentes
- Servidor con base de datos

Los pasos a seguir en una votación según nuestra solución serían:

- Paso 0: el votante dispone de una tarjeta NFC donde tiene guardados todos sus datos de una forma confidencial. Esta tarjeta, como el documento nacional de identidad, es personal e intransferible. Dicha tarjeta, se proporcionaría en cualquier administración pública.

- Una vez el votante llega a su mesa, acerca su tarjeta NFC al lector RFID. El lector lee los datos asociados del votante y los envía al sensor de huellas dactilar.

- Al llegar los datos al sensor de huellas, estos se cargan temporalmente en la memoria flash. A continuación, el votante inscribe su huella en el sensor dos veces: la primera para generar una imagen de la huella introducida y la segunda para comprobar que son la misma.

  Una vez se ha hecho la inscripción de la huella, y se ha comprobado que coinciden la huella introducida y la recibida por el lector RFID, se envía al servidor local instalado en ese colegio.

- El último paso es la comprobación en el servidor local. Una vez llegan los datos del votante procedentes del sensor de huellas, se comprueba en la base de datos si los datos recibidos existen y si se puede votar. En caso que no se encuentren los datos o el votante ya haya votado, se mostrará un mensaje por la pantalla instalada en el servidor.

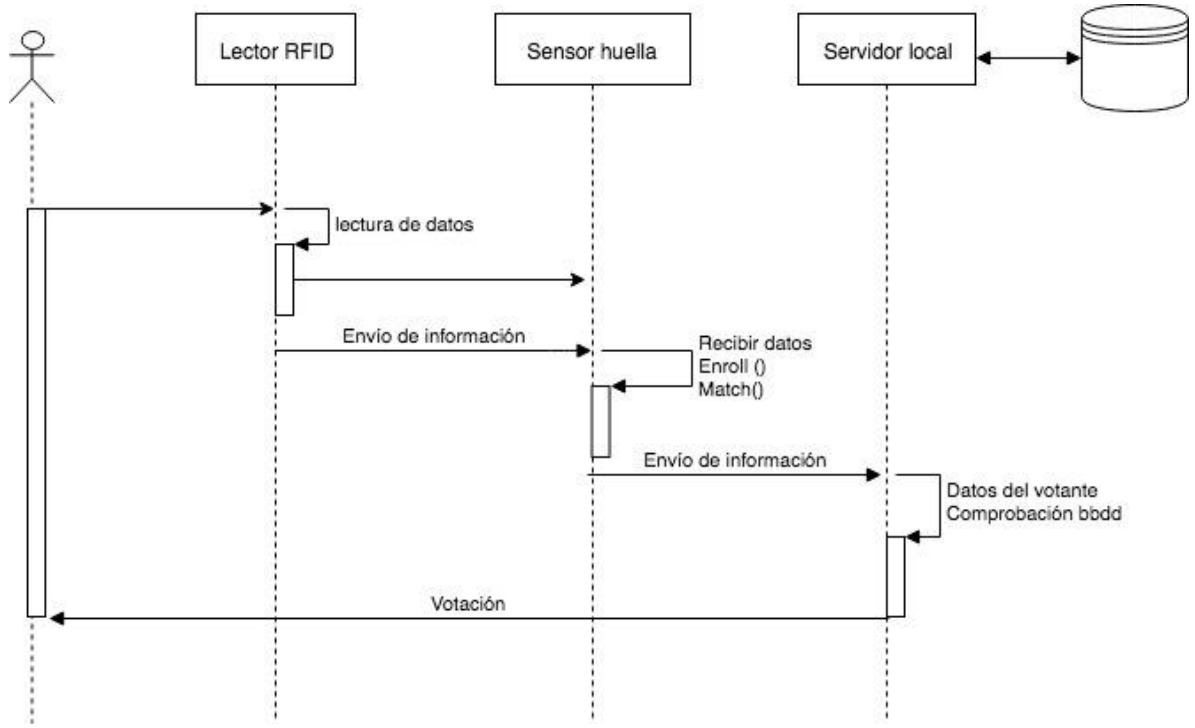


Fig. 3-I: Diagrama de secuencia del proyecto

4. Diseño de la solución

Este apartado, explicará qué opciones existen en el mercado para llevar a cabo este proyecto y por qué hemos elegido estos componentes.

Para el lector de tarjetas, se ha escogido el PN532. Este módulo de un coste moderado (alrededor de 25€) nos permite leer y escribir en tags RFID. Para controlar este dispositivo se ha usado el microcontrolador ARDUINO Uno permitiendo una fácil integración entre ambos componentes.

En referencia al sensor de huellas, hay una gran variedad de componentes en el mercado como por ejemplo el de Adafruit [3] de 49,95€, el de Digi-Key de 43,44€ [4] o el de Kookye [5] de 29,99€. De entre las múltiples opciones, teniendo en cuenta temas económicos y logísticos y que para nuestro caso de uso no nos hace falta una gran memoria flash, se ha escogido el del fabricante Kookye. Este componente es compatible con el de Adafruit y de esta forma nos podemos beneficiar de la gran comunidad de programadores de Adafruit. Para alimentar este sensor se necesitan de 5V por lo que se ha escogido el microcontrolador Arduino para alimentar dicho componente y nos permite utilizar las librerías de Adafruit de una forma sencilla.

Finalmente, para gestionar todo el sistema de votación se ha escogido la placa computadora de bajo coste por excelencia: Raspberry Pi 3 Model B. Esta placa, nos permite la conexión a Internet mediante Wi-Fi y nos da la posibilidad de instalar una base de datos y gestionar desde la placa el control de las votaciones.

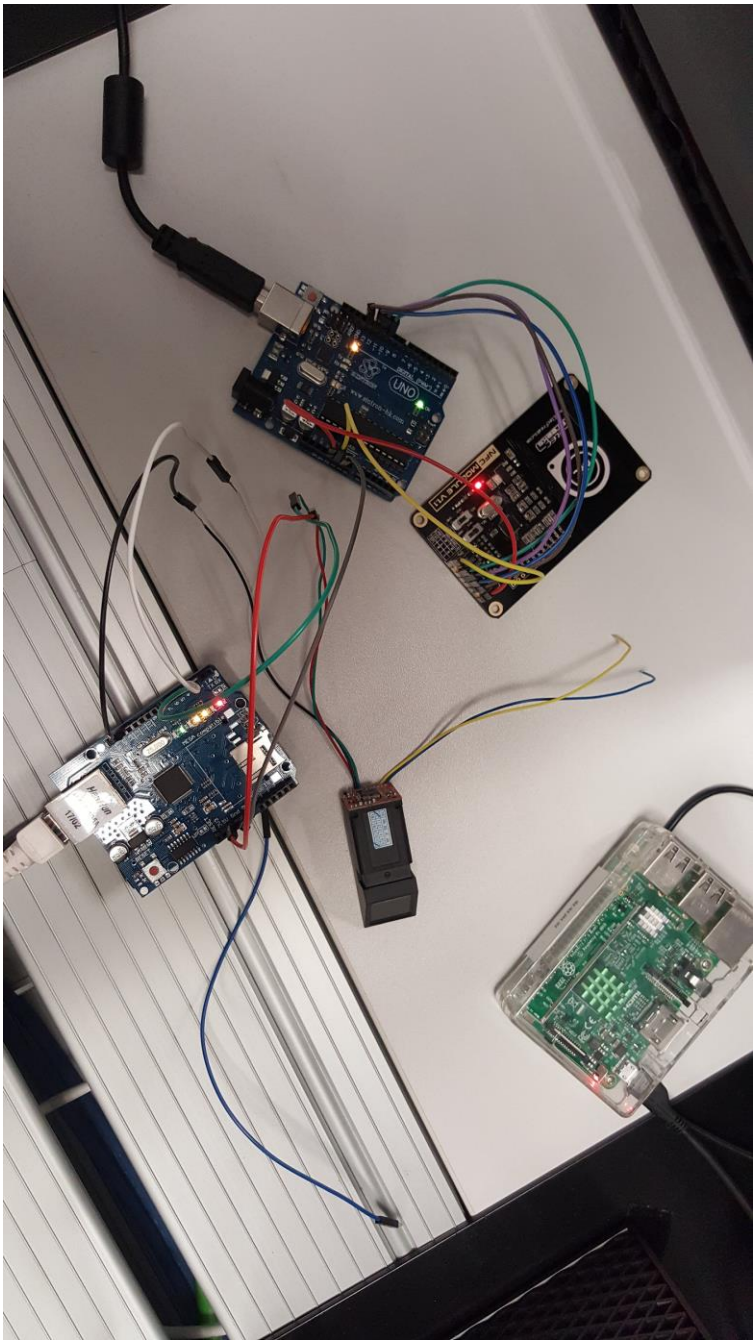


Fig. 4-I: Solución completa

En referencia a los métodos de comunicación entre componentes, se ha escogido los siguientes:

- Comunicación entre el lector RFID y el sensor de huellas: dado que estos elementos estarán cerca el uno del otro y que para esta prueba de concepto no requiere del envío de muchos datos, se ha escogido enviar los datos via série.

- Comunicación entre sensor de huella y servidor local: Para dicha comunicación y dado que los elementos pueden no estar cerca, se ha escogido el protocolo de envío de mensajes MQTT. Un problema podría ser la conexión a Internet pero se ha supuesto que en todos los colegios electorales se dispone de Internet.

En este caso el servidor local se comportará como broker y cliente escuchando los mensajes que lleguen para cada tópico al que está suscrito. Cuando el controlador del sensor de huellas publique mensajes en ese tópico y sean recibidos por el servidor local, este hará las comprobaciones necesarias en la base de datos y permitirá al votante votar si todas las comprobaciones han sido satisfactorias.

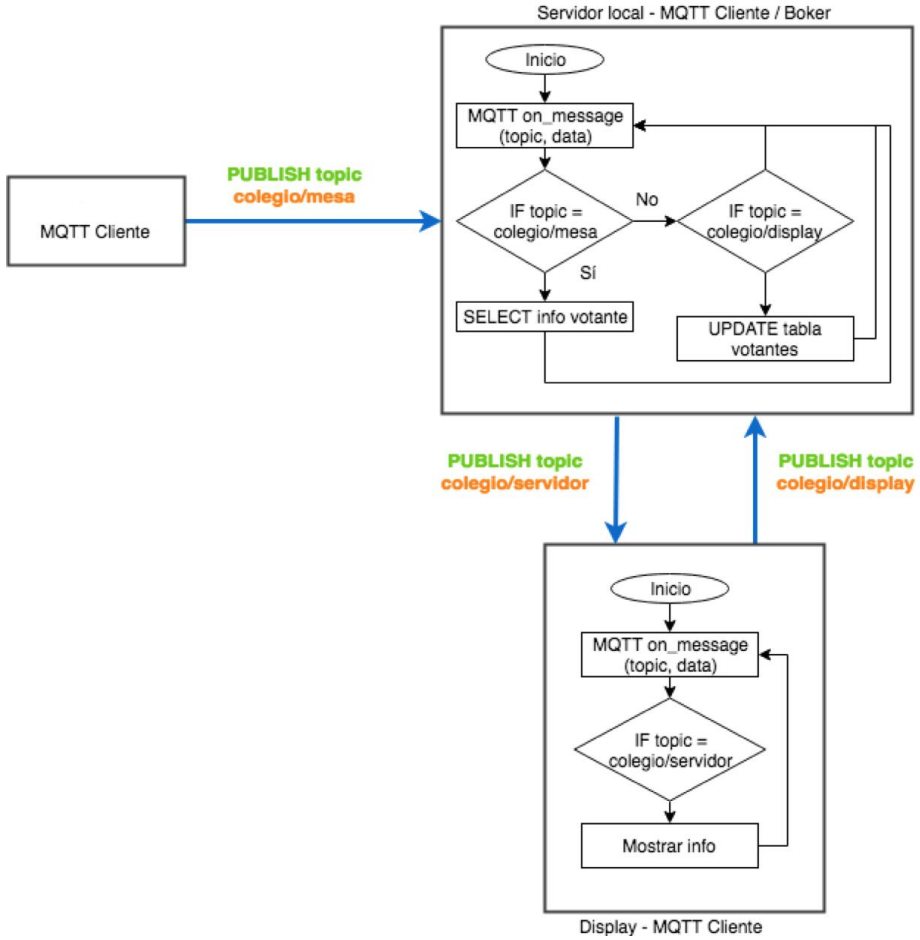


Fig. 4-II: Flujo de comunicaciones MQTT entre sensor de huellas y servidor local

5. Componentes de la solución

En esta sección, se explicarán los componentes utilizados tanto a nivel hardware como software para el desarrollo de este proyecto. En la parte hardware se explicará:

- PN532: módulo RFID que se encarga de leer tags MIFARE.
- Arduino: se usan dos Arduinos Uno. El primer microcontrolador que se encarga de gestionar el sensor de huella dactilar y transferir el resultado del emparejamiento de las huellas al servidor de datos mediante el protocolo MQTT cuya conexión a internet la consigue gracias a un shield Ethernet. El segundo, se encarga de la gestión del módulo PN532.
- Sensor de huella: el dispositivo utilizado para comparar diferentes plantillas de huella dactilar.
- Raspberry Pi: placa que se comporta como servidor local y se encarga de las funciones de bases de datos de la solución. Adicionalmente, la parte del display puede ser integrada con otra Raspberry Pi y una pantalla TFT.
- Arduino IDE: entorno de desarrollo.
- Librerías: se mencionan las librerías que se han utilizado para llevar a cabo este proyecto.

5.1. Hardware

5.1.1. PN532

Para la lectura RFID, se ha usado el módulo PN532. Este dispositivo se comunica por SPI, por lo que nos permite una fácil integración con el microcontrolador Arduino Uno. El mencionado módulo nos permite la lectura de una tarjeta RFID para posteriormente tratarla y mandarla al sensor de huella dactilar.

Fig. 5-I: Lector de tarjetas RFID PN532

Para poder trabajar con este módulo, se requiere de la librería PN532. Esta librería se explicará a continuación en este mismo apartado en la sección de librerías utilizadas.

5.1.2. Tarjeta NFC

Para guardar los datos del votante, se ha utilizado una tarjeta MIFARE Classic. Estas tarjetas nos permiten guardar la información del votante y son compatibles con el lector PN532 utilizado para leer/escribir en las tarjetas.

Fig. 5-II: Tarjeta NFC MIFARE Classic

5.1.3. Arduino

Arduino [8] es una plataforma de hardware libre que proporciona una placa de circuito impreso con un microcontrolador, normalmente un Atmel, puertos analógicos y digitales de entrada/salida y todos los componentes necesarios para su funcionamiento.

Arduino es una herramienta popular tanto dentro del desarrollo de soluciones enfocadas a IoT como para la enseñanza y el autoaprendizaje.

La primera placa fue introducida en 2005 para ayudar a estudiantes, sin conocimientos previos en electrónica ni en programación, a crear prototipos funcionales que permitieran conectar el mundo físico con el mundo digital. Cabe destacar que es posiblemente el proyecto de hardware libre más conocido actualmente y desde entonces, ha sido un referente a tener en cuenta al desarrollar proyectos que requieran de prototipado rápido.

Es importante fijarse que bajo su nomenclatura, se esconde una gran diversidad de placas con distintas características.

Para este proyecto, se ha decidido utilizar un Arduino Uno ya que nos permite una fácil integración con el sensor de huellas.

Fig. 5-III: Arduino Uno

Las principales características que proporciona este Arduino son las siguientes:

Microcontrolador	Atmega328
Voltaje de operación	5V
Voltaje de entrada (Recomendado)	7 – 12V
Voltaje de entrada (Límite)	6 – 20V
Pines para entrada- salida digital.	14 (6 pueden usarse como salida de PWM)
Pines de entrada analógica.	6
Corriente continua por pin IO	40 mA
Corriente continua en el pin 3.3V	50 mA
Memoria Flash	32 KB (0,5 KB ocupados por el bootloader)
SRAM	2 KB
EEPROM	1 KB
Frecuencia de reloj	16 MHz

5.1.3.1. Shield Ethernet

Para llevar a cabo la conexión entre el sensor de huella dactilar y el servidor local, se ha escogido el protocolo de comunicación MQTT. Para conseguir una conexión MQTT se requiere de conexión a internet.

Arduino Uno (*stand-alone*), no proporciona ningún chip Wi-Fi ni tampoco un conector RJ45. Se ha optado por la solución de utilizar una shield Ethernet para dar la capacidad de conectarse al Arduino a internet y poder así conectarse con el *broker* MQTT.

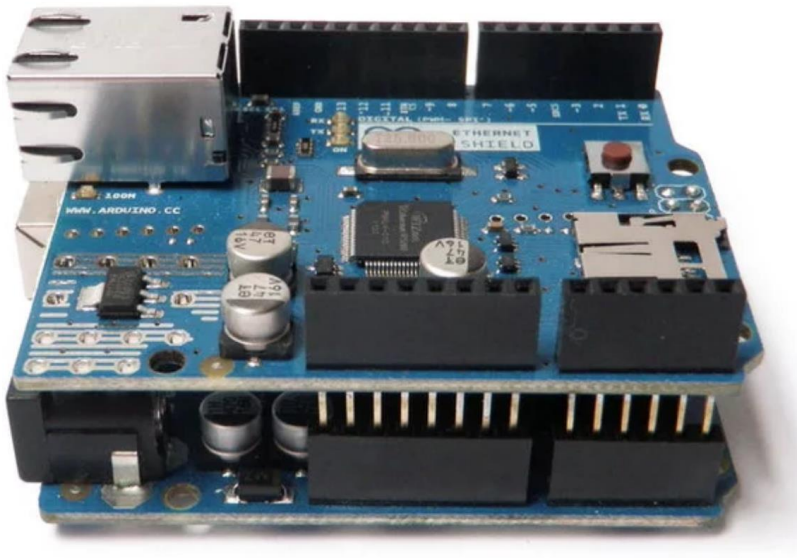


Fig. 5-IV: Shield Ethernet montado encima de un Arduino Uno

Se ha escogido la shield Ethernet del fabricante HanRun. La placa dispone de un controlador Wiznet W5100 que le da conexión a la red permitiendo el uso de los protocolos TCP y UDP/IP.

5.1.4. Sensor de huella dactilar

Para la autenticación del votante mediante la huella, se ha escogido el sensor del fabricante Kookye.

Fig. 5-V: Sensor de huella dactilar

El procesado de huellas de este sensor tiene dos fases: el registro con el sensor y la coincidencia de la huella que puede ser 1:1 o 1:N. Cuando se hace el registro, el usuario debe introducir la huella dos veces. El sistema procesará las dos imágenes y generará una plantilla de la huella basada en el proceso de los resultados.

Los principales parámetros del sensor son los siguientes [6]:

Potencia	DC 3.8V - 7.0V
Interfície	UART (TTL nivel lógico)
Baud rate	9600 bps
Tamaño de la plantilla	512 bytes

En referencia a la interfaz hardware y a la comunicación serie, se compone de 6 pines con las siguientes características:

# Pin	Nombre	Tipo	Descripción
1	Vtouch	in	Entrada de potencia de inducción
2	Sout	out	Señal de salida inductiva
3	Vin	in	Potencia de entrada
4	TD	out	Salida de datos (TTL)
5	RD	in	Entrada de datos (TTL)
6	GND	-	Tierra

Fig. 5-VI: Pines disponibles en el sensor de huella dactilar

5.1.5. Raspberry Pi

Raspberry Pi es una placa computadora de bajo coste (SBC) desarrollada en Reino Unido por la Fundación Raspberry Pi con el objetivo de incentivar la enseñanza de las ciencias de computación a los más jóvenes.

Durante los últimos años, ha presentado una gran aceptación a nivel mundial que ha llevado a la fundación a ir desarrollando nuevos modelos y nuevas versiones que mejoran sus características respecto a sus modelos predecesores.

Para este proyecto, se ha usado el modelo Raspberry Pi 3 Modelo B Rev 1.2[1] que cuenta con las siguientes características:

- Procesador Broadcom BCM2837, Cortex-A53 (ARMv8) 64-bit SoC a 1.2GHz
- 1GB LPDDR2 SDRAM de memoria RAM
- IEEE 802.11.b/g/n a 2.4Ghz (Wi-Fi) y Bluetooth 4.1
- Fast Ethernet 10/100 Gbps

[1] Para comprobar el modelo de la Raspberry Pi a través del terminal, puedes consultarlo con el comando `cat /sys/firmware/devicetree/base/model`

- GPIO 40 pines
- Salida HDMI
- 4 puertos USB 2.0
- Micro SD
- Micro USB (alimentación)
- Precio aproximado de 35 euros

Fig. 5-VII: Raspberry Pi 3 model B

Por cuestiones de diseño de la placa, el sistema operativo (SO) no se encuentra almacenado en un disco duro o una unidad de estado sólido (SSD), sino que está en una tarjeta Micro SD con una capacidad mínima requerida de 2GB. Aunque, por razones de eficiencia, se recomienda usar una tarjeta de 4GB por lo menos.

La Raspberry Pi se envía sin unidad de almacenamiento (Micro SD), por lo que debemos obtener una e instalarle una distribución de Linux . Existen múltiples opciones como Raspbian, Arch Linux, Pidora, RaspBMC, WIndows Iot Core, FreeBSD, Kali Linux, etc. En este proyecto, el SO utilizado ha sido Rasbian GNU/Linux versión 9[2].

Una interesante funcionalidad añadida respecto a sus versiones predecesoras, es la adición de una antena Wi-Fi por lo que ya no es necesario dedicar un puerto USB exclusivamente a una antena externa.

Si en el apartado anterior se veía que Arduino era una referencia en el mundo del hardware libre para microcontroladores, Raspberry Pi es su análogo dentro del sector de los microprocesadores.Cumpliendo también con la importante ventaja de contar con una gran comunidad detrás que le de respaldo.

Su pequeño tamaño, escaso consumo energético medio de aproximadamente 2.5W y la integración de una antena Wi-Fi, lo convierten en una interesante solución como servidor local de nuestra solución.

[2] Para comprobar la versión del SO instalado, se ha usado el comando de terminal `cat /etc/os-release`

5.2. Software

5.2.1. Arduino IDE

Para el desarrollo de la solución para el sensor de huella dactilar y el lector de RFID, se ha utilizado Arduino IDE (*integrated development environment*).

Arduino IDE es un entorno de desarrollo y en él se realiza la programación de las diferentes placas Arduino. El código fuente es desarrollado bajo la licencia pública general GNU. Este entorno soporta los lenguajes C y C++. El mencionado entorno suministra una librería de software llamada Wiring que proporciona muchos procedimientos comunes de entrada y salida. El código solo requiere de dos funciones básicas:

- setup() : esta función se ejecuta una vez se carga el código en la placa Arduino. La mencionada función solo se ejecutará una y solo una vez.
- loop(): es el bucle principal del programa. Esta función se ejecuta cíclicamente en la placa hasta el infinito.

El Arduino IDE emplea el programa avrdude para convertir el código ejecutable en un archivo de texto en codificación hexadecimal que se carga en la la placa Arduino que se carga en el firmware.

Fig. 5-VIII: Ejemplo código en el Arduino IDE

5.2.2. Librerías

Esta sección pretende explicar las librerías más importantes que se han usado en este proyecto. Así pues, las librerías utilizadas han sido:

- MQTT: para el Arduino que controla el sensor de huella dactilar se ha usado la librería PubSubClient [7]. Esta librería proporciona un cliente para enviar mensajes simples con un servidor que soporte MQTT.

  Una limitación conocida de la librería es que solo se pueden enviar mensajes de hasta 128 bytes pero para nuestro caso de uso nos son suficientes. Se ha escogido

esta librería por la fácil adaptación a nuestro código y ya que también soporta Arduino Ethernet.

- Sensor de huella: para poder hacer las comprobaciones de la huella dactilar se ha tenido que modificar la librería de Adafruit [8]. La librería propietaria no proporciona la opción de cargar una plantilla en la memória flash del sensor para luego comprobar si concuerda con la introducida por el usuario.

- PN532.h es la librería utilizada para el módulo PN532 que permite leer y escribir en distintas tipos de tarjetas (NFC) mediante Arduino usando el lector PN532 conectado mediante la interfaz SPI.

- MySQLdb: interfaz multi thread para bases de datos MySQL ofreciendo una API para Python.

- Paho MQTT: librería que ofrece una interfaz para crear un cliente MQTT, conectarse al broker y realizar operaciones de publicación y suscripción.

- Tkinter: paquete standard para el desarrollo de interfaces gráficas en Python.

- json: librería para la creación y manipulación de objetos en formato JSON.

- base64: librería usada para la serialización de binarios.

6. Pruebas de la solución y resultados

Este apartado explicará los principales problemas y los resultados que se han obtenido durante la realización de este proyecto.

En referencia al lector de tarjetas, primero se escogió el lector de bajo coste RC522. Desgraciadamente, este lector no conseguía leer todos los bloques del tag MIFARE por lo que se probó con otro lector, el PN532. Afortunadamente, este lector nos permitía leer todos los bloques de una tarjeta MIFARE Classic y también nos daba la posibilidad de escribir en dicha tarjeta.

Respecto al sensor de huella, el propio fabricante no da la opción de cargar una plantilla en el sensor. Por esta razón, se tuvo que modificar la librería para poder conseguir dichos resultados. Este sensor, dispone de dos buffers donde guardar los datos, así pues, guardábamos los datos de la huella en un buffer mientras que en el otro hacíamos toda la lógica de la generación del modelo/plantilla del votante. Otro problema que nos encontramos a la hora de cargar la plantilla en el sensor era que al enviar los datos el sensor no devolvía ningún reconocimiento así que no teníamos la certeza de que el paquete con los datos de la plantilla de 512 bytes hubiera llegado correctamente y se hubiera guardado. Para solucionar el problema, introducimos un delay entre el envío de cada cadena de caracteres para cerciorarnos que tenía el tiempo necesario para guardarlo en la memoria.

Así pues, hemos conseguido un prototipo estable en el que poder simular unas votaciones computerizadas de una forma rápida y segura. En este enlace se puede ver como se votaría mediante nuestra solución. Por otro lado, si el votante ya hubiera votado, este sería el resultado.

7. Modelo de negocio

La definición de modelo de negocio es complicada y tiene muchísimas variantes. La definición clásica dice que es “el plan previo al plan de negocio que define que se va a ofrecer al mercado, cómo se va a hacer, quién va a ser el público objetivo, cómo vas a vender el producto o servicio y cuál será el método para generar ingresos“.

En definitiva, es plasmar en un documento de cómo vas a crear, desarrollar y capturar valor. Una pequeña visión de todo lo que puede ser la startup en un futuro y los diferentes aspectos sobre los que se va a construir toda la empresa. Sería como los pilares de un edificio, siendo el edificio tu negocio y esos pilares el propio modelo.

Hay que destacar que el modelo de negocio es algo más que saber de dónde vienen los ingresos. Ganar dinero será una consecuencia de todo ese proceso de saber qué se ofrece, cómo se hace, cuál es el público target y demás.

7.1. Modelo de negocio canvas

El Business Model Canvas es la “plantilla” de modelo de negocio más popular del mundo. Desde que sus creadores, Alexander Osterwalder e Yves Pigneur, publicaron el libro que dio nombre a este “lienzo”, se ha convertido en un modelo de negocio utilizado para casi todas las nuevas startups y popularizado en concursos como los Startup Weekends.

Es una herramienta que tiene distintos apartados que se encargan de cubrir todos los aspectos básicos de un negocio, desde los segmentos de clientes hasta incluso los socios claves y la estructura de costes. En general, sigue la definición de modelo de negocio y busca plasmar en un solo lugar cómo se crea, entrega y captura ese valor de tu startup.

¿Y en qué consiste cada uno de los 9 apartados del Business Model Canvas?

- **Segmentos de clientes**: segmentación de mercado o grupo de personas a los que vamos a venderles nuestro producto o servicio. Puedes agrupar los públicos por las necesidades, canales, relaciones u ofertas. Algunos ejemplos de segmentos serían el mercado de masas (muy amplios), los nichos de mercado (muy específicos), los diversificados (distintos públicos muy distintos) o los multi-segmentos (que dependen de varios clientes a la vez).

  En nuestro caso: Creamos valor para el gobierno de turno pues da mayor veracidad a los resultados electorales. Crea valor a las consultoras encargadas de la logística de las votaciones pues reduce costes de estructura necesaria.

- **Propuesta de valor**: características y beneficios que se encargan de crear valor para cada uno de esos segmentos. En esta parte debes explicar qué es lo que ofreces a tus clientes y por qué van a comprarlo. Algunas características de esta propuesta podrían ser la novedad, el rendimiento, la personalización, el diseño o el precio.

  En nuestro caso: Votación con autenticación mediante carnet que contiene la huella dactilar y detección in situ de la misma. Mayor seguridad en la autenticación y reducción radical de la posibilidad de fraude. Reducción de la estructura necesaria para llevar a cabo las votaciones.

- **Canales**: medios a través de los que te vas a comunicar y vas a hacer llegar tu propuesta de valor al cliente. Pueden ser canales propios (de los socios) o externos y directos o indirectos y están divididos en 5 fases (notoriedad, evaluación, compra, entrega y postventa).

  En nuestro caso: Los canales serán mediante socios/empresas con conexiones con el gobierno de turno. Estos socios conocerán a la perfección las empresas encargadas de la logística de votaciones.

- **Relación con el cliente:** tipo de relación entre la startup y el cliente. Puede ser asistencia personal, self-service o automatizado (mezcla de ambas).

  En nuestro caso: Relación de asistencia contínua a nivel técnico de la solución: actualizaciones de la solución como resultado de una mejora continua. Contacto contínuo con clientes y partners para mantenernos como referencia para ellos.

- **Fuente de ingresos**: ¿de dónde va a llegar el dinero a la empresa? ¿Cómo se va a generar el beneficio? Algunos modelos de fuente de ingreso podrían ser la venta directa en un único pago, el pago por uso o la suscripción.

  En nuestro caso: Lógicamente nuestros clientes. Pagan para optimizar el coste de las vacaciones y para mejorar la veracidad del proceso. Nuestro cliente final son los gobiernos de distintos territorios.

- **Recursos clave**: los recursos más importantes para que todo lo anterior funcione. Pueden ser físicos (vehículos, edificios, …), intelectuales (patentes, copyrights, …), humanos (expertos clave, empleados muy valorados, …) o financieros (efectivo, crédito, …).

  En nuestro caso: Para desarrollar la solución un equipo de ingenieros y diseñadores. La elección de nuestros partners es fundamental para localizar nuestros potenciales clientes, empresas consultoras que se encargan de la logística de las votaciones en un territorio.

- **Actividades clave**: si hay recursos clave, también tiene que haber actividades claves. ¿Cuáles son las actividades sin las que tu negocio moriría? ¿Son de producción? ¿De solución a problemas individuales? ¿De una plataforma a través de la que funciona toda la startup?

  En nuestro caso: Venta de la patente de nuestra solución a nuestros clientes locales. Es muy importante la búsqueda de las empresas clave en cada territorio. Realización de un prototipo para la demostración a futuros clientes.

- **Socios clave**: colaboradores y personas que son claves para que el negocio arranque y funcione. ¿Y por qué se buscan estos socios clave? Porque se busca optimizar los recursos (contratar proveedores), reducir riesgos con alianzas estratégicas y adquirir recursos y actividades que no tienes en tu propia startup.

  En nuestro caso: Nuestros clientes son las empresas consultoras que en cada territorio se encargan de la logística y los sistemas de las votaciones. Nuestros partners son asesorías con conocimientos del tejido empresarial y gubernamental de un territorio. Nuestro proveedor principal es el que se encarga de mejorar el producto constantemente.

- **Estructura de costes**: el clásico desglose de los gastos que va a tener tu modelo de negocio. Se incluyen los costes fijos, variables, las economías de escala para reducir costes y todo lo relacionado con el gasto.

  En nuestro caso: Costes de los partners en cada territorio. Costes de la subcontrata encargada de mejorar la solución. Costes salariales de los directivos. Costes generales.

The Business Model Canvas

Socios clave

- Nuestros clientes son las empresas consultoras que en cada territorio se encargan de la logística y los sistemas de las votaciones
- Nuestros partners son asesorías con conocimientos del tejido empresarial y gubernamental de un territorio
- Nuestro proveedor principal es el que se encarga de mejorar el producto constantemente

Actividades clave

- Venta de la patente de nuestra solución a nuestros partners locales
- Es muy importante la búsqueda de las empresas clave en cada territorio.
- Realización de un prototipo para la demostración a futuros clientes

Recursos clave

- Para desarrollar la solución un equipo de ingenieros y diseñadores
- La elección de nuestros partners es fundamental para localizar nuestros potenciales clientes, empresas consultoras que se encargan de la logística de las votaciones en un territorio

Propuestas de valor

- Votación con autenticación mediante carnet que contiene la huella dactilar y detección in situ de la misma.
- Mayor seguridad en la autenticación y reducción radical de la posibilidad de fraude.
- Reducción de la estructura necesaria para llevar a cabo las votaciones

Relaciones con clientes

- Relación de asistencia contínua a nivel técnico de la solución: actualizaciones de la solución como resultado de una mejora continua.
- Contacto contínuo con clientes y partners para mantenernos como referencia para ellos.

Canales

- Los canales serán mediante socios/empresas con conexiones con el gobierno de turno.
- Estos socios conocerán a la perfección las empresas encargadas de la logística de votaciones.

Segmentos de cliente

- Creamos valor para el gobierno de turno pues da mayor veracidad a los resultados electorales.
- Crea valor a las consultoras encargadas de la logística de las votaciones pues reduce costes de estructura necesaria.

Estructura de costes

- Costes de los partners en cada territorio
- Costes de la subcontrata encargada de mejorar la solución
- Costes salariales de los directivos
- Costes generales

Fuentes de ingresos

- Lógicamente nuestros clientes. Pagan para optimizar el coste de las vacaciones y para mejorar la veracidad del proceso.
- Nuestro cliente final son los gobiernos de distintos territorios.

7.2. Plan de Marketing

7.2.1. Análisis D.A.F.O.

Fig. 7-I: Puntos más importantes del análisis DAFO

7.2.2. Valor diferencial

Queremos que las soluciones IoT que vamos a comercializar se diferencien en el mercado por los siguientes aspectos:

- **Prestaciones:** la convergencia de tecnologías para originar nuestras soluciones hace que las prestaciones de nuestros productos sean altísimas, así como su rendimiento.
- **Calidad:** nuestros sistemas se elaboran siguiendo unos estándares muy rigurosos y con las mejores empresas desarrolladoras del mundo.
- **Fiabilidad:** siempre probamos sobradamente nuestras soluciones. Respondemos si el resultado final no es el esperado.
- **Precio:** El coste de nuestro producto será muy inferior al de otras soluciones o a simplemente no hacer nada.

7.2.3. Política de Producto.

La política de producto de nuestra empresa se centrará en idear las mejores soluciones en el mercado, basada en tecnología IoT.

Con ese fin hemos diseñado una estrategia de producto que se apoya en los aspectos que se enumeran a continuación.

- **Tecnología propia:** las sucesivas generaciones de nuestros productos se basarán en I+D que realizamos externamente, pero con tecnología que pasa a nuestro poder.
- **Esfuerzo de I+D:** re-invertiremos una parte considerable de nuestros ingresos en impulsar la investigación y desarrollo de las soluciones para mantenernos siempre en la vanguardia de nuestro sector.
- **Planificación a largo plazo:** actualmente disponemos de un "roadmap" de producto y queremos seguir trabajando siempre con este margen de previsión en el futuro.
- **Especialización:** nos centramos en el desarrollo y comercialización de soluciones de tecnología IoT..
- **Proveedores de calidad:** todos los grupos de investigación y subcontratas en general y desarrollo deberán superar un estricto proceso de selección, para asegurar los estándares de calidad.
- **Soluciones a tiempo:** creemos que el verdadero valor añadido en el suministro de nuestras soluciones, es el control sobre los proyectos de desarrollo para poder finalizar los proyectos a tiempo y con las calidades requeridas.
- **Garantía**: parte de nuestros proyectos es probar nuestras aplicaciones para poder garantizar los resultados. Además damos una asistencia postventa para la correcta utilización del producto.

7.2.4. Política de Servicio y Atención al cliente

Nuestra empresa comercializa a través de una consultora o empresa con el encargo de realizar las votaciones en un determinado territorio, con el cliente final que utiliza nuestro producto. Partiendo de este modelo comercial, toda la actividad de atención al cliente se realizará a través de la empresa que compra directamente nuestro producto, con el cliente final e incluso con nuestro partner. Además ofreceremos un segundo nivel de atención especializada destinado al "profesional" para resolver las consultas técnicas que nos hagan llegar respecto al funcionamiento de nuestro producto.

Esta política de atención al cliente se basará en los siguientes elementos:

- **Información transparente:** desde el momento en que accede a nuestra web o consulta nuestros materiales de marketing, el profesional recibe una información clara sobre nuestro producto, prestaciones, componentes, precios, plazos de entrega, etc.

- **Asesoramiento experto:** el cliente tiene la posibilidad de aclarar cualquier duda por teléfono o e-mail mediante nuestra web.
- **Área de clientes:** está previsto incorporar un área de clientes en la web, en la que el profesional podrá acceder a la información técnica de los productos, realizar pedidos, descargar documentación, etc.
- **Seguimiento:** contactaremos periódicamente con los profesionales que usan nuestros productos para conocer sus impresiones sobre los mismos, dificultades de implementación, etc.
- **Visitas a instalaciones:** nuestros técnicos también visitarán cuando sea posible algunas pruebas realizadas para verificar cómo funciona la implantación de nuestro producto sobre el terreno.
- **Formación:** organizaremos actividades de formación periódicas para que los profesionales del sector puedan conocer mejor nuestros productos y fomentar que los tengan en cuenta en sus proyectos futuros.

En conclusión, se trata de lograr que el cliente se sienta atendido, cuidado y respetado al trabajar con nuestra empresa. Esto nos ayudará a conseguir su confianza y lograr que nuestro producto sea la elección preferente en sus proyectos, al percibir el apoyo que le da nuestra empresa.

7.2.5. Política de Precios.

Los precios de nuestros productos se situarán en una banda media de precios, debido a las prestaciones derivadas de la complejidad de la tecnología utilizada.

7.2.6. Estrategia de publicidad y promoción

Nuestra estrategia de publicidad y promoción se basa en comunicar y promocionar adecuadamente nuestros productos entre dos grandes targets:

7.2.6.1. Profesionales

Son las empresas intermediarias que desean incorporar nuestra solución para incluirlos en los proyectos de sus clientes. Con ese fin, debemos darles a conocer la empresa y convencerles de que nuestro producto es el más adecuado mediante las siguientes herramientas:

- **Página web:** será la principal tarjeta de presentación de nuestra empresa, por lo que vamos a desarrollar un sitio web que dada la internacionalización que se quiere estará en inglés . En esta página web se ofrecerá información detallada sobre la empresa y los productos, con la posibilidad de descargar productos y realizar consultas de soporte técnico. La web será realizada por una empresa especializada e invertiremos un presupuesto anual para su promoción y posicionamiento.
- **Folletos de producto:** todos nuestros productos dispondrán de diversos materiales de marketing, que incluyen un folleto de presentación, una hoja de producto con las especificaciones técnicas, guías de instalación, operación y mantenimiento y otros documentos de tipo técnico. Los elaborará una empresa especializada en documentación comercial.
- **Vídeos de producto:** en la web también se podrá acceder a vídeos divulgativos que describen las prestaciones de nuestros productos. Así mismo, habrá otros vídeos más técnicos que explican paso a paso procedimientos como la instalación y activación del producto, la utilización de las funciones, etc.
- **Buscador de instaladores:** en nuestra web incorporaremos un directorio con las empresas que distribuyen nuestro producto en cada zona geográfica. De esta forma les ayudaremos a generar negocio a partir de las visitas de clientes finales que reciba nuestro sitio y estrecharemos los lazos con ellos.
- **Publicidad en medios especializados:** mantendremos una presencia constante mediante anuncios sobre nuestro producto en las principales publicaciones dirigidas a los profesionales del sector.
- **Ferias:** participaremos como expositores en las principales ferias nacionales e internacionales del sector de IoT y digitalización en general. En todas ellas dispondremos de un stand propio para realizar demostraciones de nuestro producto y establecer contactos.

7.2.6.2. Usuarios finales (Gobiernos de estado, territorio o ciudad)

Aunque no nos dirigimos a los usuarios finales en calidad de compradores de nuestros productos, sí nos interesa mantener un flujo constante de comunicación con ellos para dar a conocer nuestros productos, posicionar la marca y lograr que la identifiquen y exijan a sus ingenierías/consultores. Con ese fin emplearemos las herramientas que se indican a continuación:

- **Publicidad:** tendremos una participación publicitaria periódica en los especiales divulgativos sobre bioenergías y medioambiente que periódicamente realizan los medios impresos y audiovisuales. La idea es que el consumidor vea nuestra marca

siempre que se habla sobre el futuro del medio ambiente, con objeto de ir calando en su mente.

- **Marketing viral:** también tenemos previsto desarrollar algunos vídeos y animaciones de marketing viral centrados en la temática “La votación digital ya está aquí”. Los ofreceremos como descargas promocionales en nuestra web y para generar notoriedad de marca en los portales y redes sociales.
- **Product placement:** también perseguiremos activamente las oportunidades para participar en los showrooms e instalaciones de demostración dirigidas al gran público que se instalan en ferias de muestras, exposiciones sobre IoT. Para ello cedemos gratuitamente nuestro producto y soporte técnico.

Todos estos materiales y acciones serán desarrollados e implementados por una agencia externa de marketing y publicidad contratada a tal efecto.

7.2.7. Estrategia de comunicación y relaciones públicas

Las actividades de comunicación y relaciones públicas serán el otro gran eje de la estrategia de marketing de nuestra empresa. En este caso se trata de lograr presencia en los medios de comunicación sin necesidad de una contraprestación económica directa, con el fin de llegar a nuestros públicos.

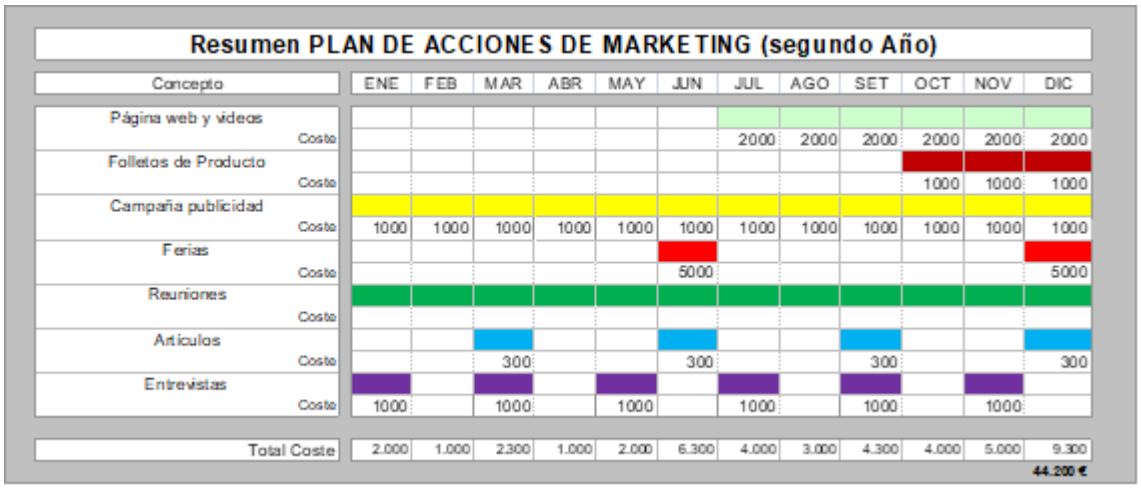
Pensamos que la tecnología IoT es una “historia” que puede resultar atractiva para los medios de comunicación generalistas, por sus implicaciones de transformación en los entornos en los que nos vivimos y trabajamos.
Para ello prevemos utilizar las siguientes herramientas:

- Reuniones con empresas objetivo
- **Notas de prensa:** las novedades de la empresa y la evolución de los proyectos se notificarán mediante comunicados a los medios para obtener cobertura en las secciones de noticias.
- **Reportajes**: ofreceremos a los medios la posibilidad de escribir reportajes sobre la empresa y especialmente sobre los productos IoT asociándose con el futuro.
- **Artículos**: propondremos a los medios generalistas y especializados artículos de divulgación sobre IoT, AI etc.
- **Entrevistas**: queremos posicionar a nuestro CEO como portavoz y persona mediática sobre las ventajas de estas tecnologías.

Estas actividades serán realizadas por una agencia de comunicación especializada en divulgación de nuevas tecnologías que contrataremos a tal efecto. El objetivo es conseguir una presencia continuada de nuestra marca en los medios de comunicación siempre que traten sobre estos temas.

7.2.8. Plan de Acción de Marketing

Durante el segundo año, una vez la empresa tenga suficiente financiación, se realizará un plan de marketing para comenzar a vender la empresa y que ésta sea conocida.



Resumen PLAN DE ACCIONES DE MARKETING (segundo Año)

Concepto	ENE	FEB	MAR	ABR	MAY	JUN	JUL	AGO	SET	OCT	NOV	DIC
Página web y videos												
Coste							2000	2000	2000	2000	2000	2000
Folletos de Producto												
Coste										1000	1000	1000
Campaña publicidad												
Coste	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
Ferias												
Coste						5000						5000
Reuniones												
Coste												
Articulos												
Coste			300			300			300			300
Entrevistas												
Coste	1000		1000		1000		1000		1000		1000	
Total Coste	2.000	1.000	2.300	1.000	2.000	6.300	4.000	3.000	4.300	4.000	5.000	9.300

44.200 €

7.2.9. Plan de Establecimiento.

Desde hace un año, los tres promotores de la empresa han iniciado los preparativos para la puesta en marcha de su actividad prevista en el plazo de seis meses. Los pasos que se han dado hasta la fecha incluyen:

- Arrendamiento de las instalaciones de la empresa en centro científico-tecnológico o incubadora tecnológica.
- Acondicionamiento básico de los espacios.
- Selección de los proveedores clave.
- Búsqueda de las aplicaciones clave a desarrollar.
- Elaboración del presente plan de negocio.
- Búsqueda de financiación e inversores.

7.3. Plan de Ventas

7.3.1. Estrategia de Ventas.

Como hemos comentado anteriormente los canales de venta serán tres:

- **Partners:** son las empresas intermediarias que conocen los mecanismos de acercamiento y posicionamiento a las empresas consultoras encargadas de votaciones.

- **Consultoras:** Se tratan de las empresas encargadas de las votaciones y quienes prescribirán nuestra solución.

- **Gobiernos:** son los promotores de elecciones, y tienen gran influencia sobre la empresa consultora de turno.

Nuestra estrategia de ventas varía en función del mercado al que nos dirijamos, ya que está diseñada para aprovechar al máximo las oportunidades disponibles actualmente y asegurar un flujo de ingresos desde el principio. A continuación detallamos la estrategia de ventas en cada región:

- **España y Unión Europea**: por proximidad y afinidad cultural será el mercado inicial de nuestra empresa y el que vamos a desarrollar más en profundidad. Para ello queremos establecer relaciones sólidas con gobiernos selectos que nos confiera una cobertura nacional. Debido a su tamaño, la construcción de una relación privilegiada con estos distribuidores tomará tiempo. Pero estamos seguros de que la calidad y funcionalidad de nuestra solución nos ayudará a penetrar progresivamente en los principales mercados. Si constatamos que algún país funciona especialmente bien, tenemos previsto abrir una oficina comercial de apoyo.

- **Estados Unidos y Canadá:** Es un mercado global más apetitoso por la gran población y, el poder adquisitivo de sus clases medias y altas, Sin embargo, reconocemos que es un mercado difícil de penetrar debido a la fuerte competencia local. Por lo tanto, negociaremos un acuerdo para crear una “joint venture” con una empresa que comercializa soluciones IoT. A través de una marca local específica vinculada a este socio, esperamos que la red comercial de la cual dispone nos ayude a generar un volumen notable de ventas.

- **Latinoamérica:** se trata de mercados con niveles de desarrollo desiguales, aunque existen oportunidades en países como Brasil o México. En esta región nuestro departamento comercial se encargará de ir seleccionando los mejores distribuidores para cada país con objeto de tener presencia en la región. Son países con cierta tradición de votación digital.
- **Resto del mundo:** en el periodo inicial de 5 años que contempla este plan de negocio no tenemos previsto entrar en otros mercados internacionales. La única excepción será Australia y Nueva Zelanda.

7.3.2. El personal de Ventas.

Hay que destacar que nuestras soluciones no son productos de gran producción y consumo sino productos B2B de utilización más puntual y donde no se utilizan grandes cantidades de una sustancia. Nuestra empresa sólo dispondrá de una personas para realizar toda la tarea: el CEO y un apoyo de Director de Desarrollo de Negocio. Los contactos comerciales de los dos primeros años, donde nos abriremos al mercado de la unión europea, se realizarán por estas dos personas.

A partir del primer año la empresa se abrirá al mercado norteamericano, Latinoamérica y Australia y se contratará una segunda persona de apoyo. Esta expansión internacional no se ha considerado en los resultados numéricos aunque como se verá más adelante será obligatoria para no bajar rendimientos económicos de la empresa y por tanto para la viabilidad de la misma.

La gestión de la fuerza de ventas se basará en los siguientes aspectos básicos:

- **Trabajo por objetivos:** la remuneración de los miembros del departamento comercial tendrá una parte fija y un componente variable, ligado a la consecución de los objetivos establecidos.
- **Calidad de la venta:** en todo momento el equipo de ventas estará obligado a seguir los procedimientos determinados por la empresa y a mantener los estándares de calidad. Con este fin, permanentemente se realizará una evaluación de la calidad del servicio ofrecido mediante elementos como encuestas a los clientes, revisiones comerciales y otros.
- **Formación continua:** todos los miembros de la fuerza de ventas estarán obligados a asistir a las formaciones que impartirá la empresa. Además, se les animará y apoyará para que amplíen su formación en aspectos que contribuyan a la mejora de su preparación para el trabajo que desempeñan en la empresa.

- **Fomento de la competitividad**: mediante la remuneración variable y los premios que se establecerán para recompensar a los vendedores que demuestren un mayor rendimiento (tanto en resultados como en servicio al cliente), se quiere fomentar una competencia sana entre los miembros de este departamento.

7.3.3. Herramientas de venta

La fuerza de ventas dispondrá para su actividad de una amplia gama de instrumentos comerciales desarrollados por la empresa para apoyar su actividad. Estas herramientas de venta incluyen:

- **Documentación de producto:** folletos, hojas de producto, documentos técnicos, vídeos y otros materiales pensados para facilitar la exposición de las bondades de nuestras soluciones a los posibles clientes.
- **Documentación comercial:** adicionalmente el vendedor dispondrá de documentos específicos para su tarea comercial como argumentarios de producto, respuesta a preguntas frecuentes, fichas para cualificar a los clientes, etc.
- **Showroom:** en las instalaciones de la empresa se habilitará una sala con de video que podrán usar indistintamente los departamentos de I+D, Control de Calidad y Ventas. Estos videos reproducirán un entorno real de aplicación del producto.
- **Merchandising:** además de la documentación comercial, produciremos un pequeño surtido de objetos promocionales para los clientes como bolígrafos, calendarios, tacos de notas, etc.

7.3.4. Barreras de venta

Por nuestra experiencia y conocimiento del mercado, somos conscientes de los obstáculos que cualquier empresa encuentra en su actividad comercial, especialmente si se trata de una nueva compañía. Por ese motivo, hemos previsto las principales “barreras de venta” que nuestros vendedores se pueden encontrar en su actividad y les daremos formación específica para superarlas.

Las principales barreras de venta a las que nos enfrentamos son:

- **Desconocimiento de la marca:** ante la desconfianza del cliente frente a una marca de la que no ha oído hablar o que lleva poco tiempo en el mercado, nos referiremos a la experiencia del equipo fundador y a las fortalezas de la tecnología que ofrecemos.

- **Desconfianza en la calidad:** así mismo, puede suceder que el cliente no quiera arriesgarse a probar un producto nuevo en un proyecto importante, así que el vendedor deberá hacer un énfasis especial en la garantía de solución que ofrecemos.
- **Movimientos de los competidores:** es previsible que en los grandes clientes y proyectos nos enfrentemos a los principales competidores, para lo cual nuestros vendedores deberán saber comunicar mejor las ventajas de nuestra tecnología.
- **Precio del producto:** el precio de nuestro producto se encuentra en la banda media puede ser percibido como “algo caro” por el cliente si no sabemos comunicar adecuadamente todas las prestaciones y beneficios que ofrece.

A medida que se detecten barreras de venta más específicas o puntuales en el trato con los clientes, se desarrollarán estrategias para superarlas.

7.4. Aspectos Legales y Societarios

7.4.1. La sociedad.

A continuación se detallan los principales datos de la sociedad:

- **Denominación:** Digital Polls S.L.
- **Domicilio social:** C/ Pedro Pérez (Barcelona, España)
- **Capital social:** 100.000 euros
- **Objeto:** desarrollo y comercialización de soluciones IOT.
- **Fecha de constitución:** 1-7-2018

7.4.2. Estatutos y acuerdos de los socios

La gestión y administración de la empresa se encarga a un órgano social. Este órgano directivo está formado por la Junta General y por los administradores.

7.4.2.1. Junta general

La junta general es el órgano de deliberación y de decisión. Los asuntos que puede tratar la Junta son censuras de la gestión, la aprobación de las cuentas anuales, el nombramiento y destitución de los administradores y la modificación de los estatutos. La convocatoria de la

Junta General corresponde a los administradores, que lo harán dentro de los seis primeros meses de cada ejercicio social. La finalidad es censurar la gestión social, aprobar, en su caso, las cuentas del ejercicio anterior y resolver sobre la aplicación del resultado. Esta convocatoria es tan importante que de no hacerse podría realizarla el Juez de 1ª Instancia del domicilio social a instancia de cualquier socio. También lo pueden hacer siempre que lo consideren necesario o en los plazos que determinen los estatutos. Los administradores deberán convocar Junta General cuando así lo soliciten los socios que supongan un 5% del capital social.

Los administradores tienen la obligación de dar publicidad a la convocatoria de Junta, mediante anuncio publicado en el BORME y en uno de los diarios de mayor circulación en el término municipal en que esté situado el domicilio social. Los estatutos podrán establecer, en sustitución del sistema anterior, que la convocatoria se realice mediante anuncio publicado en un determinado diario de circulación en el término municipal en que esté situado el domicilio social, o por cualquier procedimiento de comunicación, individual y escrita, que asegure la recepción del anuncio por todos los socios en el domicilio designado al efecto o en el que conste en el Libro registro de socios.En el caso de socios que residan en el extranjero, los estatutos podrán prever que sólo serán individualmente convocados si hubieran designado un lugar del territorio nacional para notificaciones. Entre convocatoria y celebración de la Junta General debe haber una antelación mínima de 15 días.

Junta Universal: La Junta General queda válidamente constituida con carácter de "Universal". Es decir, que estando presente todo el capital se decida por unanimidad la celebración de la reunión y el orden del día de la misma.

7.4.2.2. Administradores

La administración se confiará a un Consejo de Administración (tres administradores). El Consejo de Administración podrá delegar todas o algunas de sus facultades en uno o varios de sus miembros, que tomará la denominación de Consejero Delegado, debiéndose determinar el modo y limitaciones en que se ejercerán esas facultades.

Los administradores deben cumplir una serie de requisitos:

- No podrán dedicarse, por cuenta ajena, al mismo género de comercio que constituya el objeto de la sociedad, salvo aprobación de la Junta General.
- Ejercerán el cargo durante el período de tiempo que se señale en los estatutos (que podrá ser indefinido) y podrán ser destituidos en cualquier momento por la Junta General, incluso aunque este punto no estuviese incluido en el orden del día.
- Para llevar a cabo las cuentas anuales deberán seguir las normas de las sociedades promiscuas.

- No es necesario que sean socios de la empresa, aunque los estatutos podrán establecer lo contrario, incluso otra serie de requisitos.

7.4.3. Derechos de los socios

Cada uno de los tiene los siguientes derechos:

- Derecho a participar en el reparto de beneficios y en el patrimonio de la sociedad en caso de liquidación.
- Derecho de tanteo en la adquisición de las participaciones de los socios salientes.
- Derecho a participar en las decisiones sociales y a ser elegidos como administradores.
- Derecho de información en los períodos establecidos en las escrituras.
- Derecho de obtener información sobre los datos contables de la Sociedad.

7.4.4. Obligaciones legales.

Estamos en proceso de culminar la tramitación de todos los requisitos necesarios para ejercer la actividad de fabricación en España:

1. Hemos constituido la empresa (Digital Polls s.l.)

2. Hemos formalizado el alta de la sociedad en el Impuesto de Actividades Económicas (IAE).

3. Hemos solicitado la licencia de apertura y actividad para la nave que acogerá las oficinas de la empresa.

4. Tendremos a todo el personal contratado y dado de alta en la Seguridad Social, como requisito previo a su incorporación.

5. Impartimos al personal toda la formación requerida en cuanto a la Ley de Prevención de Riesgos Laborales y otras normativas.

6. Estamos en trámites para contratar los seguros obligatorios y opcionales que den cobertura debida a nuestra actividad.

7. Estamos tramitando las patentes de nuestros productos y tecnologías en España y en los diversos mercados en que operaremos.

Además, adoptaremos las medidas necesarias para cumplir todas las normativas y regulaciones que afectan a nuestra actividad.

7.4.5. Licencias y derechos

La actual sociedad ha registrado las siguientes marcas y dominios:
Digital Polls S.L.: en el Registro Mercantil de Barcelona, Hoja xxx, Folio xx, Tomo xxx, Libro xxx, Sección xx, CIF: X-xxxxxxxx.
Dominios de Internet:

www.digitalpolls.com

7.5 Plan económico-financiero

7.5.1. Confidencialidad del documento

La información que contiene este documento es confidencial y propiedad de nuestra empresa, que se la ha facilitado exclusivamente a usted para su valoración personal. No está autorizado a divulgar, copiar o reproducir este documento, o a comentar la información que contiene con ninguna otra persona, a menos que obtenga previamente un permiso por escrito. Al aceptar la entrega de este documento, usted acepta mantener la confidencialidad de los contenidos y no revelarlos a terceras partes.

7.5.2. Proyecciones y previsiones

Este documento contiene proyecciones y previsiones de futuro que están basadas en suposiciones respecto a eventos del futuro que son inciertos y subjetivos. Nuestra empresa no garantiza en ningún caso que se obtendrán los resultados proyectados, dado que estas proyecciones y previsiones están basadas en suposiciones y estimaciones subjetivas y los resultados finales podrían diferir sustancialmente de los esperados.
Antes de invertir o decidir cualquier acción a partir de la información contenida en este documento, debe realizar sus propias investigaciones sobre la empresa y las previsiones realizadas con objeto de determinar los riesgos y consecuencias de su decisión.

7.5.3. Hipótesis

Los cálculos del presente plan se fundamentan en estas premisas:

- Durante los últimos 6 meses previos al año 0 (2019), nos dedicamos a rehabilitar nuestras oficinas y a buscar proveedores que mejor se adapten a nuestras necesidades y a buscar aplicaciones susceptibles de ser desarrollables.
- En los planes de ventas del capítulo de ventas se reflejan, ventas de unidades y precio por unidad.
- Hemos supuesto el mismo sueldo para las cinco personas de 60.000 €uros brutos anuales durante todos los años.
- Hemos despreciado la inflación.
- El segundo año hay una campaña de marketing, antes de comercializar las siguientes soluciones.

7.5.4. Descripción económica

ESTRUCTURA DE COSTES E INGRESOS

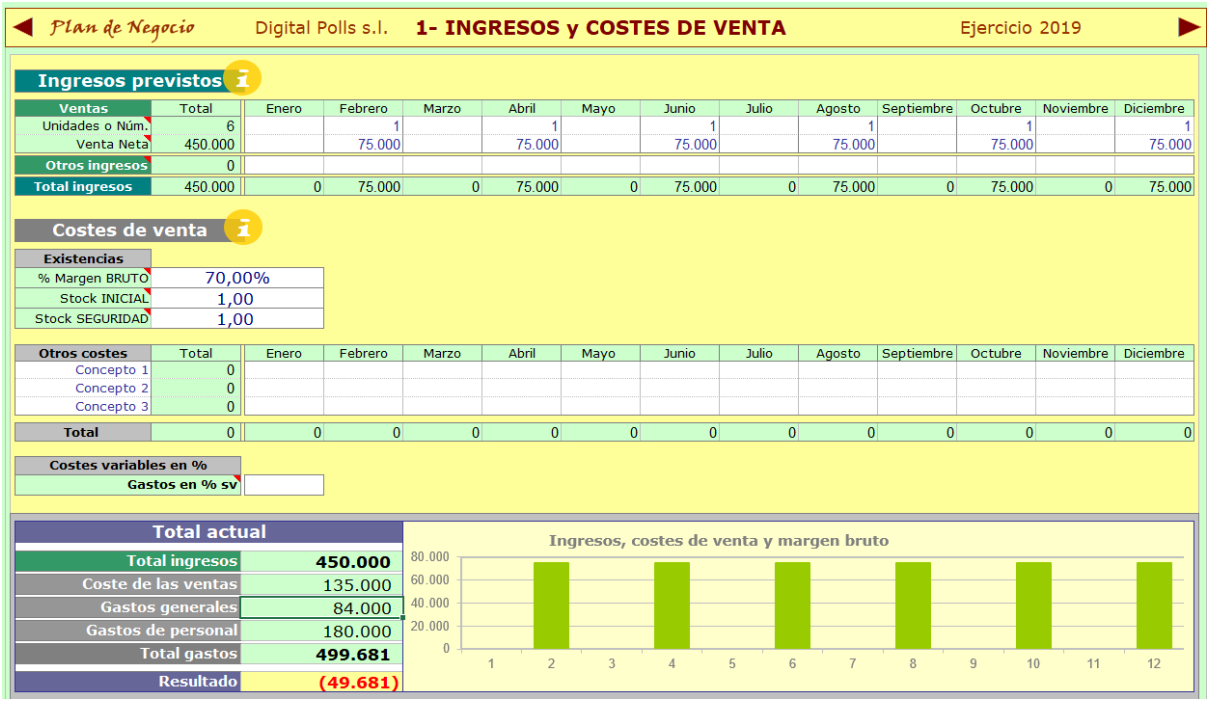
Financiación:	
Capital	100.000 €
Préstamo a 4 años	65.000 €

Puesta en marcha (6 meses)	
mano obra directa	55.000 €
mano obra indirecta	15.000 €
gastos generales	10.000 €
materia prima	5.000 €
gastos por administración	5.000 €
imprevistos	10.000 €
Total	100.000 €

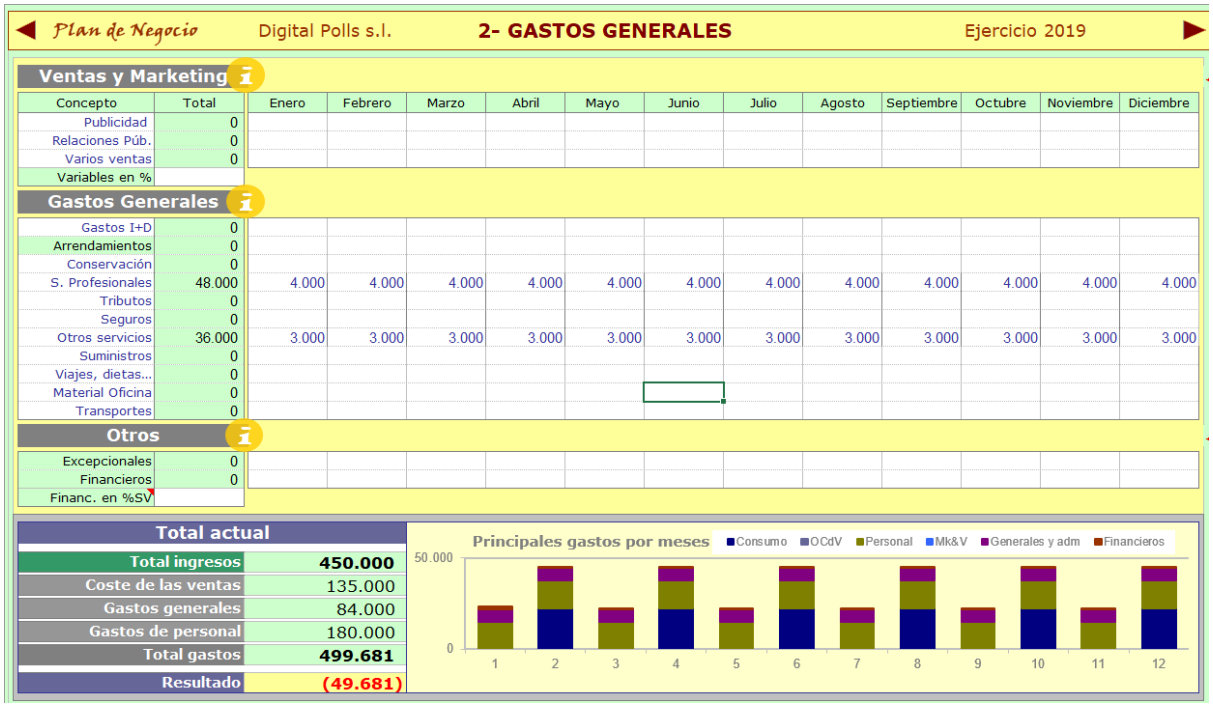
Ingresos proyectados	
nº clientes mensuales	0,5
ingresos por cliente	75.000 €
ingresos mensuales	37.500 €
TOTAL INGRESOS	450.000 €

Gastos mensuales	
personal (3 directivos)	15.000 €
Subcontratas	4.000 €
gastos generales*	3.000 €
coste de venta (partners)	11.250 €
Total	33.250 €
TOTAL GASTOS	399.000 €

* alquiler oficina, comunicaciones, utilities etc



Los ingresos vienen de las ventas. Conseguimos una nueva venta cada dos meses. La facturación por permiso de uso de nuestra patente es de 75.000 €uros para cada votación. Un 30% de la facturación es el coste de nuestro socio-conseguidor.



Plan de Negocio — Digital Polls s.l. — 2- GASTOS GENERALES — Ejercicio 2019

Ventas y Marketing

Concepto	Total	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
Publicidad	0												
Relaciones Púb.	0												
Varios ventas	0												
Variables en %													

Gastos Generales

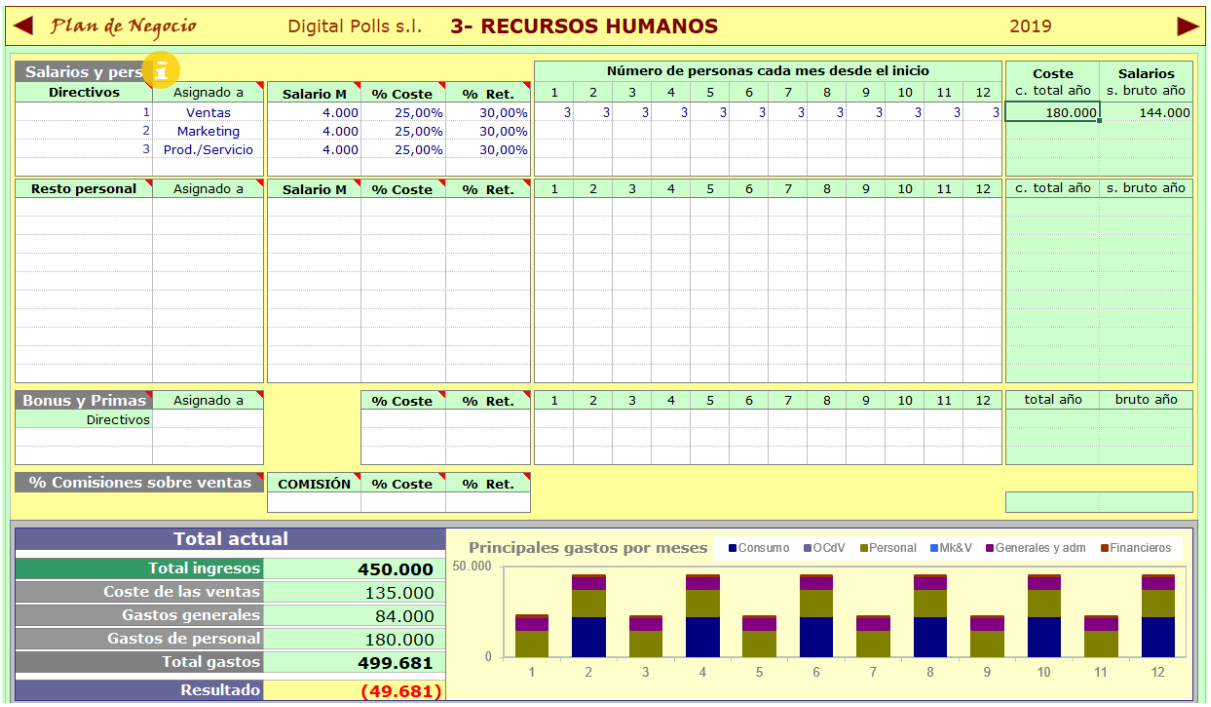
Gastos I+D	0												
Arrendamientos	0												
Conservación	0												
S. Profesionales	48.000	4.000	4.000	4.000	4.000	4.000	4.000	4.000	4.000	4.000	4.000	4.000	4.000
Tributos	0												
Seguros	0												
Otros servicios	36.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
Suministros	0												
Viajes, dietas...	0												
Material Oficina	0												
Transportes	0												

Otros

Excepcionales	0												
Financieros	0												
Financ. en %SV													

Total actual	
Total ingresos	450.000
Coste de las ventas	135.000
Gastos generales	84.000
Gastos de personal	180.000
Total gastos	499.681
Resultado	(49.681)

Los gastos generales se derivan de los servicios profesionales de la subcontrata que mejora nuestra solución de manera continua y 3.000 €uros de alquiler oficinas etc.



Plan de Negocio — Digital Polls s.l. — 3- RECURSOS HUMANOS — 2019

Salarios y pers

Directivos	Asignado a	Salario M	% Coste	% Ret.	Número de personas cada mes desde el inicio 1	2	3	4	5	6	7	8	9	10	11	12	Coste c. total año	Salarios s. bruto año
1	Ventas	4.000	25,00%	30,00%	3	3	3	3	3	3	3	3	3	3	3	3	180.000	144.000
2	Marketing	4.000	25,00%	30,00%														
3	Prod./Servicio	4.000	25,00%	30,00%														

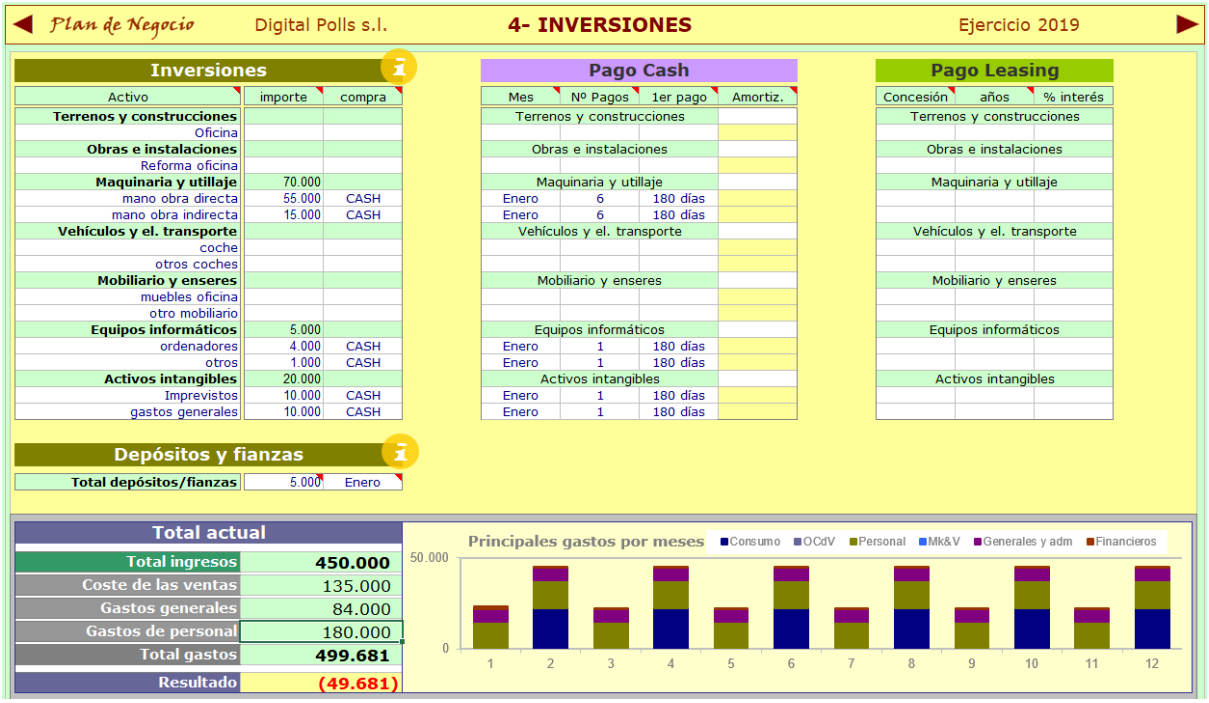
Resto personal	Asignado a	Salario M	% Coste	% Ret.	1	2	3	4	5	6	7	8	9	10	11	12	c. total año	s. bruto año

Bonus y Primas	Asignado a	% Coste	% Ret.	1	2	3	4	5	6	7	8	9	10	11	12	total año	bruto año
Directivos																	

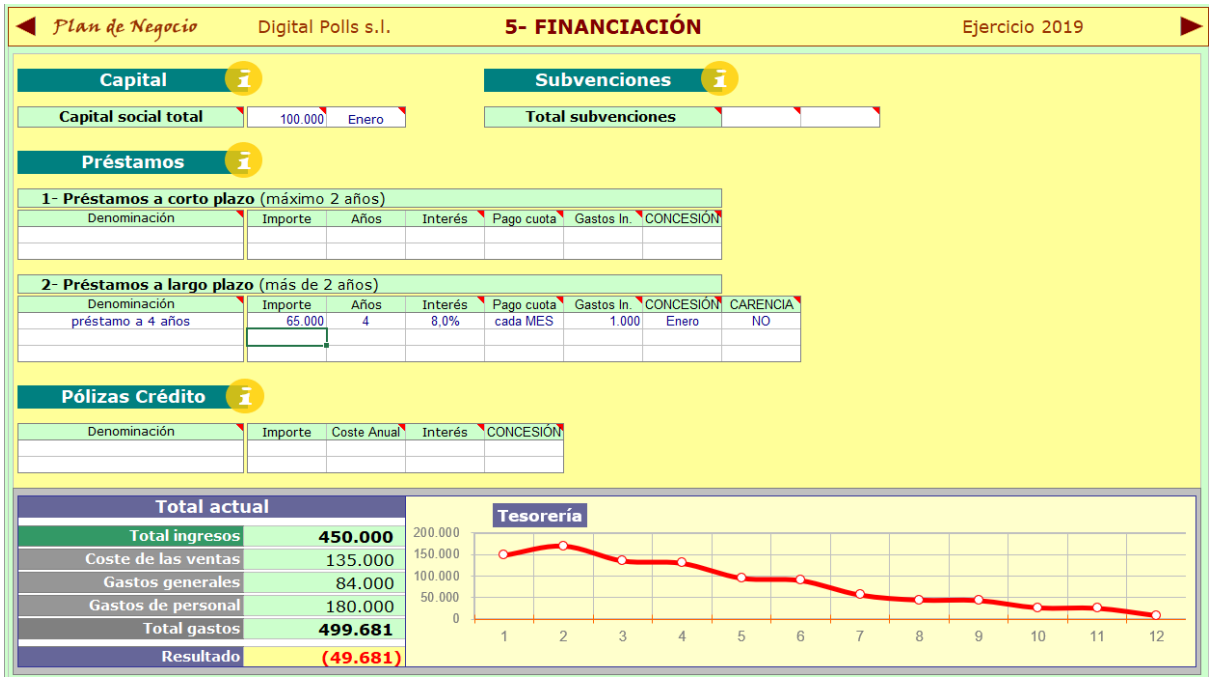
% Comisiones sobre ventas	COMISIÓN	% Coste	% Ret.

Total actual	
Total ingresos	450.000
Coste de las ventas	135.000
Gastos generales	84.000
Gastos de personal	180.000
Total gastos	499.681
Resultado	(49.681)

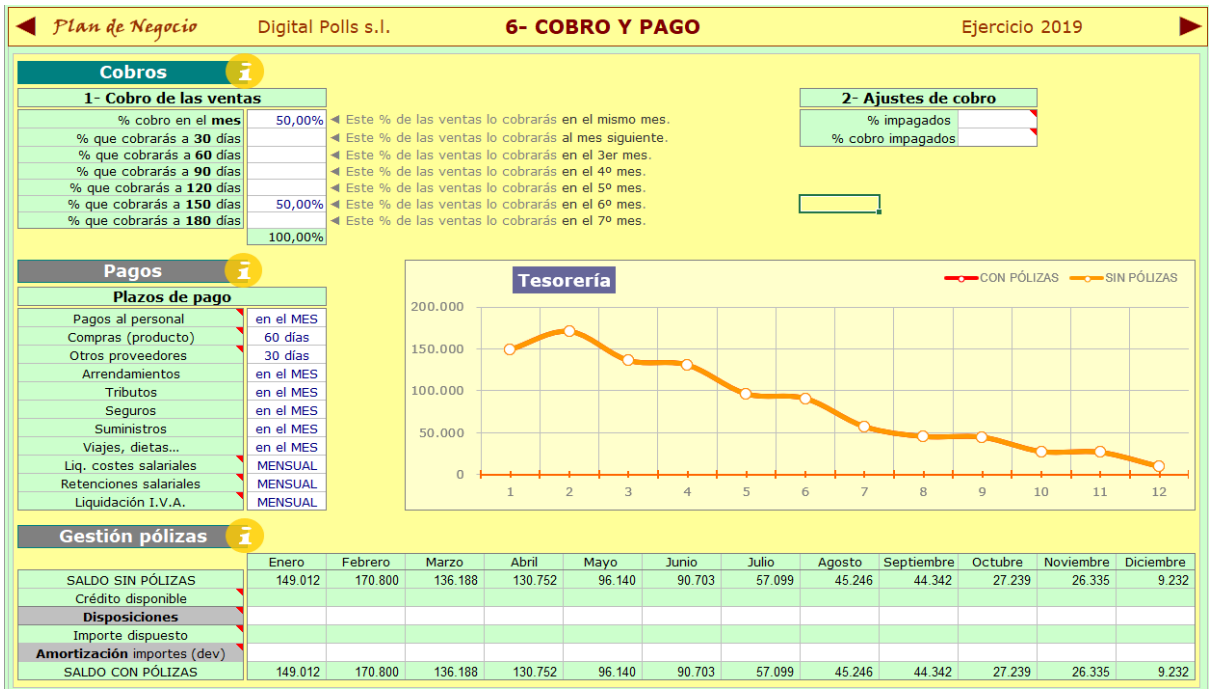
El gasto de recursos humanos es del salario de los tres socios a razón de 60.000 €uros por persona coste empresa.



La inversión es el coste de llevar la idea de ese estatus al de una solución real, patentable y que suponga un valor añadido al proceso de votación.



La financiación proviene del capital social y de un préstamo a 4 años.



Plan de Negocio — Digital Polls s.l. — **6- COBRO Y PAGO** — Ejercicio 2019

Cobros

1- Cobro de las ventas

% cobro en el **mes**	50,00%	Este % de las ventas lo cobrarás en el mismo mes.
% que cobrarás a **30** días		Este % de las ventas lo cobrarás al mes siguiente.
% que cobrarás a **60** días		Este % de las ventas lo cobrarás en el 3er mes.
% que cobrarás a **90** días		Este % de las ventas lo cobrarás en el 4º mes.
% que cobrarás a **120** días		Este % de las ventas lo cobrarás en el 5º mes.
% que cobrarás a **150** días	50,00%	Este % de las ventas lo cobrarás en el 6º mes.
% que cobrarás a **180** días		Este % de las ventas lo cobrarás en el 7º mes.
	100,00%	

2- Ajustes de cobro

% impagados	
% cobro impagados	

Pagos

Plazos de pago

Pagos al personal	en el MES
Compras (producto)	60 días
Otros proveedores	30 días
Arrendamientos	en el MES
Tributos	en el MES
Seguros	en el MES
Suministros	en el MES
Viajes, dietas...	en el MES
Liq. costes salariales	MENSUAL
Retenciones salariales	MENSUAL
Liquidación I.V.A.	MENSUAL

Gestión pólizas

	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
SALDO SIN PÓLIZAS	149.012	170.800	136.188	130.752	96.140	90.703	57.099	45.246	44.342	27.239	26.335	9.232
Crédito disponible												
Disposiciones												
Importe dispuesto												
Amortización importes (dev)												
SALDO CON PÓLIZAS	149.012	170.800	136.188	130.752	96.140	90.703	57.099	45.246	44.342	27.239	26.335	9.232

Cobramos un 50% al mes y la otra mitad a 150 días, después de las votaciones.

7.5.5 Cuenta de pérdidas y ganancias

Plan de Negocio — CUENTA DE PÉRDIDAS Y GANANCIAS — Ejercicio 2019

Digital Polls s.l.

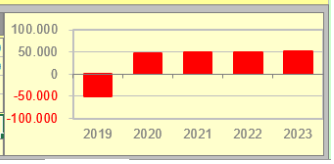
Pérdidas y Ganancias Previstas — Ejercicio 2019

INGRESOS	Total	%	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
Venta neta total	450.000	100,0%	0	75.000	0	75.000	0	75.000	0	75.000	0	75.000	0	75.000
Otros ingresos	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Total Ingresos	450.000		0	75.000	0	75.000	0	75.000	0	75.000	0	75.000	0	75.000
GASTOS	**Total**	**%**	**Enero**	**Febrero**	**Marzo**	**Abril**	**Mayo**	**Junio**	**Julio**	**Agosto**	**Septiembre**	**Octubre**	**Noviembre**	**Diciembre**
Consumo	135.000	30,0%	0	22.500	0	22.500	0	22.500	0	22.500	0	22.500	0	22.500
Costes de venta	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Personal	180.000	40,0%	15.000	15.000	15.000	15.000	15.000	15.000	15.000	15.000	15.000	15.000	15.000	15.000
comisiones	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
salarios previos	0	0,0%	0											
producción/servicio	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
marketing/ventas	180.000	40,0%	15.000	15.000	15.000	15.000	15.000	15.000	15.000	15.000	15.000	15.000	15.000	15.000
administración/DG	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Marketing y ventas	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Publicidad	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Relaciones Púb.	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Varios ventas	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
variables	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Generales y adm	84.000	18,7%	7.000	7.000	7.000	7.000	7.000	7.000	7.000	7.000	7.000	7.000	7.000	7.000
Gastos I+D	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Arrendamientos	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Conservación	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
S. Profesionales	48.000	10,7%	4.000	4.000	4.000	4.000	4.000	4.000	4.000	4.000	4.000	4.000	4.000	4.000
Tributos	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Seguros	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Otros servicios	36.000	8,0%	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
Suministros	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Viajes, dietas...	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Material Oficina	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Transportes	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Excepcionales	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Insolvencias	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Total gastos	399.000	88,7%	22.000	44.500	22.000	44.500	22.000	44.500	22.000	44.500	22.000	44.500	22.000	44.500
Amortizaciones	95.000	21,1%	7.917	7.917	7.917	7.917	7.917	7.917	7.917	7.917	7.917	7.917	7.917	7.917
Res. Explotación	-44.000	-9,8%	-29.917	22.583	-29.917	22.583	-29.917	22.583	-29.917	22.583	-29.917	22.583	-29.917	22.583
Res. Financiero	-5.681	-1,3%	-1.390	-390	-390	-390	-390	-390	-390	-390	-390	-390	-390	-390
Gastos Financieros	5.681	1,3%	1.390	390	390	390	390	390	390	390	390	390	390	390
Intereses	4.681		390	390	390	390	390	390	390	390	390	390	390	390
Otros gastos financ.	1.000		1.000	0	0	0	0	0	0	0	0	0	0	0
RESULTADO	**Total**	**%**	**Enero**	**Febrero**	**Marzo**	**Abril**	**Mayo**	**Junio**	**Julio**	**Agosto**	**Septiembre**	**Octubre**	**Noviembre**	**Diciembre**
antes de impuestos	-49.681	-11,0%	-31.307	22.193	-30.307	22.193	-30.307	22.193	-30.307	22.193	-30.307	22.193	-30.307	22.193
impuesto s/beneficio	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
RESULTADO NETO	-49.681	-11,0%	-31.307	22.193	-30.307	22.193	-30.307	22.193	-30.307	22.193	-30.307	22.193	-30.307	22.193
			-31.307	-9.114	-39.420	-17.227	-47.534	-25.341	-55.647	-33.454	-63.761	-41.568	-71.874	-49.681

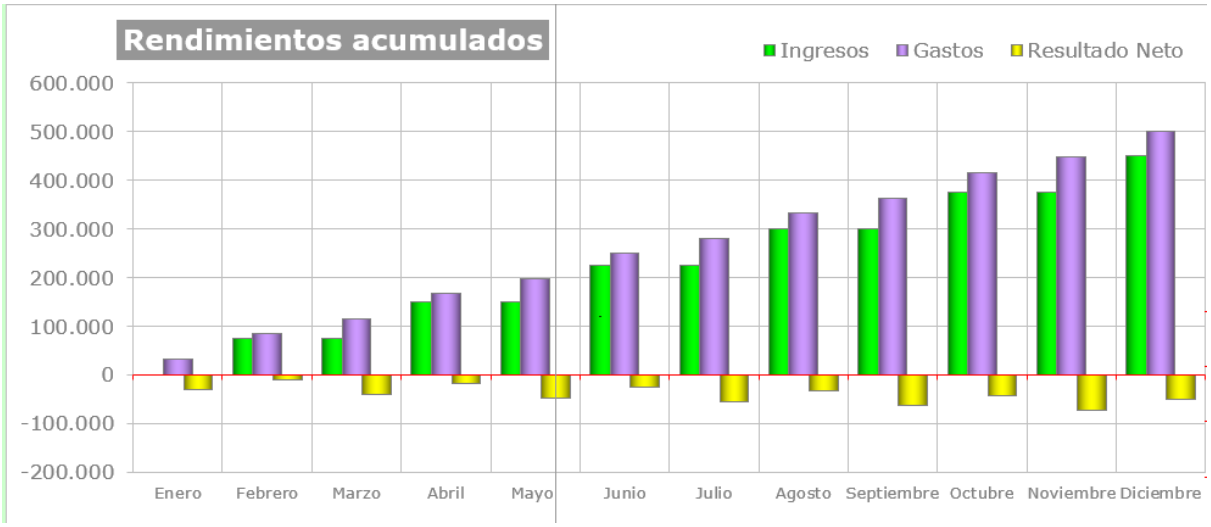
Plan de Negocio — Digital Polls s.l. — **7- PREVISIONES A CINCO AÑOS**

Ingresos	2019	2020	2021	2022	2023	2020	2021	2022	2023
Venta Neta	450.000	450.000	450.000	450.000	450.000				
Otros ingresos									
Gastos	**2019**	**2020**	**2021**	**2022**	**2023**	**2020**	**2021**	**2022**	**2023**
Coste Ventas									
Publicidad									
Relaciones Púb.									
Varios ventas									
Gastos I+D									
Arrendamientos									
Conservación									
S. Profesionales	48.000	48.000	48.000	48.000	48.000				
Tributos									
Seguros									
Otros servicios	36.000	36.000	36.000	36.000	36.000				
Suministros									
Viajes, dietas...									
Material Oficina									
Transportes									
Excepcionales									
Financieros (no %)									
Personal	**2019**	**2020**	**2021**	**2022**	**2023**	**2020**	**2021**	**2022**	**2023**
Pers. prod/servicio									
Marketing y ventas	180.000	180.000	180.000	180.000	180.000				
Admin. y dirección									

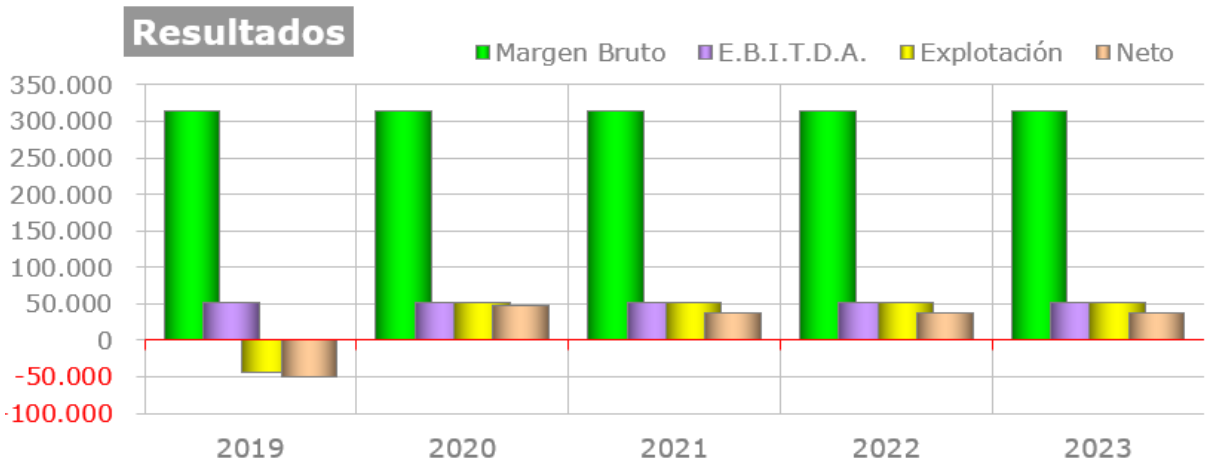
Resultados	2019	2020	2021	2022	2023
Ingresos netos	450.000	450.000	450.000	450.000	450.000
Gastos Operativos	399.000	399.000	399.000	399.000	399.000
Amortizaciones	95.000				
Gastos Financieros	5.681	3.489	2.198	800	
Resultado	**-49.681**	**47.511**	**48.802**	**50.200**	**51.000**
margen s/ventas	-11,04%	10,56%	10,84%	11,16%	11,33%



Como vemos en el análisis de pérdidas y ganancias, los primeros años en los que todavía no tenemos las aplicaciones dispuestas para ser comercializadas, y tenemos todos los gastos de sueldos y pagos a proveedores, los resultados son malos mientras que conforme tenemos más soluciones y más aún cuando empezamos a vender las primeras los resultados cambian ostensiblemente.



La mayor parte de los gastos vienen del coste de nuestros partners. Después son los de personal, a pesar de que los sueldos que hemos puesto son modestos. Los gastos cogen una inercia casi igual muy pronto y conforme suben la ventas se incrementan los beneficios.



El primer año el resultado es negativo mientras que a partir del segundo entramos en ganancias.

7.5.6. Cuenta de Tesorería

Plan de Negocio	PRESUPUESTO DE TESORERÍA	Ejercicio 2019

Digital Polls s.l.

Presupuesto de Tesorería — Ejercicio **2019**

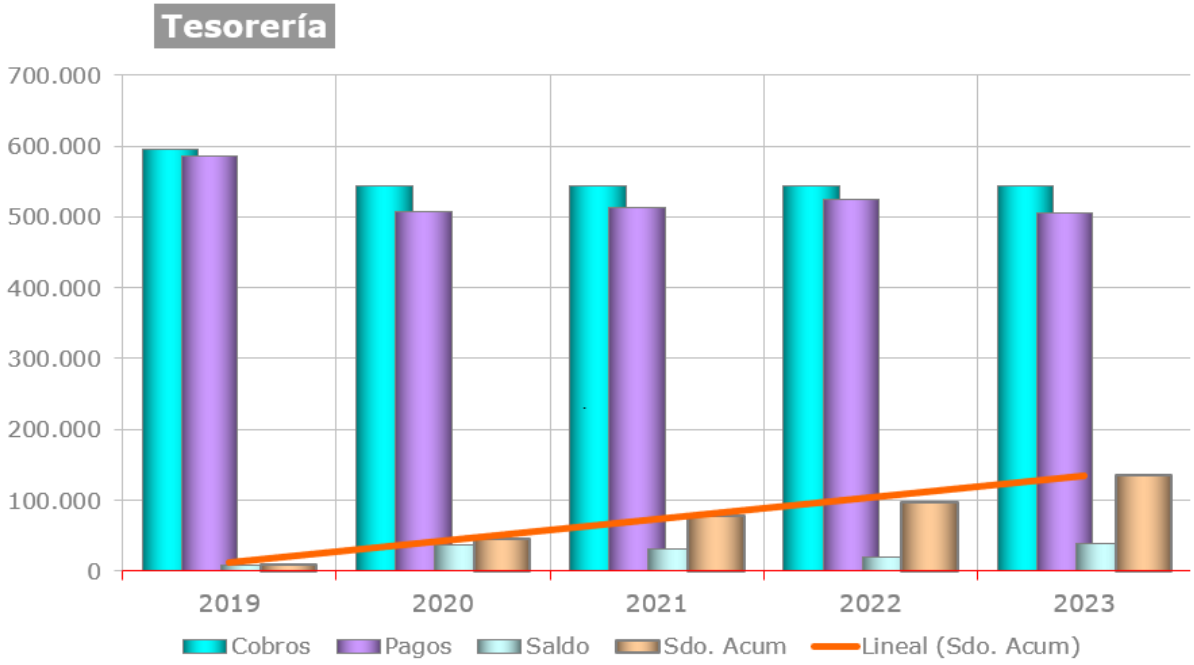
CASH FLOW	Total	%	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre
Saldo **acumulado inicio** del mes				149.012	170.800	136.188	130.752	96.140	90.703	57.099	45.246	44.342	27.239	26.335
TOTAL COBROS	**594.795**	%	**165.000**	**46.845**	**0**	**46.845**	**0**	**46.845**	**45.375**	**54.545**	**45.375**	**49.295**	**45.375**	**49.295**
Cobros por ventas	**408.375**	68,7%	**0**	**45.375**	**0**	**45.375**	**0**	**45.375**	**45.375**	**45.375**	**45.375**	**45.375**	**45.375**	**45.375**
Ventas	408.375	100,0%	0	45.375	0	45.375	0	45.375	45.375	45.375	45.375	45.375	45.375	45.375
% impagados	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
% cobro impagados	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Otros ingresos	**186.420**	31,3%	**165.000**	**1.470**	**0**	**1.470**	**0**	**1.470**	**0**	**9.170**	**0**	**3.920**	**0**	**3.920**
Otros ingresos	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Socios	100.000	16,8%	100.000	0	0	0	0	0	0	0	0	0	0	0
Préstamos	65.000	10,9%	65.000	0	0	0	0	0	0	0	0	0	0	0
Ingresos financieros	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Subvenciones	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
I.V.A. y otros	21.420	3,6%	0	1.470	0	1.470	0	1.470	0	9.170	0	3.920	0	3.920
TOTAL PAGOS	**585.563**	98%	**15.988**	**25.057**	**34.612**	**52.282**	**34.612**	**52.282**	**78.979**	**66.399**	**46.279**	**66.399**	**46.279**	**66.399**
Pagos operativos	**363.096**	62,0%	**8.401**	**19.870**	**19.870**	**47.095**	**19.870**	**47.095**	**19.870**	**47.095**	**19.870**	**47.095**	**19.870**	**47.095**
Salarios e incentivos	100.800	17,2%	8.400	8.400	8.400	8.400	8.400	8.400	8.400	8.400	8.400	8.400	8.400	8.400
Comisiones	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Compras	136.126	23,2%	1	0	0	27.225	0	27.225	0	27.225	0	27.225	0	27.225
Otros costes	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Variables prod/servicio	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Publicidad	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Relaciones Púb.	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Gastos de Ventas	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Variables de Ventas	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Gastos I+D	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Arrendamientos	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Conservación	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
S. Profesionales	53.240	9,1%	0	4.840	4.840	4.840	4.840	4.840	4.840	4.840	4.840	4.840	4.840	4.840
Tributos	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Seguros	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Otros servicios	39.930	6,8%	0	3.630	3.630	3.630	3.630	3.630	3.630	3.630	3.630	3.630	3.630	3.630
Suministros	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Viajes, dietas...	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Material Oficina	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Transportes	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Liq. costes salariales	33.000	5,6%	0	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
Gastos excepcionales	0	0,0%	0	0	0	0	0	0	0	0	0	0	0	0
Otros pagos	**222.467**	38,0%	**7.587**	**5.187**	**14.742**	**5.187**	**14.742**	**5.187**	**59.109**	**19.304**	**26.409**	**19.304**	**26.409**	**19.304**
Amort. préstamos	14.361	2,5%	1.197	1.197	1.197	1.197	1.197	1.197	1.197	1.197	1.197	1.197	1.197	1.197
Gastos financieros	5.681	1,0%	1.390	390	390	390	390	390	390	390	390	390	390	390
Compra activos	119.950	20,5%	5.000	0	0	0	0	0	44.367	14.117	14.117	14.117	14.117	14.117
Liquidación I.V.A.	42.875	7,3%	0	0	9.555	0	9.555	0	9.555	0	7.105	0	7.105	0
Retenciones salariales	39.600	6,8%	0	3.600	3.600	3.600	3.600	3.600	3.600	3.600	3.600	3.600	3.600	3.600
Saldo neto **mensual**			**149.012**	**21.788**	**-34.612**	**-5.437**	**-34.612**	**-5.437**	**-33.604**	**-11.854**	**-904**	**-17.104**	**-904**	**-17.104**
Saldo **acumulado a final** de mes			**149.012**	**170.800**	**136.188**	**130.752**	**96.140**	**90.703**	**57.099**	**45.246**	**44.342**	**27.239**	**26.335**	**9.232**
Saldo **acumulado con pólizas**			**149.012**	**170.800**	**136.188**	**130.752**	**96.140**	**90.703**	**57.099**	**45.246**	**44.342**	**27.239**	**26.335**	**9.232**

Plan de Negocio **PRESUPUESTO DE TESORERÍA - 5 AÑOS**

Digital Polls s.l. **Presupuesto de Tesorería - 5 años**

CASH FLOW previsional	2019	2020	%	2021	%	2022	%	2023	%
Saldo acumulado al inicio		**9.232**		**45.747**	395,5%	**77.704**	69,9%	**98.004**	26,1%
Ventas	408.375	408.375	0,0%	408.375	0,0%	408.375	0,0%	408.375	0,0%
Ventas - descuento efectos	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Impagados	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Cobro impagados	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Total cobros por ventas	**408.375**	**544.500**	**33,3%**	**544.500**	**0,0%**	**544.500**	**0,0%**	**544.500**	**0,0%**
Otros ingresos	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Socios	100.000		-100,0%		0,0%		0,0%		0,0%
Préstamos	65.000	0	-100,0%	0	0,0%	0	0,0%	0	0,0%
Ingresos financieros	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Subvenciones	0		0,0%		0,0%		0,0%		0,0%
I.V.A. y otros	21.420	0	-100,0%	0	0,0%	0	0,0%	0	0,0%
Total otros ingresos	**186.420**	**0**	**-100,0%**	**0**	**0,0%**	**0**	**0,0%**	**0**	**0,0%**
TOTAL COBROS	**594.795**	**544.500**	**-8,5%**	**544.500**	**0,0%**	**544.500**	**0,0%**	**544.500**	**0,0%**
Salarios e incentivos	100.800	105.677	4,8%	105.677	0,0%	105.677	0,0%	105.677	0,0%
Comisiones	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Compras	136.126	163.350	20,0%	163.350	0,0%	163.350	0,0%	163.350	0,0%
Otros costes	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Variables prod/servicio	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Publicidad	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Relaciones Púb.	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Gastos de Ventas	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Variables de Ventas	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Gastos I+D	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Arrendamientos	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Conservación	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
S. Profesionales	53.240	58.080	9,1%	58.080	0,0%	58.080	0,0%	58.080	0,0%
Tributos	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Seguros	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Otros servicios	39.930	43.560	9,1%	43.560	0,0%	43.560	0,0%	43.560	0,0%
Suministros	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Viajes, dietas...	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Material Oficina	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Transportes	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Liq. costes salariales	33.000	29.613	-10,3%	29.032	-2,0%	29.032	0,0%	29.032	0,0%
Gastos extraordinarios	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Total pagos operativos	**363.096**	**400.280**	**10,2%**	**399.700**	**-0,1%**	**399.700**	**0,0%**	**399.700**	**0,0%**
Amort. préstamos	14.361	15.553	8,3%	16.844	8,3%	18.242	8,3%	0	-100,0%
Gastos financieros	5.681	3.489	-38,6%	2.198	-37,0%	800	-63,6%	0	-100,0%
Compra activos	119.950	0	-100,0%	0	0,0%	0	0,0%	0	0,0%
Liquidación I.V.A.	42.875	43.547	1,6%	48.510	11,4%	48.510	0,0%	48.510	0,0%
Impuesto sociedades		0	0,0%	0	0,0%	11.658	0,0%	12.550	7,7%
Retenciones salariales	39.600	45.116	13,9%	45.290	0,4%	45.290	0,0%	45.290	0,0%
Total otros pagos	**222.467**	**107.705**	**-51,6%**	**112.842**	**4,8%**	**124.500**	**10,3%**	**106.350**	**-14,6%**
TOTAL PAGOS	**585.563**	**507.985**	**-13,2%**	**512.542**	**0,9%**	**524.200**	**2,3%**	**506.050**	**-3,5%**
Saldo neto ejercicio	**9.232**	**36.515**	295,5%	**31.958**	-12,5%	**20.300**	-36,5%	**38.450**	89,4%
Saldo acumulado al final	**9.232**	**45.747**	395,5%	**77.704**	69,9%	**98.004**	26,1%	**136.454**	39,2%

El saldo en tesorería es positivo en cualquier momento y va paulatinamente creciendo.



La tesorería es positiva los primeros cuatro años gracias a la póliza de crédito que permite afrontar todos los gastos. A partir de ahí la empresa es capaz de generar cash.

7.5.7 Balance de situación

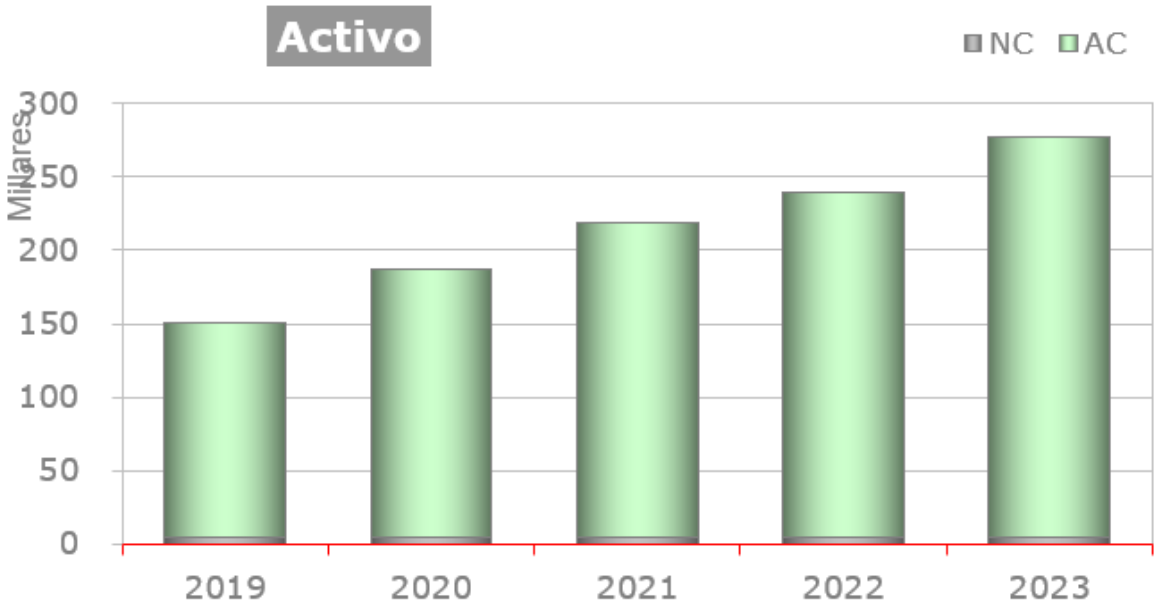
Plan de Negocio — **BALANCES PREVISIONALES - 5 AÑOS**

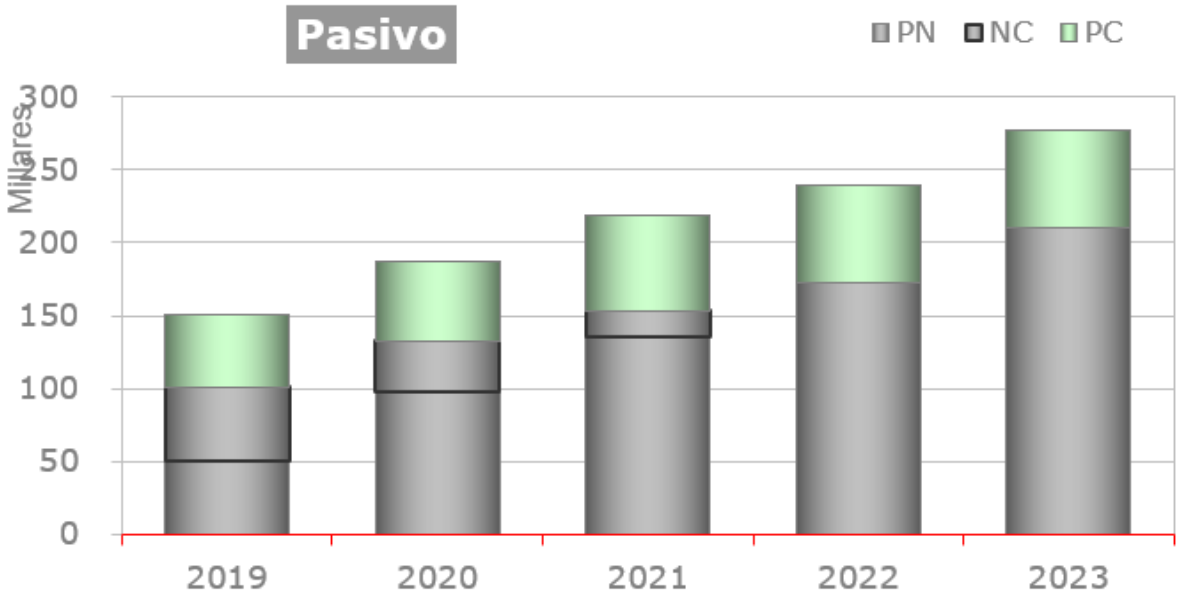
Digital Polls s.l.

Balances Previsionales - 5 años

BALANCES previsionales	2019	2020	%	2021	%	2022	%	2023	%
Activo No Corriente	**5.000**	**5.000**	0,0%	**5.000**	0,0%	**5.000**	0,0%	**5.000**	0,0%
Inmovilizado INTANGIBLE	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Inmovilizado MATERIAL	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Inversiones FINANCIERAS	5.000	5.000	0,0%	5.000	0,0%	5.000	0,0%	5.000	0,0%
Activo Corriente	**145.358**	**181.873**	25,1%	**213.830**	17,6%	**234.130**	9,5%	**272.580**	16,4%
Existencias	1	1	0,0%	1	0,0%	1	0,0%	1	0,0%
Realizable	136.125	136.125	0,0%	136.125	0,0%	136.125	0,0%	136.125	0,0%
Disponible	9.232	45.747	395,5%	77.704	69,9%	98.004	26,1%	136.454	39,2%
TOTAL **ACTIVO**	**150.358**	**186.873**	24,3%	**218.830**	17,1%	**239.130**	9,3%	**277.580**	16,1%
PATRIMONIO NETO	**50.319**	**97.830**	94,4%	**134.974**	38,0%	**172.624**	27,9%	**210.874**	22,2%
FONDOS PROPIOS	50.319	97.830	94,4%	134.974	38,0%	172.624	27,9%	210.874	22,2%
Capital	100.000	100.000	0,0%	100.000	0,0%	100.000	0,0%	100.000	0,0%
Resultados	-49.681	-2.170	-95,6%	34.974	-1711,7%	72.624	107,7%	110.874	52,7%
SUBVENCIONES	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
PASIVO NO CORRIENTE	**50.639**	**35.086**	-30,7%	**18.242**	-48,0%	**0**	-100,0%	**0**	0,0%
Préstamos largo plazo	50.639	35.086	-30,7%	18.242	-48,0%	0	-100,0%	0	0,0%
PASIVO CORRIENTE	**49.400**	**53.957**	9,2%	**65.615**	21,6%	**66.507**	1,4%	**66.707**	0,3%
Provisiones	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Deudas Entidades Crédito	0	0	0,0%	0	0,0%	0	0,0%	0	0,0%
Proveedores	35.695	35.695	0,0%	35.695	0,0%	35.695	0,0%	35.695	0,0%
Otras cuentas a pagar	13.705	18.262	33,2%	29.920	63,8%	30.812	3,0%	31.012	0,6%
TOTAL **P. NETO Y PASIVO**	**150.358**	**186.873**	24,3%	**218.830**	17,1%	**239.130**	9,3%	**277.580**	16,1%
FONDO DE MANIOBRA	**95.958**	**127.916**	33,3%	**148.216**	15,9%	**167.624**	13,1%	**205.874**	22,8%

En los balances podemos ver que el activo aumenta constantemente en paralelo al patrimonio neto.





La rentabilidad de la empresa es buena pero a largo plazo. Es lo habitual en este tipo de empresas tecnológicas y en especial de tecnologías emergentes.

La rentabilidad económica se va incrementando pero luego baja debido al estancamiento de beneficios. Lo que quiere decir que a empresa deberá ir hacia una fase de expansión en soluciones que será internacional etc.

Todos los ratios son normales dentro de este tipo de empresas y de sus comienzos. La TIR no es excesivamente alta lo que puede ser debido a que hemos sido conservadores a la hora de vender los productos por solo un 30% más de su coste, lo cual es posiblemente poco.

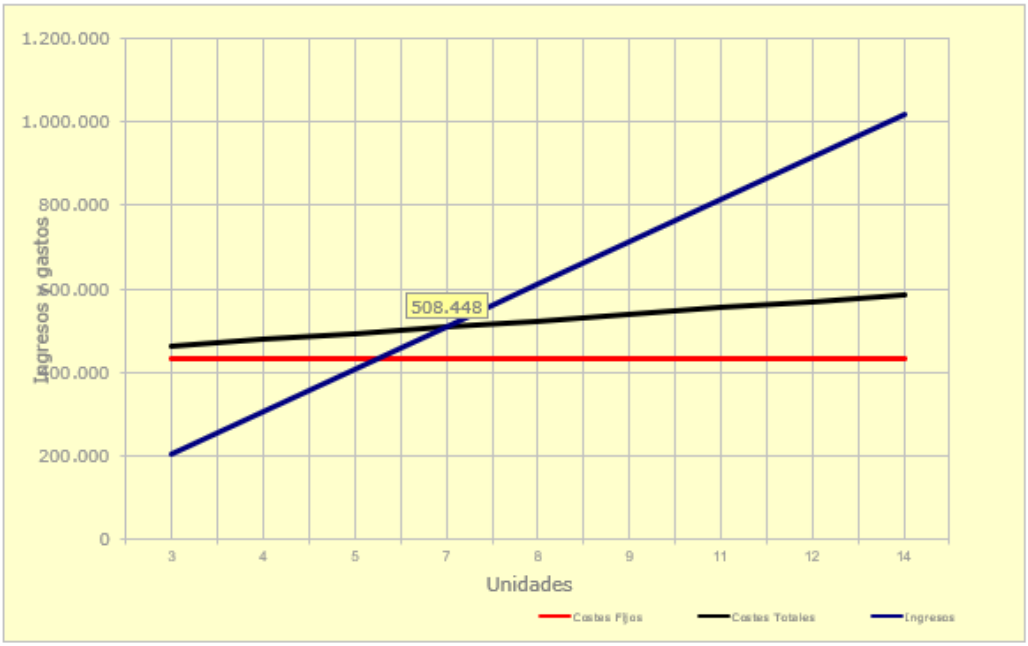
7.5.8 Punto de equilibrio

Plan de Negocio — Análisis del PUNTO DE EQUILIBRIO

Digital Polls s.l. — Plan de negocio

ANÁLISIS del PUNTO de EQUILIBRIO

Ventas mínimas anuales en unidades	**7**
Precio medio previsto	75.000,00
Facturación necesaria para alcanzar el Punto de Equilibrio	**508.448**
% s/Total ventas previstas	112,99%
Total días para alcanzar el Punto de Equilibrio anual	**407**



El punto de equilibrio se da a la 7ª venta, aproximadamente 13 meses después del comienzo de la empresa.

8. Conclusiones

- Digital Polls S.L. es una empresa viable que se basa en soluciones IoT.
- Las soluciones serán patentadas en los veinte países más importantes y con más votaciones.
- La elección de los partners para cada país es el punto clave del negocio, después de la propia solución obtenida.
- La solución obtenida es optimizable a nivel de dispositivos. Tendremos un servicio de mejora continua de la solución por parte de una subcontrata de confianza.
- Los ratios económico-financieros muestran muy buena evolución en tesorería y activos, pero estancamiento en ganancias. La causa es que solo tenemos una patente. Los beneficios de esta primera patente nos financiará en parte, futuras soluciones.
- Se realizará una expansión geográfica por Europa y Norteamérica, y a continuación por Latinoamérica y Australia.
- El punto de equilibrio se da en el momento de la séptima venta de la solución.
- La disponibilidad de la tesorería en todo momento está garantizada.
- Nuestros activos no pararán de subir en paralelo al incremento de nuestro patrimonio neto.

9. Mejoras futuras

El presente proyecto de votaciones computerizadas puede avanzar en múltiples direcciones. A corto plazo, los principales aspectos de mejora serían:

- Mejoras de seguridad: en el mencionado proyecto, se ha utilizado un tag MIFARE Classic. Esta tarjeta ha sido señalada como una tarjeta vulnerable [9] por lo que no se recomienda su uso. Es por eso, que se tendría que utilizar una tarjeta segura que permita llevar a cabo todas las operaciones sin ningún problema de seguridad. Una posible opción sería utilizar una tarjeta MIFARE Ultralight.

- Optimización de recursos: una posible opción para reducir costes podría estar en la elección de los microcontroladores. Se han utilizado dos Arduino UNO para controlar el lector de huellas y el lector de tarjetas RFID. Una posible mejora sería la elección de un solo microcontrolador que pudiera ser capaz de alimentar ambos componentes como por ejemplo un ESP32 o un Pycom. Ambos microcontroladores serían compatibles con nuestros componentes y permiten conexiones Wi-Fi lo que nos ahorraría el tener que utilizar un shield Ethernet.

A medio y largo plazo se podría:

- Aplicación: Al ser un servicio para la ciudadanía y, a priori, comprado con fondos públicos y siguiendo los principios del open data, se debe proporcionar todos los datos captados a la ciudadanía para que los puedan explotar. La manera más accesible sería la de desarrollar una aplicación móvil para que los ciudadanos pudieran consultar los datos de las votaciones en tiempo real.

- Big data y Cloud: Debido a la cantidad ingente de datos que se pueden recopilar en unas votaciones, se podrían generar modelos basados en el comportamiento de los votantes, así como las horas donde hay más afluencia de gente y el momento del día que se vota más por edades entre muchas opciones. Estos datos, se deberían almacenar en servidores en la nube para así poder garantizar su seguridad y disponibilidad.

10. Bibliografía

[1] https://www.cnccookbook.com/need-know-now-iot-internet-things-cnc-manufacturing/

[2] https://eu.usatoday.com/story/news/politics/2012/10/30/hurricane-sandy-election-impact/1668579/

[3] https://www.forbes.com/sites/jacobmorgan/2014/05/13/simple-explanation-internet-things-that-anyone-can-understand/#3a8cbc071d09

[4] https://www.adafruit.com/product/751

[5] https://www.digikey.es/product-detail/es/seeed-technology-co.,-ltd/101020057/1597-1118-ND/5482596?utm_adgroup=Sensors&mkwid=s&pcrid=278489403493&pkw=&pmt=&pdv=c&productid=5482596&slid=&gclid=Cj0KCQjwlqLdBRCKARIsAPxTGaXqpCd5WRHKJsFJyd11J6sjBfP959c3K7Sz7NVP0HY2X7jsWCP4v4UaAn05EALw_wcB

[6] http://kookye.com/2016/07/24/use-arduino-drive-fingerprint-sensor/

[7] https://www.arduino.cc/en/Guide/Introduction

[8] https://github.com/knolleary/pubsubclient

[9] https://github.com/hmxmghl/Modified_AdafruitFingerprintSensor_Library

[10] https://www.blackhat.com/docs/sp-14/materials/arsenal/sp-14-Almeida-Hacking-MIFARE-Classic-Cards-Slides.pdf

11. Anexo

11.1. Código fuente

En este apartado se adjuntará el código utilizado para la realización de este proyecto.

11.1.1. PN532

Lectura_DNI_s

```
#include <PN532.h>
#include <SoftwareSerial.h>
#define SCK 13
#define MOSI 11
#define SS 10
#define MISO 12

PN532 nfc(SCK, MISO, MOSI, SS);
SoftwareSerial myserial(4,5);
String DNI;

void setup(void) {
	Serial.begin(9600);
	Serial.println("Preparado para lectura de tarjeta");
	nfc.begin();
	nfc.SAMConfig();
	myserial.begin(9600);
}

void loop(void) {
	uint32_t id;
	id = nfc.readPassiveTargetID(PN532_MIFARE_ISO14443A);

	if (id != 0)
	{
	uint8_t keys[]= {0xFF,0xFF,0xFF,0xFF,0xFF,0xFF};
	for(uint8_t blockn=1;blockn=1;blockn) {
	if(nfc.authenticateBlock(1, id ,blockn,KEY_A,keys))
	{
		uint8_t block[16];

	    if(nfc.readMemoryBlock(1,blockn,block))
		{
```

```
                for(uint8_t i=7;i<16;i++)
                {
                DNI +=String(block[i], HEX);
                }

              Serial.print("\n\nnumero de DNI:  ");
                DNI.toUpperCase();
                char charBuf[10];
              DNI.toCharArray(charBuf,10);
                Serial.write(charBuf);
                myserial.write(charBuf);
                }
          delay(2000);
        }
   }

  }
  delay(2000);
}
```

Escritura_tarjetas_DNI

```
#include <PN532.h>
#define SCK 13
#define MOSI 11
#define SS 10
#define MISO 12

PN532 nfc(SCK, MISO, MOSI, SS);

uint8_t written=0;

void setup(void) {
  Serial.begin(9600);
  Serial.println("Hello!");

  nfc.begin();

  uint32_t versiondata = nfc.getFirmwareVersion();
  if (! versiondata) {
          Serial.print("Didn't find PN53x board");
          while (1); // halt
  }
  // Got ok data, print it out!
  Serial.print("Found chip PN5"); Serial.println((versiondata>>24) & 0xFF, HEX);
  Serial.print("Firmware ver. "); Serial.print((versiondata>>16) & 0xFF, DEC);
```

```
  Serial.print('.'); Serial.println((versiondata>>8) & 0xFF, DEC);
  Serial.print("Supports "); Serial.println(versiondata & 0xFF, HEX);

  // configure board to read RFID tags and cards
  nfc.SAMConfig();
}

void loop(void) {
  uint32_t id;

  id = nfc.readPassiveTargetID(PN532_MIFARE_ISO14443A);

  if (id != 0)
  {
          Serial.println();
          String ReceivedInput = WaitForInput("Read = 0 , Write = 1");
          Serial.print("Action selected=  ");
          Serial.println(ReceivedInput);

          int action= ReceivedInput.toInt();

          if(action == 1)  // action is Write
          {

          String ReceivedInput = WaitForInput("Block?=");
          Serial.print("Block selected=  ");
          Serial.println(ReceivedInput);

          int block_selected= ReceivedInput.toInt();

          uint8_t keys[]= {0xFF,0xFF,0xFF,0xFF,0xFF,0xFF};
          uint8_t writeBuffer[16];

          writeBuffer[0]=0;
          writeBuffer[1]=0;
          writeBuffer[2]=0;
          writeBuffer[3]=0;
          writeBuffer[4]=0;
          writeBuffer[5]=0;
          writeBuffer[6]=0;
          writeBuffer[7]=1;
          writeBuffer[8]=1;
          writeBuffer[9]=1;
          writeBuffer[10]=1;
          writeBuffer[11]=1;
          writeBuffer[12]=1;
          writeBuffer[13]=1;
          writeBuffer[14]=1;
```

```
        writeBuffer[15]=10;


        if(nfc.authenticateBlock(1, id ,block_selected,KEY_A,keys))
        {
        written = nfc.writeMemoryBlock(1,block_selected,writeBuffer);
        if(written) Serial.println("Write Successful");
        }
        }
        else
        {
        String ReceivedInput = WaitForInput("Block?=");
        Serial.print("Block selected=  ");
        Serial.println(ReceivedInput);

        int block_selected= ReceivedInput.toInt();
        uint8_t block[16];
        uint8_t keys[]= {0xFF,0xFF,0xFF,0xFF,0xFF,0xFF};
        if(nfc.authenticateBlock(1, id ,block_selected,KEY_A,keys))
        {
        if(nfc.readMemoryBlock(1,block_selected,block))
        {

        for(uint8_t i=0;i<16;i++)
        {
        //print memory block
        Serial.print(block[i], HEX);
        Serial.print(" ");
        }
        Serial.println();
        }
        else
        {
        Serial.print("No read");
        }
        }
   }

}

  delay(500);
}

String WaitForInput(String Question) {
Serial.println(Question);

  while(!Serial.available()) {
        // wait for input
```

```
 }
  return Serial.readStringUntil(10);
}
```

11.1.2. Sensor huella dactilar

Parte modificada de la librería de Adafruit:

Adafruit_Fingerprint.cpp

```
/***************************************************
  This is a library for our optical Fingerprint sensor

  Designed specifically to work with the Adafruit Fingerprint sensor
  ----> http://www.adafruit.com/products/751

  These displays use TTL Serial to communicate, 2 pins are required to
  interface
  Adafruit invests time and resources providing this open source code,
  please support Adafruit and open-source hardware by purchasing
  products from Adafruit!

  Written by Limor Fried/Ladyada for Adafruit Industries.
  BSD license, all text above must be included in any redistribution
 ****************************************************/

#include "Adafruit_Fingerprint.h"
#if defined(__AVR__) || defined(ESP8266)
    #include <SoftwareSerial.h>
#endif

//#define FINGERPRINT_DEBUG

#if ARDUINO >= 100
  #define SERIAL_WRITE(...) mySerial->write(__VA_ARGS__)
#else
  #define SERIAL_WRITE(...) mySerial->write(__VA_ARGS__, BYTE)
#endif

#define SERIAL_WRITE_U16(v) SERIAL_WRITE((uint8_t)(v>>8));
SERIAL_WRITE((uint8_t)(v & 0xFF));

#define GET_CMD_PACKET(...) \
  uint8_t data[] = {__VA_ARGS__}; \
```

```
  Adafruit_Fingerprint_Packet packet(FINGERPRINT_COMMANDPACKET,
sizeof(data), data); \
  writeStructuredPacket(packet); \
  if (getStructuredPacket(&packet) != FINGERPRINT_OK) return
FINGERPRINT_PACKETRECIEVEERR; \
  if (packet.type != FINGERPRINT_ACKPACKET) return
FINGERPRINT_PACKETRECIEVEERR;

#define SEND_CMD_PACKET(...) GET_CMD_PACKET(__VA_ARGS__); return
packet.data[0];

/**************************************************************************
**
 PUBLIC FUNCTIONS

**************************************************************************
*/


#if defined(__AVR__) || defined(ESP8266)
/**************************************************************************
*/
/*!
    @brief  Instantiates sensor with Software Serial
    @param  ss Pointer to SoftwareSerial object
    @param  password 32-bit integer password (default is 0)
*/
/**************************************************************************
*/
Adafruit_Fingerprint::Adafruit_Fingerprint(SoftwareSerial *ss, uint32_t
password) {
  thePassword = password;
  theAddress = 0xFFFFFFFF;

  hwSerial = NULL;
  swSerial = ss;
  mySerial = swSerial;
}
#endif

/**************************************************************************
*/
/*!
    @brief  Instantiates sensor with Hardware Serial
    @param  hs Pointer to HardwareSerial object
    @param  password 32-bit integer password (default is 0)
```

```
*/
/**************************************************************************
*/
Adafruit_Fingerprint::Adafruit_Fingerprint(HardwareSerial *hs, uint32_t
password) {
  thePassword = password;
  theAddress = 0xFFFFFFFF;

#if defined(__AVR__) || defined(ESP8266)
  swSerial = NULL;
#endif
  hwSerial = hs;
  mySerial = hwSerial;
}

/**************************************************************************
*/
/*!
    @brief  Initializes serial interface and baud rate
    @param  baudrate Sensor's UART baud rate (usually 57600, 9600 or
115200)
*/
/**************************************************************************
*/
void Adafruit_Fingerprint::begin(uint32_t baudrate) {
  delay(1000);  // one second delay to let the sensor 'boot up'

  if (hwSerial) hwSerial->begin(baudrate);
#if defined(__AVR__) || defined(ESP8266)
  if (swSerial) swSerial->begin(baudrate);
#endif
}

/**************************************************************************
*/
/*!
    @brief  Verifies the sensors' access password (default password is
0x0000000). A good way to also check if the sensors is active and
responding
    @returns True if password is correct
*/
/**************************************************************************
*/
boolean Adafruit_Fingerprint::verifyPassword(void) {
  return checkPassword() == FINGERPRINT_OK;
}
```

```
uint8_t Adafruit_Fingerprint::checkPassword(void) {
Serial.println("Verifying password...");
  GET_CMD_PACKET(FINGERPRINT_VERIFYPASSWORD,
                 (uint8_t)(thePassword >> 24), (uint8_t)(thePassword >>
16),
                 (uint8_t)(thePassword >> 8), (uint8_t)(thePassword &
0xFF));
  if (packet.data[0] == FINGERPRINT_OK)
    return FINGERPRINT_OK;
  else
    return FINGERPRINT_PACKETRECIEVEERR;
}

/**************************************************************************
*/
/*!
    @brief   Ask the sensor to take an image of the finger pressed on
surface
    @returns <code>FINGERPRINT_OK</code> on success
    @returns <code>FINGERPRINT_NOFINGER</code> if no finger detected
    @returns <code>FINGERPRINT_PACKETRECIEVEERR</code> on communication
error
    @returns <code>FINGERPRINT_IMAGEFAIL</code> on imaging error
*/
/**************************************************************************
*/
uint8_t Adafruit_Fingerprint::getImage(void) {
  SEND_CMD_PACKET(FINGERPRINT_GETIMAGE);
}

/**************************************************************************
*/
/*!
    @brief   Ask the sensor to convert image to feature template
    @param slot Location to place feature template (put one in 1 and
another in 2 for verification to create model)
    @returns <code>FINGERPRINT_OK</code> on success
    @returns <code>FINGERPRINT_IMAGEMESS</code> if image is too messy
    @returns <code>FINGERPRINT_PACKETRECIEVEERR</code> on communication
error
    @returns <code>FINGERPRINT_FEATUREFAIL</code> on failure to identify
fingerprint features
    @returns <code>FINGERPRINT_INVALIDIMAGE</code> on failure to identify
fingerprint features
*/
uint8_t Adafruit_Fingerprint::image2Tz(uint8_t slot) {
  SEND_CMD_PACKET(FINGERPRINT_IMAGE2TZ,slot);
```

```
}

/**************************************************************************
*/
/*!
    @brief   Ask the sensor to take two print feature template and create
a model
    @returns <code>FINGERPRINT_OK</code> on success
    @returns <code>FINGERPRINT_PACKETRECIEVEERR</code> on communication
error
    @returns <code>FINGERPRINT_ENROLLMISMATCH</code> on mismatch of
fingerprints
*/
uint8_t Adafruit_Fingerprint::createModel(void) {
  SEND_CMD_PACKET(FINGERPRINT_REGMODEL);
}

//package size
uint8_t Adafruit_Fingerprint::setSysParaSize(void) {
  uint8_t packet[] = {0x0e,6,0};
  writePacket(theAddress, FINGERPRINT_COMMANDPACKET, sizeof(packet)+2,
packet);
  delay(1000);
  uint8_t len = getReply(packet);

  if ((len != 1) && (packet[0] != FINGERPRINT_ACKPACKET))
   return -1;
  return packet[1];
}

/**************************************************************************
*/
/*!
    @brief   Ask the sensor to store the calculated model for later
matching
    @param   location The model location #
    @returns <code>FINGERPRINT_OK</code> on success
    @returns <code>FINGERPRINT_BADLOCATION</code> if the location is
invalid
    @returns <code>FINGERPRINT_FLASHERR</code> if the model couldn't be
written to flash memory
    @returns <code>FINGERPRINT_PACKETRECIEVEERR</code> on communication
error
*/
uint8_t Adafruit_Fingerprint::storeModel(uint16_t location) {
  SEND_CMD_PACKET(FINGERPRINT_STORE, 0x01, (uint8_t)(location >> 8),
(uint8_t)(location & 0xFF));
```

```
}

/**************************************************************************
*/
/*!
    @brief   Ask the sensor to load a fingerprint model from flash into
buffer 1
    @param   location The model location #
    @returns <code>FINGERPRINT_OK</code> on success
    @returns <code>FINGERPRINT_BADLOCATION</code> if the location is
invalid
    @returns <code>FINGERPRINT_PACKETRECIEVEERR</code> on communication
error
*/
uint8_t Adafruit_Fingerprint::loadModel(uint16_t location) {
  SEND_CMD_PACKET(FINGERPRINT_LOAD, 0x01, (uint8_t)(location >> 8),
(uint8_t)(location & 0xFF));
}

/**************************************************************************
*/
/*!
    @brief   Ask the sensor to transfer 256-byte fingerprint template from
the buffer to the UART
    @returns <code>FINGERPRINT_OK</code> on success
    @returns <code>FINGERPRINT_PACKETRECIEVEERR</code> on communication
error
*/
uint8_t Adafruit_Fingerprint::getModel(void) {
  SEND_CMD_PACKET(FINGERPRINT_UPLOAD, 0x01);
}


/**************************************************************************
*/
/*!
    @brief   Ask the sensor to delete a model in memory
    @param   location The model location #
    @returns <code>FINGERPRINT_OK</code> on success
    @returns <code>FINGERPRINT_BADLOCATION</code> if the location is
invalid
    @returns <code>FINGERPRINT_FLASHERR</code> if the model couldn't be
written to flash memory
    @returns <code>FINGERPRINT_PACKETRECIEVEERR</code> on communication
error
*/
uint8_t Adafruit_Fingerprint::deleteModel(uint16_t location) {
```

```
  SEND_CMD_PACKET(FINGERPRINT_DELETE, (uint8_t)(location >> 8),
(uint8_t)(location & 0xFF), 0x00, 0x01);
}

/**************************************************************************
*/
/*!
    @brief   Ask the sensor to delete ALL models in memory
    @returns <code>FINGERPRINT_OK</code> on success
    @returns <code>FINGERPRINT_BADLOCATION</code> if the location is
invalid
    @returns <code>FINGERPRINT_FLASHERR</code> if the model couldn't be
written to flash memory
    @returns <code>FINGERPRINT_PACKETRECIEVEERR</code> on communication
error
*/
uint8_t Adafruit_Fingerprint::emptyDatabase(void) {
  SEND_CMD_PACKET(FINGERPRINT_EMPTY);
}

/**************************************************************************
*/
/*!
    @brief   Ask the sensor to search the current slot 1 fingerprint
features to match saved templates. The matching location is stored in
<b>fingerID</b> and the matching confidence in <b>confidence</b>
    @returns <code>FINGERPRINT_OK</code> on fingerprint match success
    @returns <code>FINGERPRINT_NOTFOUND</code> no match made
    @returns <code>FINGERPRINT_PACKETRECIEVEERR</code> on communication
error
*/
/**************************************************************************
*/
uint8_t Adafruit_Fingerprint::fingerFastSearch(void) {
  // high speed search of slot #1 starting at page 0x0000 and page #0x00A3
  GET_CMD_PACKET(FINGERPRINT_HISPEEDSEARCH, 0x01, 0x00, 0x00, 0x00, 0xA3);
  fingerID = 0xFFFF;
  confidence = 0xFFFF;

  fingerID = packet.data[1];
  fingerID <<= 8;
  fingerID |= packet.data[2];

  confidence = packet.data[3];
  confidence <<= 8;
  confidence |= packet.data[4];
```

```
  return packet.data[0];
}

/**************************************************************************
*/
/*!
    @brief   Ask the sensor for the number of templates stored in memory.
The number is stored in <b>templateCount</b> on success.
    @returns <code>FINGERPRINT_OK</code> on success
    @returns <code>FINGERPRINT_PACKETRECIEVEERR</code> on communication
error
*/
/**************************************************************************
*/
uint8_t Adafruit_Fingerprint::getTemplateCount(void) {
  GET_CMD_PACKET(FINGERPRINT_TEMPLATECOUNT);

  templateCount = packet.data[1];
  templateCount <<= 8;
  templateCount |= packet.data[2];

  return packet.data[0];
}

/**************************************************************************
*/
/*!
    @brief   Set the password on the sensor (future communication will
require password verification so don't forget it!!!)
    @param   password 32-bit password code
    @returns <code>FINGERPRINT_OK</code> on success
    @returns <code>FINGERPRINT_PACKETRECIEVEERR</code> on communication
error
*/
/**************************************************************************
*/
uint8_t Adafruit_Fingerprint::setPassword(uint32_t password) {
  SEND_CMD_PACKET(FINGERPRINT_SETPASSWORD, (password >> 24), (password >>
16), (password >> 8), password);
}


uint8_t Adafruit_Fingerprint::uploadModelTemplate(uint8_t packet2[],
uint8_t packet3[], uint8_t packet4[], uint8_t packet5[], uint8_t
packet6[], uint8_t packet7[]
```

```
  uint8_t packet8[], uint8_t packet9[],uint8_t packet10[],uint8_t
packet11[], uint8_t packet12[], uint8_t packet13[],uint8_t packet14[],
uint8_t packet15[], uint8_t packet16[],
  uint8_t packet17[]){
  Serial.println("Uploading model...");
    uint8_t packet[] = {FINGERPRINT_DOWNLOAD, 0x02};
    writePacket(theAddress, FINGERPRINT_COMMANDPACKET, sizeof(packet)+2,
packet);
    delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet2)+2,
packet2);
    delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet3)+2,
packet3);
    delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet4)+2,
packet4);
     delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet5)+2,
packet5);
     delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet6)+2,
packet6);
    delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet7)+2,
packet7);
    delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet8)+2,
packet8);
     delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet9)+2,
packet9);
     delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet10)+2,
packet10);
     delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet11)+2,
packet11);
     delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet12)+2,
packet12);
    delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet13)+2,
packet13);
    delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet14)+2,
packet14);
```

```
     delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet15)+2,
packet15);
     delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet16)+2,
packet16);
     delay(100);
    writePacket(theAddress, FINGERPRINT_ENDDATAPACKET, sizeof(packet17)+2,
packet17);
     delay(100);
    uint8_t len = getReply(packet);
    if ((len != 1) && (packet[0] != FINGERPRINT_ACKPACKET))
        return -1;
    return packet[1];
}

uint8_t Adafruit_Fingerprint::uploadModel(void) {
    Serial.println("Uploading model...");
    uint8_t packet[] = {FINGERPRINT_DOWNLOAD, 0x02};
uint8_t packet2[] = {0x3, 0x1, 0x5D, 0x11, 0x87, 0x0, 0xC0, 0x2, 0x80,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x80, 0x2, 0x80, 0x2, 0x80,
0x2, 0x80, 0x6, 0xC0, 0x6, 0xC0, 0xE, 0xE0, 0x1E, 0x6, 0xFD};
uint8_t packet3[] = {0xFF, 0xFE, 0xFF, 0xFE, 0xFF, 0xFE, 0xFF, 0xFE, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x3C, 0x84, 0xA0, 0x36, 0x4B, 0x89, 0x60, 0x9E, 0xB, 0x80};
uint8_t packet4[] = {0x48, 0x10, 0x9A, 0xBE, 0x26, 0x93, 0x5E, 0x5E, 0x4A,
0x97, 0x97, 0x9E, 0x35, 0x98, 0xC1, 0xFE, 0x53, 0x9F, 0xAA, 0x9E, 0x4D,
0xAC, 0x17, 0x1E, 0x30, 0xB2, 0x17, 0x9E, 0x2B, 0xA9, 0xC1, 0xDF, 0xF,
0x5E};
uint8_t packet5[] = {0x28, 0x31, 0x81, 0x9F, 0x12, 0x1D, 0x1C, 0x3C, 0x1E,
0x1A, 0x4, 0xBA, 0x18, 0x9C, 0x5, 0xB2, 0x1D, 0x9E, 0x9C, 0x32, 0x1A,
0x14, 0x9D, 0xB8, 0x1D, 0x14, 0x5E, 0xF7, 0x0, 0x0, 0x0, 0x0, 0x9, 0x17};
uint8_t packet6[] = {0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x24};
uint8_t packet7[] = {0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x24};
uint8_t packet8[] = {0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x24};
uint8_t packet9[] = {0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x24};
uint8_t packet10[] = {0x3, 0x1, 0x51, 0xD, 0x7F, 0x0, 0xFF, 0xFE, 0xFF,
0xFE, 0xF9, 0xFE, 0xE0, 0x6, 0xE0, 0x2, 0x80, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x2, 0x0, 0x2, 0x80, 0x2, 0x9, 0xC4};
```

```
uint8_t packet11[] = {0x80, 0x6, 0xC0, 0x6, 0xC0, 0xE, 0xE0, 0x3E, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x4C, 0x97, 0xCC, 0x1E, 0x22, 0x9B, 0x21, 0xFE, 0x7, 0x5};
uint8_t packet12[] = {0x38, 0xA2, 0xA1, 0x3E, 0x45, 0x2D, 0x99, 0xFE,
0x24, 0xB0, 0xDD, 0xDE, 0x49, 0x33, 0xD6, 0xBE, 0x34, 0xB6, 0x1, 0x7E,
0x1E, 0x37, 0x84, 0x5E, 0x53, 0x3B, 0xAA, 0x1E, 0x3F, 0x97, 0xA3, 0x5A,
0xE, 0x4E};
uint8_t packet13[] = {0x17, 0xB2, 0x1D, 0xBA, 0x43, 0x99, 0xE4, 0x78,
0x1C, 0x30, 0x5E, 0x38, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x4, 0xDE};
uint8_t packet14[] = {0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x24};
uint8_t packet15[] = {0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x24};
uint8_t packet16[] = {0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x24};
uint8_t packet17[] = {0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0,
0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x2A};
    writePacket(theAddress, FINGERPRINT_COMMANDPACKET, sizeof(packet)+2,
packet);
    delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet2)+2,
packet2);
    delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet3)+2,
packet3);
    delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet4)+2,
packet4);
     delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet5)+2,
packet5);
     delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet6)+2,
packet6);
    delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet7)+2,
packet7);
    delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet8)+2,
packet8);
     delay(100);
```

```
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet9)+2,
packet9);
     delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet10)+2,
packet10);
     delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet11)+2,
packet11);
     delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet12)+2,
packet12);
    delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet13)+2,
packet13);
    delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet14)+2,
packet14);
     delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet15)+2,
packet15);
     delay(100);
    writePacket(theAddress, FINGERPRINT_DATAPACKET, sizeof(packet16)+2,
packet16);
     delay(100);
    writePacket(theAddress, FINGERPRINT_ENDDATAPACKET, sizeof(packet17)+2,
packet17);
     delay(100);
    uint8_t len = getReply(packet);
    if ((len != 1) && (packet[0] != FINGERPRINT_ACKPACKET))
        return -1;
    return packet[1];
  }
uint8_t Adafruit_Fingerprint::getMatch(void) {
Serial.println("MATCHING...");
confidence = 0xFFFF;

     uint8_t packet[] = {FINGERPRINT_MATCH, 0x01};
     writePacket(theAddress, FINGERPRINT_COMMANDPACKET, sizeof(packet)+2,
packet);
    delay(1000);
    uint8_t len = getReply(packet);
  if ((len != 1) && (packet[0] != FINGERPRINT_ACKPACKET))
   return -1;


  confidence = packet[2];
  confidence <<= 8;
```

```
  confidence |= packet[3];

  return packet[1];
}
void Adafruit_Fingerprint::writePacket(uint32_t addr, uint8_t packettype,
                                       uint16_t len, uint8_t *packet) {
#ifdef FINGERPRINT_DEBUG
  Serial.print("---> 0x");
  Serial.print((uint8_t)(FINGERPRINT_STARTCODE >> 8), HEX);
  Serial.print(" 0x");
  Serial.print((uint8_t)FINGERPRINT_STARTCODE, HEX);
  Serial.print(" 0x");
  Serial.print((uint8_t)(addr >> 24), HEX);
  Serial.print(" 0x");
  Serial.print((uint8_t)(addr >> 16), HEX);
  Serial.print(" 0x");
  Serial.print((uint8_t)(addr >> 8), HEX);
  Serial.print(" 0x");
  Serial.print((uint8_t)(addr), HEX);
  Serial.print(" 0x");
  Serial.print((uint8_t)packettype, HEX);
  Serial.print(" 0x");
  Serial.print((uint8_t)(len >> 8), HEX);
  Serial.print(" 0x");
  Serial.print((uint8_t)(len), HEX);
#endif

#if ARDUINO >= 100
  mySerial->write((uint8_t)(FINGERPRINT_STARTCODE >> 8));
  mySerial->write((uint8_t)FINGERPRINT_STARTCODE);
  mySerial->write((uint8_t)(addr >> 24));
  mySerial->write((uint8_t)(addr >> 16));
  mySerial->write((uint8_t)(addr >> 8));
  mySerial->write((uint8_t)(addr));
  mySerial->write((uint8_t)packettype);
  mySerial->write((uint8_t)(len >> 8));
  mySerial->write((uint8_t)(len));
#else
  mySerial->print((uint8_t)(FINGERPRINT_STARTCODE >> 8), BYTE);
  mySerial->print((uint8_t)FINGERPRINT_STARTCODE, BYTE);
  mySerial->print((uint8_t)(addr >> 24), BYTE);
  mySerial->print((uint8_t)(addr >> 16), BYTE);
  mySerial->print((uint8_t)(addr >> 8), BYTE);
  mySerial->print((uint8_t)(addr), BYTE);
  mySerial->print((uint8_t)packettype, BYTE);
  mySerial->print((uint8_t)(len >> 8), BYTE);
  mySerial->print((uint8_t)(len), BYTE);
```

```
#endif

  uint16_t sum = (len>>8) + (len&0xFF) + packettype;
  for (uint8_t i=0; i< len-2; i++) {
#if ARDUINO >= 100
    mySerial->write((uint8_t)(packet[i]));
#else
    mySerial->print((uint8_t)(packet[i]), BYTE);
#endif
#ifdef FINGERPRINT_DEBUG
    Serial.print(" 0x"); Serial.print(packet[i], HEX);
#endif
    sum += packet[i];
  }
#ifdef FINGERPRINT_DEBUG
  //Serial.print("Checksum = 0x"); Serial.println(sum);
  Serial.print(" 0x"); Serial.print((uint8_t)(sum>>8), HEX);
  Serial.print(" 0x"); Serial.println((uint8_t)(sum), HEX);
#endif
#if ARDUINO >= 100
  mySerial->write((uint8_t)(sum>>8));
  mySerial->write((uint8_t)sum);
#else
  mySerial->print((uint8_t)(sum>>8), BYTE);
  mySerial->print((uint8_t)sum, BYTE);
#endif
}

uint8_t Adafruit_Fingerprint::getReply(uint8_t packet[], uint16_t timeout)
{
  uint8_t reply[20], idx;
  uint16_t timer=0;

  idx = 0;
#ifdef FINGERPRINT_DEBUG
  Serial.print("<--- ");
#endif
while (true) {
    while (!mySerial->available()) {
      delay(1);
      timer++;
      if (timer >= timeout) return FINGERPRINT_TIMEOUT;
    }
    // something to read!
    reply[idx] = mySerial->read();
#ifdef FINGERPRINT_DEBUG
    Serial.print(" 0x"); Serial.print(reply[idx], HEX);
```

```
#endif
    if ((idx == 0) && (reply[0] != (FINGERPRINT_STARTCODE >> 8)))
      continue;
    idx++;

    // check packet!
    if (idx >= 9) {
      if ((reply[0] != (FINGERPRINT_STARTCODE >> 8)) ||
          (reply[1] != (FINGERPRINT_STARTCODE & 0xFF)))
          return FINGERPRINT_BADPACKET;
      uint8_t packettype = reply[6];
      //Serial.print("Packet type"); Serial.println(packettype);
      uint16_t len = reply[7];
      len <<= 8;
      len |= reply[8];
      len -= 2;
      //Serial.print("Packet len"); Serial.println(len);
      if (idx <= (len+10)) continue;
      packet[0] = packettype;
      for (uint8_t i=0; i<len; i++) {
        packet[1+i] = reply[9+i];
      }
#ifdef FINGERPRINT_DEBUG
      Serial.println();
#endif
      return len;
    }
  }
}


/**************************************************************************
*/
/*!
    @brief   Helper function to process a packet and send it over UART to
the sensor
    @param   packet A structure containing the bytes to transmit
*/
/**************************************************************************
*/

void Adafruit_Fingerprint::writeStructuredPacket(const
Adafruit_Fingerprint_Packet & packet) {
  SERIAL_WRITE_U16(packet.start_code);
  SERIAL_WRITE(packet.address[0]);
  SERIAL_WRITE(packet.address[1]);
  SERIAL_WRITE(packet.address[2]);
```

```
  SERIAL_WRITE(packet.address[3]);
  SERIAL_WRITE(packet.type);

  uint16_t wire_length = packet.length + 2;
  SERIAL_WRITE_U16(wire_length);

  uint16_t sum = ((wire_length)>>8) + ((wire_length)&0xFF) + packet.type;
  for (uint8_t i=0; i< packet.length; i++) {
    SERIAL_WRITE(packet.data[i]);
    sum += packet.data[i];
  }

  SERIAL_WRITE_U16(sum);
  return;
}


//transfer a fingerprint template from Char Buffer 1 to host computer
uint8_t Adafruit_Fingerprint::getModel2(void) {
    uint8_t packet[] = {FINGERPRINT_UPLOAD, 0x02};
    writePacket(theAddress, FINGERPRINT_COMMANDPACKET, sizeof(packet)+2,
packet);
    uint8_t len = getReply(recvPacket);

    if ((len != 1) && (recvPacket[0] != FINGERPRINT_ACKPACKET))
        return -1;
    return recvPacket[1];
    return recvPacket[2];
}

/**************************************************************************
*/
/*!
    @brief   Helper function to receive data over UART from the sensor and
process it into a packet
    @param   packet A structure containing the bytes received
    @param   timeout how many milliseconds we're willing to wait
    @returns <code>FINGERPRINT_OK</code> on success
    @returns <code>FINGERPRINT_TIMEOUT</code> or
<code>FINGERPRINT_BADPACKET</code> on failure
*/
/**************************************************************************
*/
uint8_t
Adafruit_Fingerprint::getStructuredPacket(Adafruit_Fingerprint_Packet *
packet, uint16_t timeout) {
  uint8_t byte;
```

```
  uint16_t idx=0, timer=0;

  while(true) {
    while(!mySerial->available()) {
      delay(1);
      timer++;
      if( timer >= timeout) {
#ifdef FINGERPRINT_DEBUG
      Serial.println("Timed out");
#endif
      return FINGERPRINT_TIMEOUT;
      }
    }
    byte = mySerial->read();
#ifdef FINGERPRINT_DEBUG
    Serial.print("<- 0x"); Serial.println(byte, HEX);
#endif
    switch (idx) {
      case 0:
        if (byte != (FINGERPRINT_STARTCODE >> 8))
        continue;
        packet->start_code = (uint16_t)byte << 8;
        break;
      case 1:
        packet->start_code |= byte;
        if (packet->start_code != FINGERPRINT_STARTCODE)
        return FINGERPRINT_BADPACKET;
        break;
      case 2:
      case 3:
      case 4:
      case 5:
        packet->address[idx-2] = byte;
        break;
      case 6:
      packet->type = byte;
      break;
      case 7:
      packet->length = (uint16_t)byte << 8;
      break;
      case 8:
      packet->length |= byte;
      break;
      default:
        packet->data[idx-9] = byte;
        if((idx-8) == packet->length)
          return FINGERPRINT_OK;
```

```
        break;
    }
    idx++;
  }
  // Shouldn't get here so...
  return FINGERPRINT_BADPACKET;
}
```

Fingerprint_sensor.ino

```
#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>
#include <SPI.h>
#include <Ethernet.h>
#include <PubSubClient.h>

char txtMsg [10];
EthernetClient ethClient;
PubSubClient client(ethClient);

SoftwareSerial mySerial(2, 3);

Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial);

void setup() {
  Serial.begin(9600);
  readData();
  byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
  IPAddress mqtt_server(192, 168, 0, 156);
  // set the data rate for the sensor serial port
  finger.begin(9600);
   if (finger.verifyPassword()) {
    Serial.println("Found fingerprint sensor!");
    if (Ethernet.begin(mac) == 0) {
      Serial.println("Failed to configure Ethernet using DHCP");
    }
    client.setServer(mqtt_server, 1883);
    //readData();
  } else {
    Serial.println("Did not find fingerprint sensor :(");
    while (1);
  }

}

void loop() {
   int id = 1;
```

```
   int p = -1;
  Serial.print("Waiting for valid finger to enroll as #");
Serial.println(id);
  while (p != FINGERPRINT_OK) {
    p = finger.getImage();
    switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Image taken");
      break;
    case FINGERPRINT_NOFINGER:
      Serial.println(".");
      break;
    default:
      Serial.println("Unknown error");
      break;
    }
  }

  // OK success!

  p = finger.image2Tz(1);
  switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Image converted");
      break;
    case FINGERPRINT_IMAGEMESS:
      Serial.println("Image too messy");
      return p;
    default:
      Serial.println("Unknown error");
      return p;
  }

  Serial.println("Remove finger");
  delay(2000);
  p = 0;
  while (p != FINGERPRINT_NOFINGER) {
    p = finger.getImage();
  }
  Serial.print("ID "); Serial.println(id);
  p = -1;
  Serial.println("Place same finger again");
  while (p != FINGERPRINT_OK) {
    p = finger.getImage();
    switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Image taken");
```

```
    break;
  case FINGERPRINT_NOFINGER:
    Serial.print(".");
    break;
  default:
    Serial.println("Unknown error");
    break;
  }
}

// OK success!

p = finger.image2Tz(2);
switch (p) {
  case FINGERPRINT_OK:
    Serial.println("Image converted");
    break;
  case FINGERPRINT_IMAGEMESS:
    Serial.println("Image too messy");
    return p;
  case FINGERPRINT_PACKETRECIEVEERR:
    Serial.println("Communication error");
    return p;
  default:
    Serial.println("Unknown error");
    return p;
}

// OK converted!
Serial.print("Creating model for #");  Serial.println(id);

p = finger.createModel();
if (p == FINGERPRINT_OK) {
  Serial.println("Prints matched!");
} else if (p == FINGERPRINT_PACKETRECIEVEERR) {
  Serial.println("Communication error");
  return p;
} else {
  Serial.println("Unknown error");
  return p;
}

Serial.print("ID "); Serial.println(id);
p = finger.storeModel(id);
if (p == FINGERPRINT_OK) {
  Serial.println("Stored!");
} else if (p == FINGERPRINT_PACKETRECIEVEERR) {
```

```
    Serial.println("Communication error");
    return p;
  } else {
    Serial.println("Unknown error");
    return p;
  }
  delay(1000);
  p = finger.loadModel(id);
  if (p == FINGERPRINT_OK) {
    Serial.println("Template loaded!");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("Communication error");
    return p;
  } else {
    Serial.println("Unknown error");
    return p;
  }
  p = finger.uploadModel();
  switch (p) {
    case FINGERPRINT_OK:
      Serial.print("upload "); Serial.print(id); Serial.println("
loaded");Serial.println(p);
      Serial.read();
      break;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      return p;
    default:
      Serial.println("Unknown error "); Serial.println(p);
      return p;
}
 uint8_t c = finger.getMatch();
 switch (c) {
    case FINGERPRINT_OK:
      Serial.print("Match!"); Serial.print(“confidence: “);
Serial.println(c.confidence);
      sendResults();
      break;
    default:
      Serial.println("No Match! "); Serial.println(c);
      return c;
}
  readData();
}

void sendResults(){
  while(true){
```

```
    delay(5000);
    Serial.println("Sending results to DB");
    if (!client.connected()) {
      Serial.print("Connecting ...\n");
      client.connect("Arduino Client");
    }
    else {
      client.publish("colegio/mesa", txtMsg);
      break;
    }
  }
}

void readData(){
  Serial.println("Press any key to start");
    //Receives data from PN532
    Serial.println("Waiting for data");
    while(true){
    if(Serial.available()>0){
        Serial.readBytes(txtMsg, 10);
        delay(3000);
        break;
      }
    }
}
```

11.1.3. Servidor local

iniDB.py

Script que borra el contenido de la base de datos en caso de que existan datos de votantes y lo inicializa con los datos por defecto de los votantes aún no habiendo votando

```
#!/usr/bin/python
# Autor: David Sola
# coding=utf-8
import MySQLdb

#conectar a la bd
db = MySQLdb.connect("localhost","root","root","TFM" )

#inicializar cursor
cursor = db.cursor()

#crear tabla de votantes
cursor.execute("DROP TABLE IF EXISTS votantes")

sql = """
            CREATE TABLE votantes (id INT NOT NULL PRIMARY KEY
AUTO_INCREMENT,
                        nombre VARCHAR(20),
                        apellido1 VARCHAR(20),
                        apellido2 VARCHAR(20),
                        dni VARCHAR(20),
                        sexo VARCHAR(1),
                        idColegio VARCHAR(8),
                        haVotado BOOLEAN NOT NULL DEFAULT 0,
                        fechaVotacion TIMESTAMP NULL
      );
      """
cursor.execute(sql)

#insertar a la tabla
try:
      query="""
            INSERT INTO votantes
            (`nombre`,`apellido1`,`apellido2`,`dni`,`sexo`,`idColegio`)
            VALUES
            ('ALEX', 'PEREZ', 'GONZALEZ','11111111A','H','00001193')
            """
      cursor.execute(query)
```

```
        db.commit()

        query="""
              INSERT INTO votantes
              (`nombre`,`apellido1`,`apellido2`,`dni`,`sexo`,`idColegio`)
              VALUES
              ('MARIA', 'RIERA', 'FUENTE','12345678B','M', '00001193')
              """
        cursor.execute(query)
        db.commit()

        query="""
              INSERT INTO votantes
              (`nombre`,`apellido1`,`apellido2`,`dni`,`sexo`, `idColegio`)
              VALUES
              ('JAVIER', 'SANCHEZ', 'FORNT','87654321C','H', '00001193')
              """
        cursor.execute(query)
        db.commit()

        query="""
              INSERT INTO votantes
              (`nombre`,`apellido1`,`apellido2`,`dni`,`sexo`,`idColegio`)
              VALUES
              ('TERESA', 'MORALES', 'PRAT','12348765D','M', '00001193')
              """
        cursor.execute(query)
        db.commit()
except:
        db.rollback()


db.close()
```

utilsSQL.py

Conjunto de funciones potencialmente reusables para realizar consultas, obtención y guardado de datos acerca de los votantes en la base de datos:

```
import MySQLdb
#Autor: David Sola
#conectar a la bd
db = MySQLdb.connect("localhost","root","root","TFM" )

#inicializar cursor
cursor = db.cursor()

def getInfoVotante(dni):
	try:
		query = "SELECT * FROM votantes WHERE dni='" + dni + "';"
		print query
		cursor.execute(query)
		row = cursor.fetchone()
		return row
	except:
		print "Error retrieving voter data"

def actualizarHaVotado(dni):
	try:
		query = "UPDATE votantes SET haVotado=1, fechaVotacion=now()
WHERE dni ='" + dni + "';"
		print query
		cursor.execute(query)
		db.commit()
		row = cursor.fetchone()
	except:
		print "Error actualizando el votante"

if __name__ == "__main__":
	getInfoVotante("11111111A")
```

gestorServer.py

Script encargado de gestionar las peticiones procedentes de la mesa, enviar los datos del votante al display correspondiente y actualizar la base de datos.

```
#Autor: David Sola
import paho.mqtt.client as mqtt
import utilsSQL as uSQL
import json
import base64

BROKER_ADDRESS = "192.168.1.50" #Local
#BROKER_ADDRESS = "192.168.204.122" #UPC School

# The callback for when the client receives a CONNACK response from the
server.
def on_connect(client, userdata, flags, rc):
    print("Connected with result code "+str(rc))

def on_message(client, userdata, message):
      print "Message topic: " + message.topic
      print "Message received: "  + message.payload #message.topic
message.qos message.retain

      if message.topic == "colegio/mesa":
            infoVotante = uSQL.getInfoVotante(message.payload)    # dni
voter
            print "Data from DB: %s" % (infoVotante,)
            votanteSerializado = crearVotanteJson(infoVotante)
            print("Publishing to colegio/servidor")
            client.publish("colegio/servidor",
json.dumps(votanteSerializado))
      elif message.topic == "colegio/display":
            resultado = json.loads(message.payload)
            print(resultado["dni"])
            if resultado["ok"] == 1:
                  uSQL.actualizarHaVotado(resultado["dni"])

def crearVotanteJson(infoVotante):
      data = {}
      data["dni"] = infoVotante[4]
      data["primerNombre"] = infoVotante[1]
      data["primerApellido"] = infoVotante[2]
      data["segundoApellido"] = infoVotante[3]
      data["sexo"] = infoVotante[5]
      data["haVotado"] = infoVotante[7]
      print "Voter serialized: " + json.dumps(data)
```

```
        f=open("images/"+ data["dni"] + ".png", "rb")
        fileContent = f.read()
        byteArr = bytes(fileContent)
        encoded = base64.encodestring(byteArr)
        data["foto"] = encoded
        return data

print("creating new instance")
client = mqtt.Client("P1") #create new instance
client.on_connect = on_connect
client.on_message = on_message

print("Connecting to broker...")
client.connect(BROKER_ADDRESS) #connect to broker
print("Subscribing to topics","colegio/mesa | colegio/display")
client.subscribe([("colegio/mesa", 2),("colegio/display", 2)])

client.loop_forever()
```

11.1.4. Display

gestorDisplay.py

Script encargado de obtener la información del votante, mostrarla mediante una interfaz gráfica y enviar el resultado de la operación al servidor local.

```
#Autor: David Sola
from Tkinter import *
import paho.mqtt.client as mqtt
import os
import json
import base64
from PIL import ImageTk, Image
from PIL import ImageFile

BROKER_ADDRESS = "192.168.1.50" #Local
#BROKER_ADDRESS = "192.168.204.122" #UPC School

TOPIC_COLEGIO_SERVIDOR = "colegio/servidor"
jsonVotante = {}

def centrar(toplevel):
      toplevel.update_idletasks()
      w = toplevel.winfo_screenwidth()
      h = toplevel.winfo_screenheight()
      size = tuple(int(_) for _ in
toplevel.geometry().split('+')[0].split('x'))
      x = w/2 - size[0]/2
      y = h/2 - size[1]/2
      toplevel.geometry("%dx%d+%d+%d" % (size + (x, y)))

def on_connect(client, userdata, flags, rc):
    print("Connected with result code " + str(rc))
    client.subscribe(TOPIC_COLEGIO_SERVIDOR)

def votacionOk(ventana, jsonVotante, result):
      print("Validation successful: " + str(result))
      resultado = {}
      resultado["dni"] = jsonVotante["dni"]
      resultado["ok"] = 1
      client.publish("colegio/display", json.dumps(resultado))
      ventana.destroy()

def votacionNoOk(ventana, jsonVotante, result):
```

```
      print("Validation unseccessful " + str(result))
      resultado = {}
      resultado["dni"] = jsonVotante["dni"]
      resultado["ok"] = 0
      client.publish("colegio/display", json.dumps(resultado))
      ventana.destroy()

def comprobarHaVotado(ventana, haVotado):
      if(haVotado == 1):
            haVotado = Label(ventana, text="YA HA VOTADO", font="Helvetica
18 bold", foreground="red")
            haVotado.grid(row=0, sticky="W")

def on_message(client, userdata, msg):
    print ("Topic : " + msg.topic)
    jsonVotante = json.loads(msg.payload)

    imagePath = "images/" + jsonVotante["dni"] + ".png"
    f=open(imagePath, "w+")
    imageDecoded = base64.decodestring(jsonVotante["foto"])
    f.write(imageDecoded)
    print("Ruta imagen: " + imagePath)

    print "dni votante: " + jsonVotante["dni"]

    ventana = Tk()
    ventana.title("Votante")
    ventana.geometry("666x400")
    centrar(ventana)

    comprobarHaVotado(ventana, jsonVotante["haVotado"])

    nombre = Label(ventana, text="Nombre: " + jsonVotante["primerNombre"])
    nombre.grid(row=1, sticky="W")

    apellidos = Label(ventana, text="Apellidos: " +
jsonVotante["primerApellido"] + " " +jsonVotante["segundoApellido"])
    apellidos.grid(row=2, sticky="W")

    dni = Label(ventana, text="DNI: " + jsonVotante["dni"])
    dni.grid(row=3, sticky="W")

    sexo = Label(ventana, text="Sexo: " + jsonVotante["sexo"])
    sexo.grid(row=4, sticky="W")

    ImageFile.LOAD_TRUNCATED_IMAGES = True
    image = Image.open(imagePath)
```

```
    photo = ImageTk.PhotoImage(image)
    panelImg = Label(ventana, image=photo)
    panelImg.grid(row=5, sticky="NW")

    if(jsonVotante["haVotado"] == 0):
            botonOk = Button(ventana, text="OK", bg="green",
command=lambda: votacionOk(ventana, jsonVotante, 1))
            botonOk.grid(row=10, column=0, sticky="W")


    botonNok = Button(ventana, text="Cancelar", bg="red", command=lambda:
votacionNoOk(ventana, jsonVotante, 0))
    botonNok.grid(row=10, column=0, sticky="E")

    ventana.mainloop() # iniciar el programa


client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message
print("Subscribing to topic " + TOPIC_COLEGIO_SERVIDOR)
client.connect(BROKER_ADDRESS)

client.loop_forever()
```

11.2. Conexiones hardware

El esquema de conexiones que se ha utilizado para llevar a cabo las comunicaciones entre los diferentes microcontroladores y sus componentes es el siguiente:

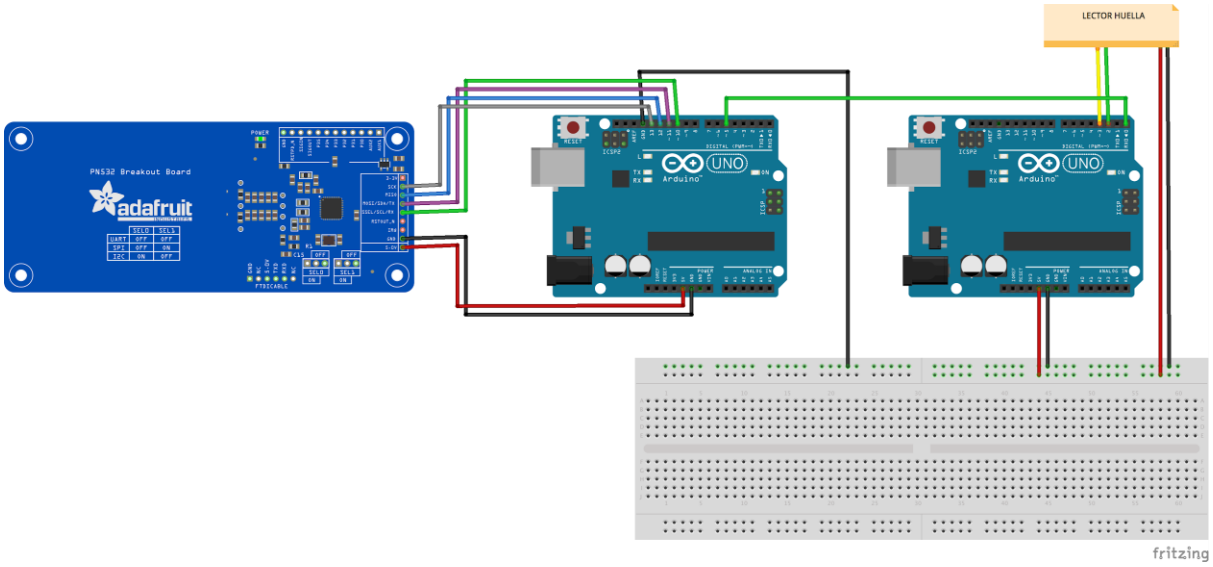


Fig. A-I: Esquema de conexiones entre los microcontroladores