

FACULTAD DE INGENIERÍA

INGENIERÍA CIVIL EN COMPUTACIÓN E INFORMÁTICA

****

Avance Final

Sistema de Seguridad y Monitoreo

“SOTER”

|  |  |
| --- | --- |
| **Integrantes:** | Ernesto García  Daniel Ramírez |
| **Curso:** | Proyecto IV |
| **Profesor:** | Diego Aracena Pizarro |

**19 de diciembre de 2022**

Historial de Avance

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor(es)** |
| 13/09/2022 | 0.1 | Se agrega el contexto, problema y solución | Ernesto García  Daniel Ramírez |
| 20/09/2022 | 0.2 | Se agregan los objetivos generales y específicos, restricciones, requisitos funcionales y no funcionales | Ernesto García  Daniel Ramírez |
| 26/09/2022 | 0.3 | Se agrega la interfaz del front-end,  diagramas de casos de uso | Ernesto García  Daniel Ramírez |
| 01/10/2022 | 0.4 | Se agrega la descripción de la empresa, metodología y herramientas | Ernesto García  Daniel Ramírez |
| 02/10/2022 | 0.5 | Se agrega el personal y roles, la carta Gantt, la arquitectura del sistema y modelo de contexto del proyecto | Ernesto García  Daniel Ramírez |
| 03/11/2022 | 0.6 | Informe corregido según la revisión del avance 1 | Ernesto García  Daniel Ramírez |
| 06/11/2022 | 0.7 | Se agregan los subsistemas, avances de la implementación, diagramas de interacción y comunicación, modelo de datos y pruebas del sistema. | Ernesto García  Daniel Ramírez |
| 16/12 | 0.8 | Informe corregido según la revisión del avance 2, se agregan las implementaciones finales. | Ernesto García |

# Índice

[**1. Introducción**](#_heading=h.2zw1j2j7o97g) **[5](#_heading=h.2zw1j2j7o97g)**

[**2. Descripción de la Empresa**](#_heading=h.o5hn9hxrjx9a) **[6](#_heading=h.o5hn9hxrjx9a)**

[**3. Resumen del proyecto**](#_heading=h.cuxcz2e7l00m) **[7](#_heading=h.cuxcz2e7l00m)**

[3.1. Contexto](#_heading=h.bx9q41bj1m0a) [7](#_heading=h.bx9q41bj1m0a)

[3.2. Problema](#_heading=h.hwnekritqzy9) [7](#_heading=h.hwnekritqzy9)

[3.3. Solución](#_heading=h.f9jepwab9anr) [8](#_heading=h.f9jepwab9anr)

[3.4. Objetivos](#_heading=h.5oraic74uz27) [9](#_heading=h.5oraic74uz27)

[3.4.1. Objetivo general](#_heading=h.prpbebc9u70u) [9](#_heading=h.prpbebc9u70u)

[3.4.2. Objetivos específicos](#_heading=h.z9lngjgkgy7d) [9](#_heading=h.z9lngjgkgy7d)

[3.5. Restricciones](#_heading=h.toxzll2lxvay) [9](#_heading=h.toxzll2lxvay)

[3.6. Entregables](#_heading=h.valkrfz4g8oo) [9](#_heading=h.valkrfz4g8oo)

[**4. Requisitos del proyecto**](#_heading=h.607zns7ceke4) **[10](#_heading=h.607zns7ceke4)**

[4.1. Requisitos funcionales](#_heading=h.rssxx51cizs2) [10](#_heading=h.rssxx51cizs2)

[4.2. Requisitos no funcionales](#_heading=h.i5pd1rwgbyjc) [10](#_heading=h.i5pd1rwgbyjc)

[**5. Organización del proyecto**](#_heading=h.od9alfal5wxz) **[11](#_heading=h.od9alfal5wxz)**

[5.1. Personal y entidades externas](#_heading=h.j3rrec2prz5g) [11](#_heading=h.j3rrec2prz5g)

[5.2. Personal y entidades internas](#_heading=h.tf6rzfyx7wf3) [11](#_heading=h.tf6rzfyx7wf3)

[5.3. Roles y responsabilidad](#_heading=h.r25re0gtqvwd) [12](#_heading=h.r25re0gtqvwd)

[5.4. Mecanismo de organización](#_heading=h.kp0vsqp5ptwd) [12](#_heading=h.kp0vsqp5ptwd)

[**6. Planificación del proyecto**](#_heading=h.kbbzl61ifru9) **[13](#_heading=h.kbbzl61ifru9)**

[**7. Planificación de los procesos técnicos**](#_heading=h.ti58anqjetz2) **[14](#_heading=h.ti58anqjetz2)**

[7.1. Metodología](#_heading=h.60b32r2mqabe) [14](#_heading=h.60b32r2mqabe)

[7.2. Herramientas](#_heading=h.naqz72a4natn) [14](#_heading=h.naqz72a4natn)

[**8. Diseño del proyecto**](#_heading=h.ywhj35r7umj8) **[15](#_heading=h.ywhj35r7umj8)**

[8.1. Arquitectura del sistema](#_heading=h.znh1xo8hht5) [15](#_heading=h.znh1xo8hht5)

[8.2. Modelo de contexto](#_heading=h.6gyv7rvw93np) [16](#_heading=h.6gyv7rvw93np)

[8.3. Subsistemas](#_heading=h.lserwyzfz68t) [16](#_heading=h.lserwyzfz68t)

[8.4. Diagrama de casos de uso](#_heading=h.6gyv7rvw93np) [17](#_heading=h.6gyv7rvw93np)

[8.5. Diagramas de flujo de casos de uso](#_heading=h.ou4e3fixzunz) [18](#_heading=h.ou4e3fixzunz)

[8.5.1. Inicio de sesión](#_heading=h.g904abucnr8t) [18](#_heading=h.g904abucnr8t)

[8.5.2. Visualizar cámaras](#_heading=h.6oec7crecirg) [19](#_heading=h.6oec7crecirg)

[8.5.3. Gestionar cuentas](#_heading=h.mpozu2uegr8r) [20](#_heading=h.mpozu2uegr8r)

[8.5.4. Gestionar cámaras](#_heading=h.l8aelwrbf10v) [21](#_heading=h.l8aelwrbf10v)

[8.6. Diagramas de interacción](#_heading=h.g57mvst3sqxh) [22](#_heading=h.g57mvst3sqxh)

[8.7. Diagramas de comunicación](#_heading=h.3ah9fsgttajr) [22](#_heading=h.3ah9fsgttajr)

[8.8. Modelo de datos](#_heading=h.g8kpv0cuvafc) [22](#_heading=h.g8kpv0cuvafc)

[8.9. Interfaz del sistema](#_heading=h.yff4twvp4bq4) [23](#_heading=h.yff4twvp4bq4)

[**9. Implementación del sistema**](#_heading=h.yumz06tm57ev) **[24](#_heading=h.yumz06tm57ev)**

[**10. Pruebas del sistema**](#_heading=h.dltifsfuvp7l) **24**

[**11. Conclusiones**](#_heading=h.ixn5neo1ufqb) **25**

[**12. Referencias**](#_heading=h.jtkmg17uyms7) **26**

# Introducción

Los sistemas de seguridad han ido evolucionando durante esta era digital, abriendo todo un abanico de opciones y herramientas a utilizar, para mantener una buena seguridad en el hogar.

Durante tiempos recientes, la empresa WAKI Labs se ha visto en la necesidad de actualizar su sistema de seguridad perimetral, ya que el actual no cumple con el estándar de calidad que necesita la empresa, y ha surgido la necesidad de que este sistema les permita mantener un registro de los trabajadores en el recinto, para facilitar el trabajo de cowork que es llevado a cabo en la empresa, y poder monitorear sin problemas la residencia, debido a acontecimientos que han golpeado la región en cuanto a la delincuencia.

Es por eso que en el presente informe de avance se documenta el proceso y desarrollo del proyecto “Sistema de Seguridad y Monitoreo” denominado “SOTER”, a desarrollar para la empresa WAKI Labs. A continuación se especifica el problema y solución del proyecto, junto a un breve contexto de la situación actual de WAKI Labs; los requisitos funcionales y no funcionales que debe cumplir el sistema a desarrollar; la organización del personal a trabajar en el proyecto y la planificación de las tareas a realizar; el diseño del proyecto, incluyendo todos los diagramas y modelos necesarios para dar forma a la solución planteada; y por último, la implementación del sistema en el Frontend y Backend, así como el producto final a entregar.

# Descripción de la Empresa

La empresa WAKI Labs, como se observa en la Figura 1, es un laboratorio de innovación fundado el 2016 en la Región de Arica y Parinacota, Chile, que busca mejorar procesos industriales a través de soluciones tecnológicas escalables, desarrollando proveedores tradicionales y Start Ups.



### Figura 1. Logo de WAKI Labs.

Ellos trabajan en dar soluciones a problemas que afectan procesos industriales en el territorio del norte de Chile con creatividad, disciplina y la capacidad de adaptarse al cambio rápidamente.

Hoy en día WAKI Labs se ha ido transformando desde un simple espacio de coworking que permite a las personas trabajar en sus emprendimientos, a un laboratorio de prototipos que atiende a la industria de territorios extremos, como la que se tiene en el desierto chileno.

# Resumen del proyecto

## Contexto

WAKI Labs es un espacio de “cowork” que permite a las personas trabajar en sus emprendimientos y mantenerlos en un lugar seguro, para esto se utiliza un sistema de seguridad que a través de los años ha quedado muy desactualizado, más aún luego de la crisis sanitaria que afectó al país y al mundo.

Hoy en día las instalaciones de WAKI Labs vuelven a abrir sus puertas, con sus miembros lentamente volviendo a trabajar de manera presencial a medida que el país vuelve a la normalidad, por eso se necesita de un sistema nuevo que permita mantener los estándares de seguridad en sus instalaciones, y que sus coworkers puedan transitar sin riesgo.

## Problema

El sistema de seguridad actual no es completamente funcional ni fácil de usar para los usuarios que lo utilizan a diario. Actualmente hay 8 cámaras de seguridad CCTV conectadas a un DVR, de las cuales tan sólo 5 son funcionales, como se observa en la Figura 2. Estas cámaras son visibles a través de una pantalla de televisión, y es usada como sistema de monitoreo con un control remoto para navegar en la interfaz. Además, estas cámaras no están ubicadas de la manera más eficiente posible, por lo que hay lugares del recinto que quedan fuera de la visión de las cámaras.



### Figura 2. Sistema de seguridad actual.

## Solución

Se planea crear un sistema propio que pueda replicar las funciones esenciales del sistema anterior, siendo capaz de mostrar todas las cámaras al mismo tiempo o poder enfocarse en una de estas e ir cambiando entre ellas.

El ingreso al sistema será a través de la IGU de un servidor de páginas web, de manera que el lado del cliente sea accesible a través de dispositivos móviles conectados a internet, logrando así tener una mayor accesibilidad en caso de ser necesitado por los usuarios.

Por otro lado, se planea tener el servidor del proceso de negocio en un equipo Linux que se encargue de manejar los servicios utilizados por el sistema.

Además, se deben reposicionar aquellas cámaras que son desaprovechadas y cuyo campo de visión no es completamente importante o ya es cubierto por otras cámaras, para conseguir una mejor cobertura del recinto y el área alrededor.

## Objetivos

### Objetivo general

Implementar un sistema de seguridad y monitoreo remoto que pueda satisfacer las necesidades de seguridad de WAKI Labs.

### Objetivos específicos

* Estudiar y definir la problemática.
* Proponer la solución al problema y sus alternativas.
* Desarrollar el sistema de acuerdo a la solución de software seleccionada.
* Realizar las pruebas y análisis de resultados del sistema.

## Restricciones

1. El proyecto será ejecutado en un periodo de 3 meses.
2. El sistema será desarrollado utilizando el Framework Angular.
3. Los usuarios pertenecen a la empresa WAKI Labs.

## Entregables

1. Bitácoras semanales.
2. Documento de Requerimientos firmado por el cliente.
3. Informe de Avance 1.
4. Informe de Avance 2.
5. Informe Final.
6. Manual de usuario.
7. Wiki del proyecto.
8. Producto.

# Requisitos del proyecto

## Requisitos funcionales

|  |  |
| --- | --- |
| **ID** | **Definición** |
| RF1 | El sistema debe verificar al usuario. |
| RF2 | El sistema debe permitir el ingreso de datos. |
| RF3 | El sistema debe almacenar la información en un historial. |
| RF4 | El sistema debe permitir ver todas las cámaras activas al mismo tiempo. |
| RF5 | El sistema debe poder centrarse en una sola cámara. |
| RF6 | El sistema debe poder alternar de cámara en cámara. |

## Requisitos no funcionales

|  |  |
| --- | --- |
| **ID** | **Definición** |
| RnF1 | El sistema debe utilizar la paleta de colores de WAKI Labs. |
| RnF2 | El sistema debe ser programado en un entorno web. |
| RnF3 | El sistema debe garantizar su funcionamiento en el navegador Mozilla Firefox. |
| RnF4 | El sistema debe poder verse a través de cualquier navegador. |
| RnF5 | El sistema debe tener una interfaz intuitiva y fácil de utilizar. |
| RnF6 | El sistema deberá ser desarrollado utilizando el Framework Angular. |
| RnF7 | El sistema debe ser accesible para dispositivos móviles y de fácil adaptación. |

# Organización del proyecto

## Personal y entidades externas

Como fue mencionado anteriormente en la descripción de la empresa, el personal es el equipo de WAKI Labs, como se puede observar en la siguiente Figura 3. En caso de ser necesario se puede consultar la ayuda de los miembros del equipo para casos específicos como lo son la conexión de las cámaras en el recinto, o ayuda con distintos softwares para desarrollar una mejor solución, o consultar al cliente y mostrar prototipos para recibir retroalimentación.



### Figura 3. Personal de WAKI Labs.

## Personal y entidades internas

Para el proyecto actual, WAKI Labs reunió un equipo de 4 estudiantes universitarios, como se puede observar en la siguiente Figura 4. Tres estudiantes de Ingeniería civil en computación e informática, dos de la Universidad de Tarapacá y uno de la Universidad Santo Tomás, además de un estudiante de la carrera Ingeniería Industrial de la Universidad de Tarapacá.



### Figura 4. Equipo de proyecto SSP en su espacio de trabajo.

## Roles y responsabilidad

|  |  |  |
| --- | --- | --- |
| **Rol** | **Responsabilidad** | **Responsable** |
| Jefe de proyecto | Se encarga de la coordinación del trabajo del grupo y de la comunicación con el usuario | Eduardo Rojo |
| Desarrollador | Se encarga de definir, diseñar, desarrollar e implementar el software | Ernesto García |
| Desarrollador | Se encarga de definir, diseñar, desarrollar e implementar el software | Daniel Ramírez |
| Secretario | Se encarga de la documentación del proyecto, realizando las bitácoras, informes y estudios requeridos | Gonzalo Muñoz |

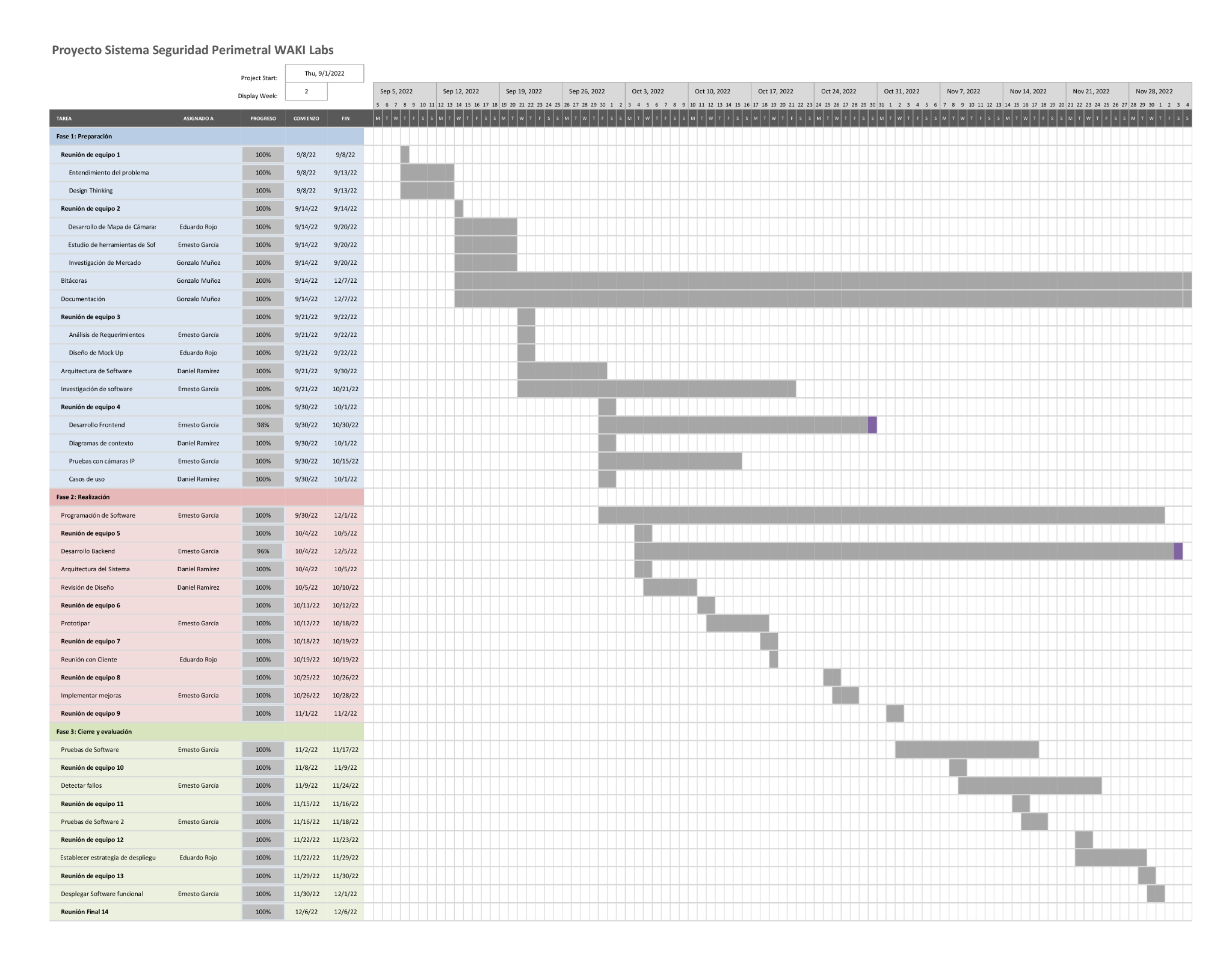
## Mecanismo de organización

El equipo de trabajo se reúne semanalmente en el recinto de WAKI Labs para discutir los avances realizados, las actividades siguientes y ponerse de acuerdo en el sprint semanal.

Los otros medios de comunicación y organización son:

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Logo** | **Función** |
| Trello |  | Utilizado como un espacio de trabajo que permite subir archivos, tareas a realizar, bitácoras e informes para mantener en un mismo sitio |
| WhatsApp |  | Utilizado como un medio para dudas y consultas sobre el proyecto, organizar reuniones o informar a los otros miembros del equipo |
| GitHub |  | Utilizado como repositorio para la solución a desarrollar, a la cual sólo tiene acceso el equipo de trabajo. |

# Planificación del proyecto



# Planificación de los procesos técnicos

## Metodología

La metodología utilizada en el proyecto es SCRUM, ya que permite una forma rápida de retroalimentación entre el equipo de desarrollo y el cliente del producto. Logrando desarrollar prototipos de este tempranamente y con menor cantidad de riesgos de equivocación en los requerimientos del cliente. La metodología es aplicada de manera que cada semana se tiene una reunión en la cual se discuten las tareas realizadas y aquellas pendientes, se planean las tareas a realizar para la semana, y se va modificando el proyecto según sea necesario para lograr los objetivos propuestos.

## Herramientas

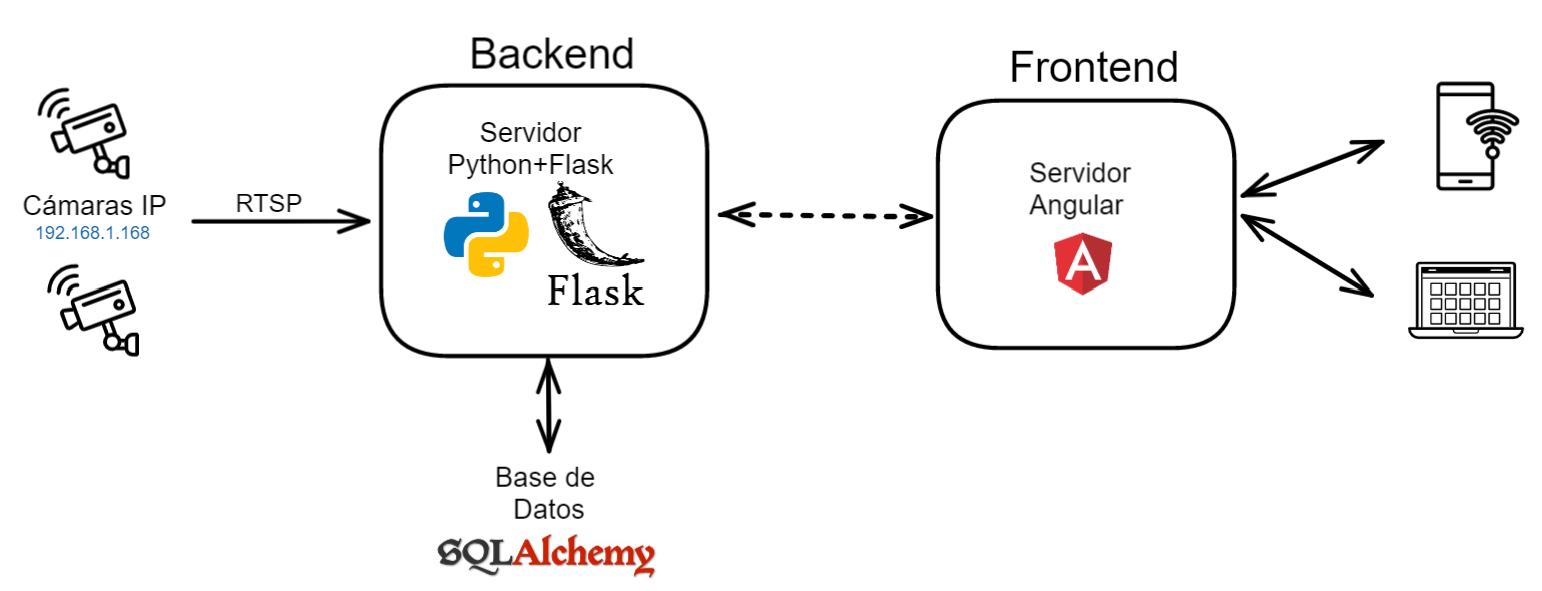
|  |  |  |
| --- | --- | --- |
| **Nombre** | **Logo** | **Función** |
| Redmine |  | Es una herramienta utilizada en el curso de proyecto para gestionar los proyectos de los estudiantes y realizar seguimiento de otros. |
| Visual Studio Code |  | Es un editor de código fuente desarrollado por Microsoft. Cuenta con soporte para depuración de código y un sinnúmero de extensiones. |
| Angular |  | Es un framework mantenido por Google, que sirve para desarrollar páginas web de estilo SPA (Single Page Application). |
| Flask |  | Es un framework módulo de Python que permite desarrollar páginas web fácilmente. |
| Python |  | Es un lenguaje de programación interpretado, multiparadigma. Es de tipo dinámico, multiplataforma y multipropósito. |
| SQL Alchemy |  | Es una librería para Python que facilita el acceso a una base de datos relacional, así como las operaciones a realizar sobre la misma. |

# Diseño del proyecto

## Arquitectura del sistema

A continuación, en la Figura 5 se puede apreciar la arquitectura propuesta para el SSP. Este sistema tendrá como base el lado del servidor hecho con Python y Flask, que permitirá conectarse a las cámaras IP y trabajar con ellas, además, tendrá una conexión a la base de datos creada en SQLAlchemy. Por otro lado, desde el lado del cliente, se tendrá un servidor Angular que mostrará la IGU para que el usuario pueda interactuar.

Este sistema se conecta a las cámaras IP en el recinto de WAKI y a través del protocolo de transmisión en tiempo real (RTSP) que consigue transmitir el video de cada una de ellas a al sistema.

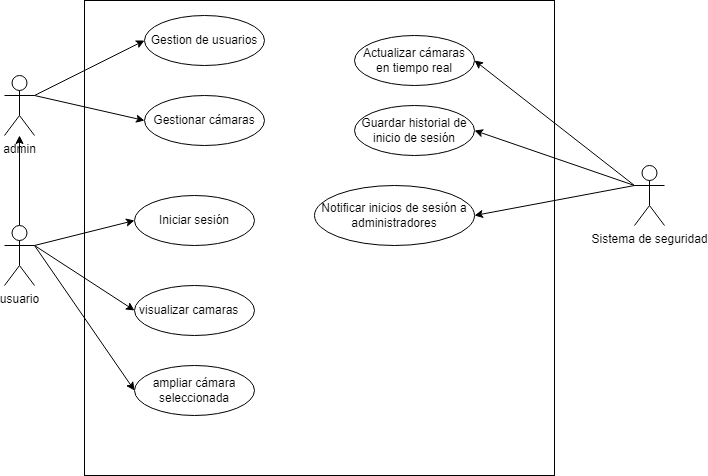


### Figura 5. Arquitectura propuesta del sistema solución.

## Diagrama de casos de uso

A continuación, en la Figura 6 se muestra el diagrama de casos de uso del sistema SSP que cuenta con un usuario y un administrador, los cuales pueden iniciar sesión, visualizar cámaras, ampliar cámaras y gestionar usuarios y cámaras, respectivamente.

Por otro lado, se tiene el sistema de seguridad que puede actualizar las cámaras en tiempo real, guardar el historial de inicios de sesión y notificar a administradores los inicios de sesión.

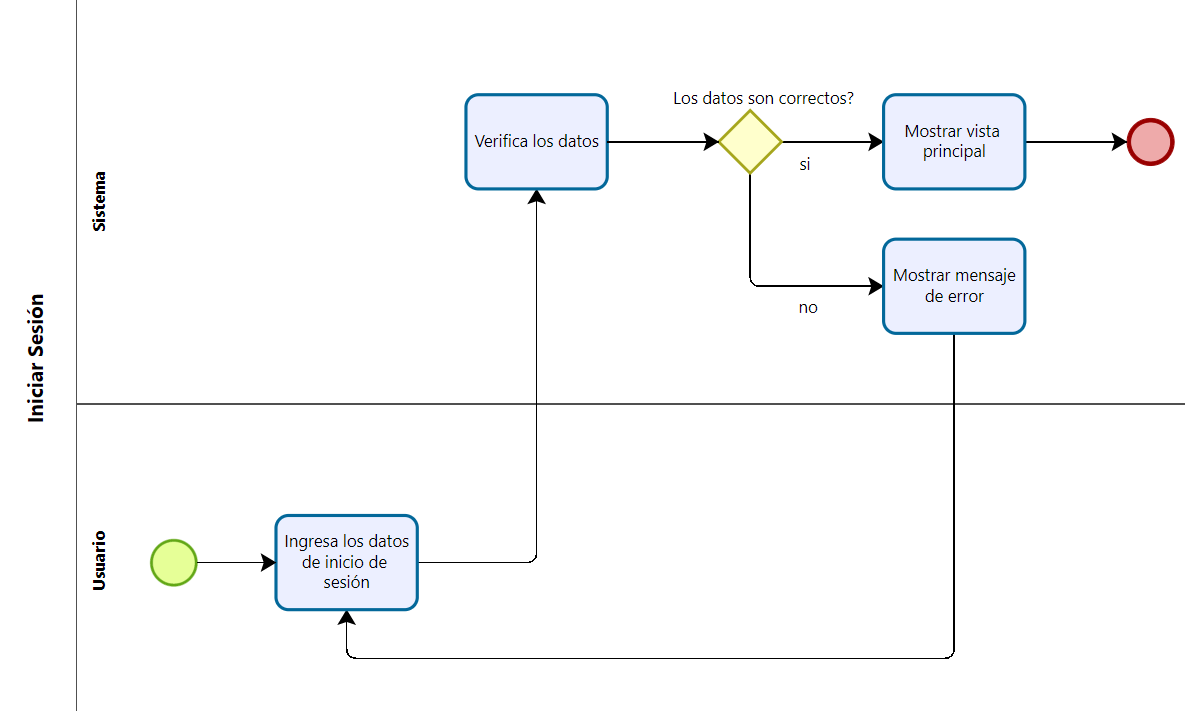


### Figura 6. Diagrama de casos de uso del sistema.

## Diagramas BPMN de casos de uso

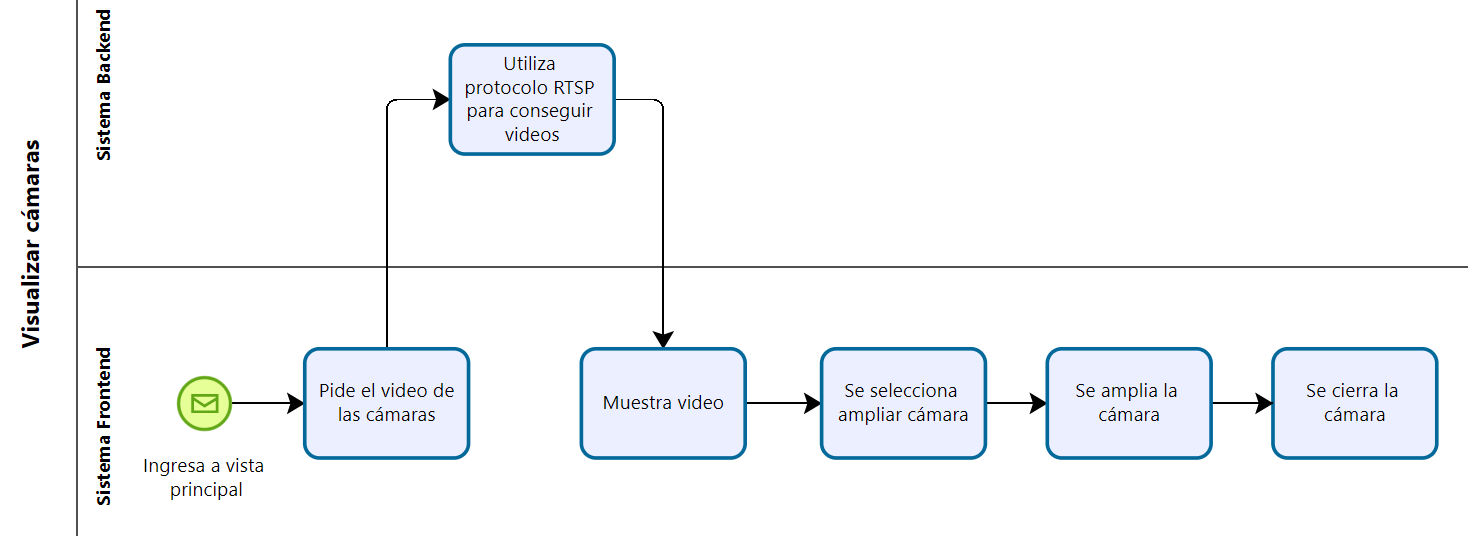
A continuación, en las Figuras 7, 8, 9 y 10 se observan los casos de uso ya diagramados, para poder entender el funcionamiento y flujo de cada uno de ellos. Estos diagramas no son finales, ya que fueron enseñados al equipo de trabajo y por la retroalimentación dada es necesario hacer cambios estos, pero de todas maneras su estado actual sirve para demostrar el funcionamiento del sistema.

### Inicio de sesión



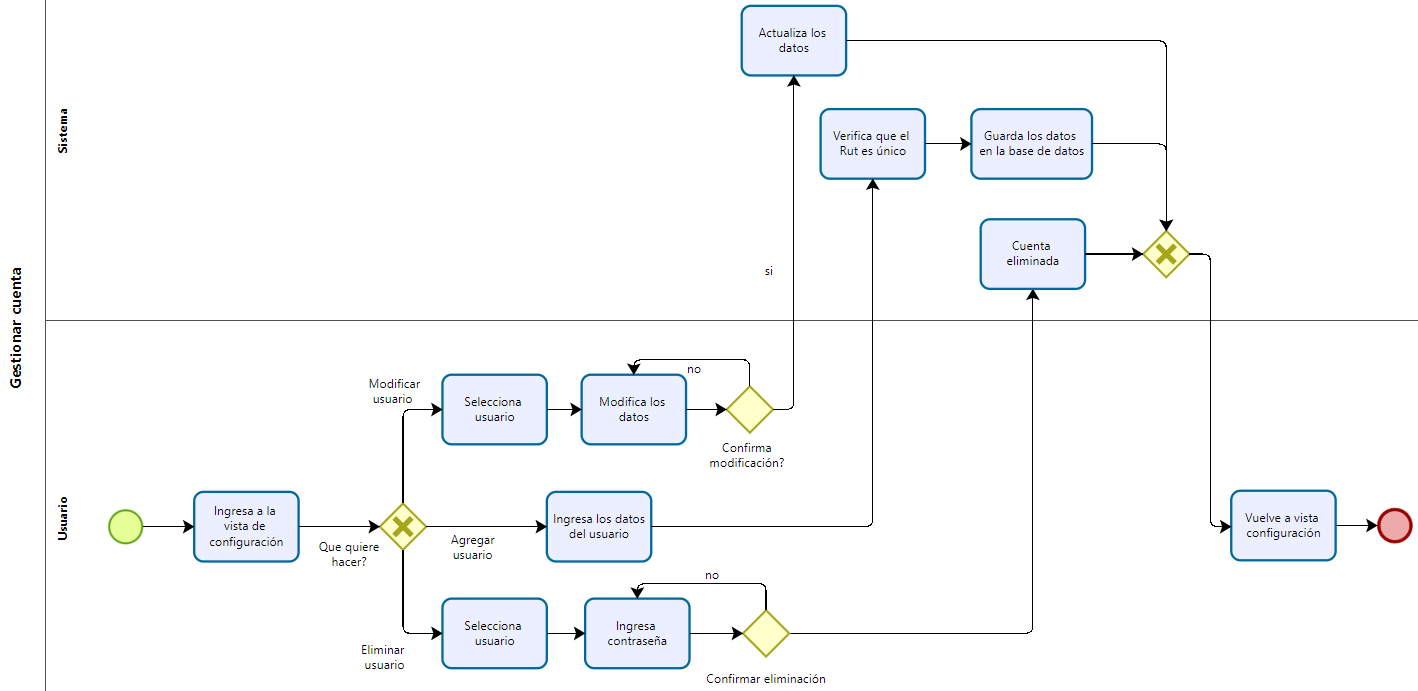
### Figura 7. Diagrama BPMN del caso de uso inicio de sesión.

### Visualizar cámaras



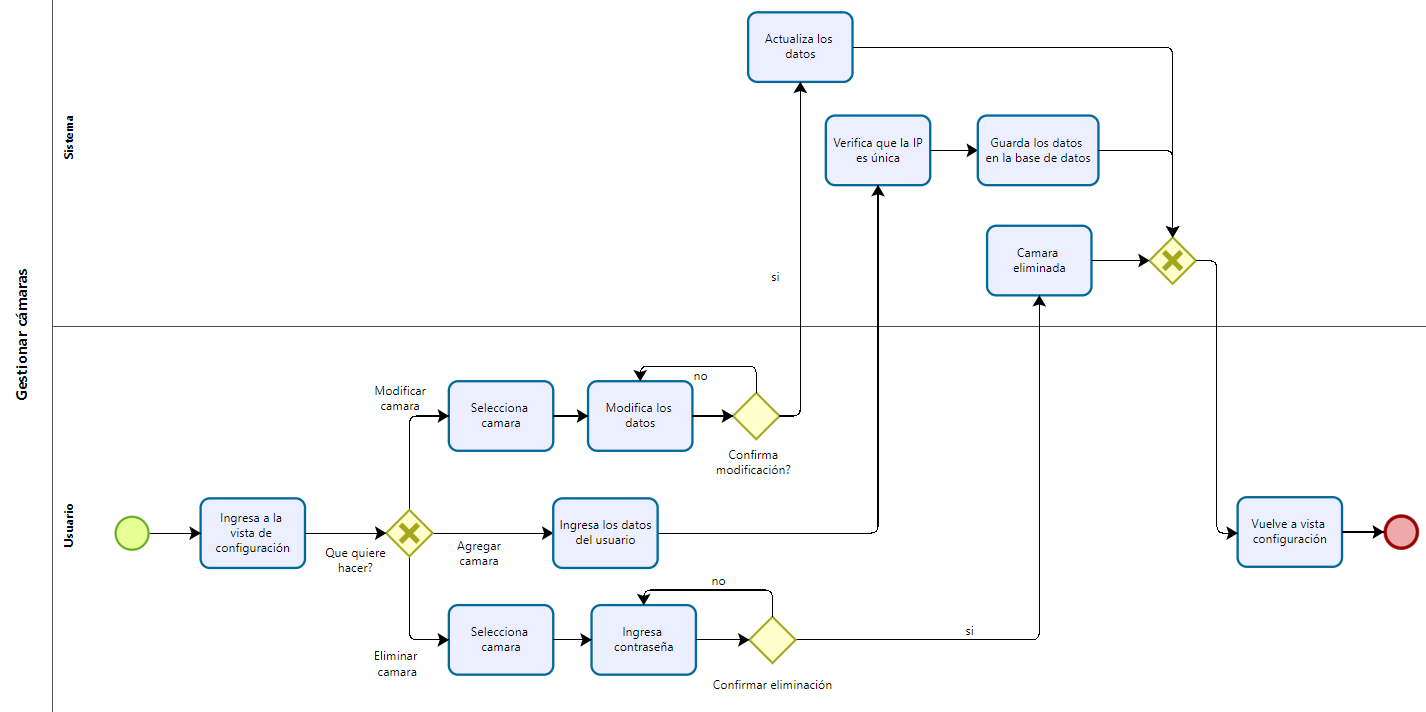
### Figura 8. Diagrama BPMN del caso de uso visualizar cámaras.

### Gestionar cuentas



### Figura 9. Diagrama del caso de uso gestionar cuentas.

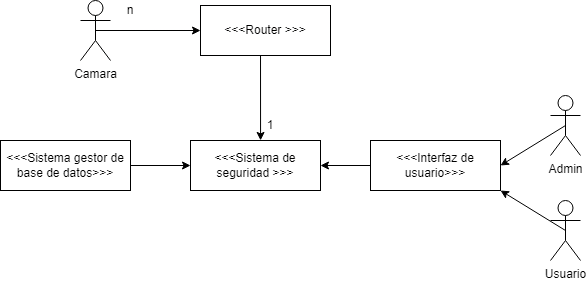
### Gestionar cámaras



### Figura 10. Diagrama del caso de uso gestionar cámaras.

## Modelo de contexto

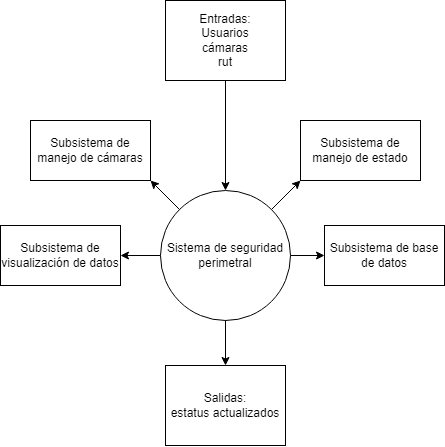
A continuación, en la Figura 11 se observa el modelo de contexto del Sistema de Seguridad Perimetral WAKI, que interactúa con las Cámaras del recinto, a través del router a internet, con el sistema gestor de base de datos y además la interfaz de usuario que permite a los usuarios ver dichas cámaras.



### Figura 11. Modelo de contexto propuesto del sistema.

## Subsistemas

En la siguiente Figura 12, se pueden observar los subsistemas que serán utilizados para lograr el funcionamiento del proyecto SSP.



### Figura 12. Diagrama de Subsistemas.

# Subsistema de manejo de manejo de cámaras: Este es el encargado de gestionar las cámaras, conectarlas al sistema y permitir ver el contenido de estas.

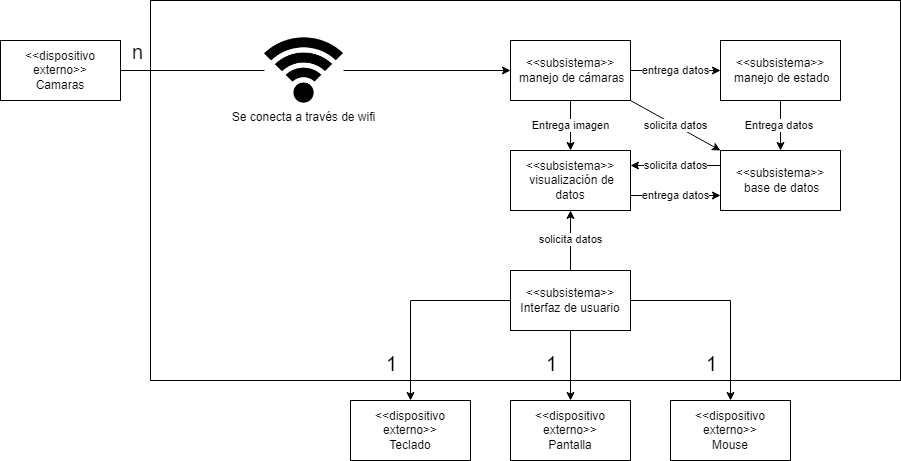
# Subsistema de manejo de manejo de estado: Este es el encargado de gestionar el estado de los trabajadores, en este caso es un booleano que simboliza si están o no dentro de WAKI Labs.

# Subsistema de base de datos: Este es el encargado de gestionar los datos que utiliza el sistema, trabajadores, datos de las cámaras y estatus de los trabajadores.

# Subsistema de visualización de datos: Este es el encargado de permitir al usuario ver en tiempo real los datos actualizados, las cámaras y los estatus de los trabajadores.

## Diagramas de interacción

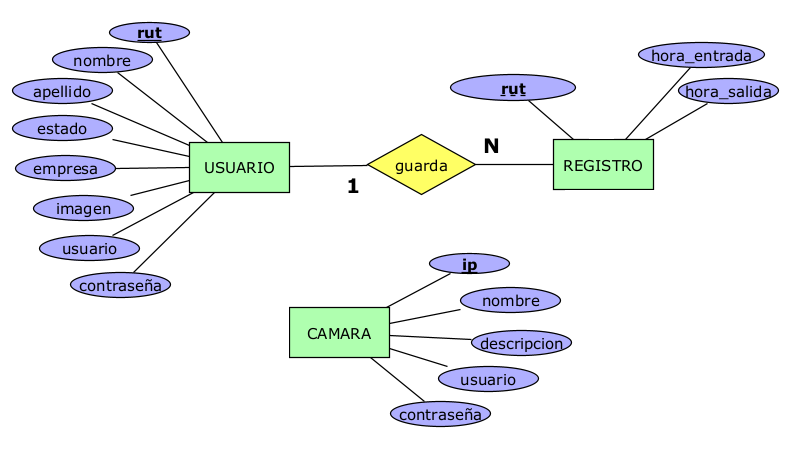
En la siguiente Figura 13 se puede apreciar la interacción entre los distintos subsistemas que forman el sistema SSP



### Figura 13. Diagrama de interacción de los subsistemas.

## Modelo de datos

Para el sistema a realizar, será necesario utilizar una base de datos para guardar los registros que se creen y la información de los miembros de WAKI, es por esto que en la siguiente Figura 14 se observa el modelo entidad-relación para la base de datos.

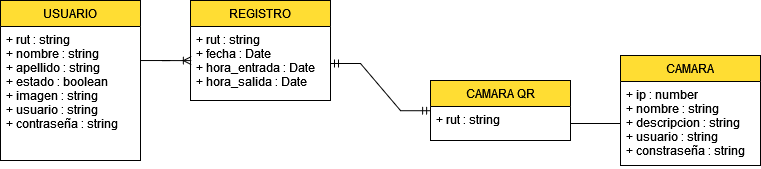
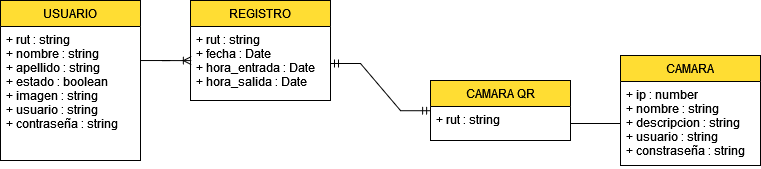


### Figura 14. Diagrama de entidad relación.

En este se tienen en cuenta 4 entidades:

* “Usuario” es la entidad que representa a cada miembro de WAKI. Posee la clave primaria “**rut”** y los datos del usuario.
* “Registro” es la entidad que representa a cada registro de entrada-salida de WAKI. Posee la clave foránea “**rut”** y la hora de entrada y salida de la persona.
* “Cámara” es la entidad que representa a cada cámara en WAKI. Posee la clave primaria “**ip”** y los datos de la cámara.

Luego se tiene el modelo de datos en la Figura 15, que especifica los tipos de variable de cada atributo.

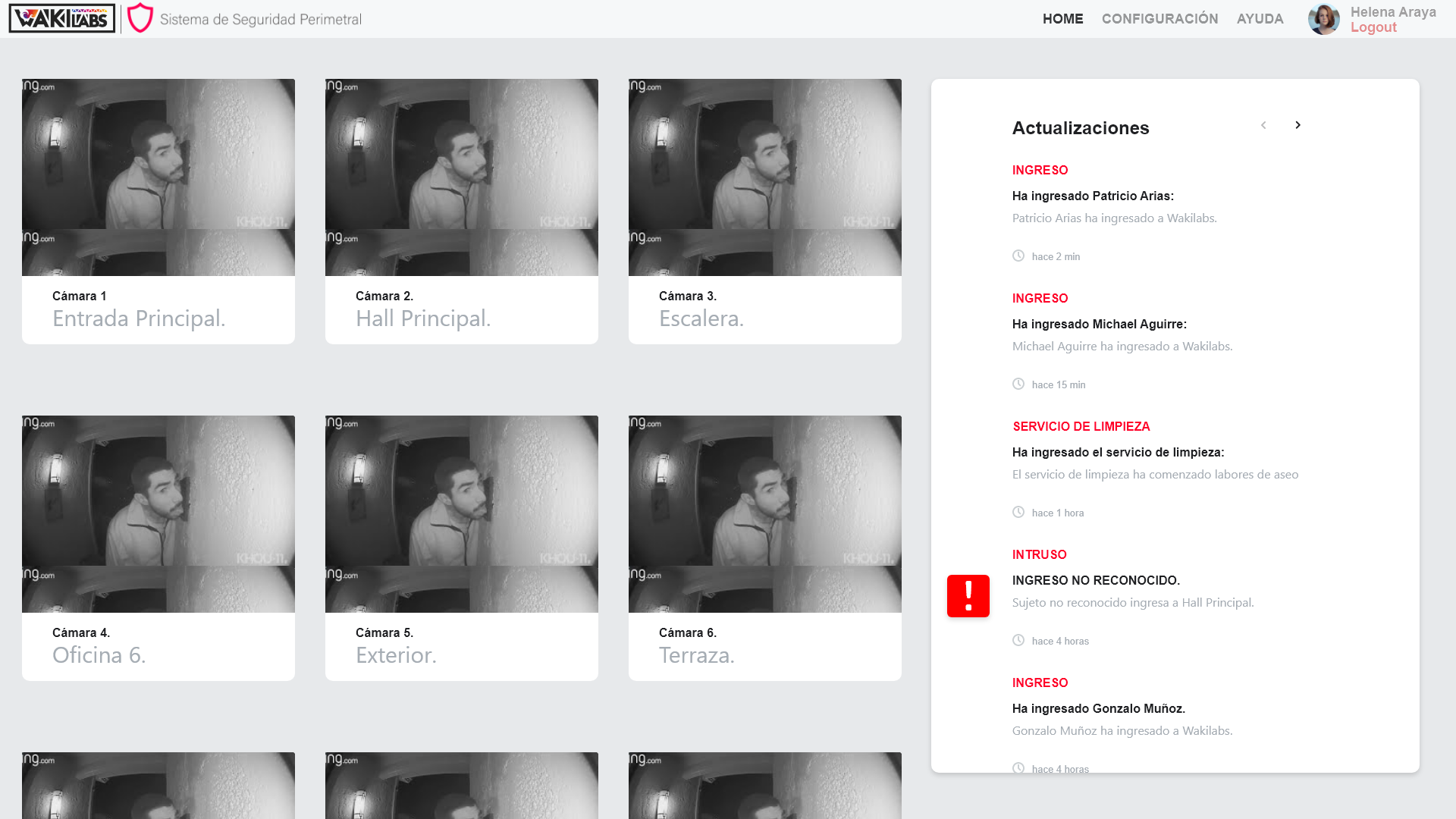
### Figura 15. Modelo de datos.

# Implementación inicial del sistema

A continuación, se explicará el proceso de la implementación inicial del sistema, dividiéndolo en dos partes, el Frontend, que incluye los aspectos visuales del sistema, como son la interfaz de usuario y vistas del sistema; y el Backend, que consiste en el corazón y núcleo de las funcionalidades más importantes del sistema, y que son llamadas por el Frontend para mostrar información importante.

## Frontend

Inicialmente, el grupo de trabajo creó el boceto de una interfaz de usuario para dar inicio al desarrollo del Frontend, y así tener una dirección específica para esta, tal como se puede observar en la Figura 16.



### Figura 16. Vista principal del sistema.

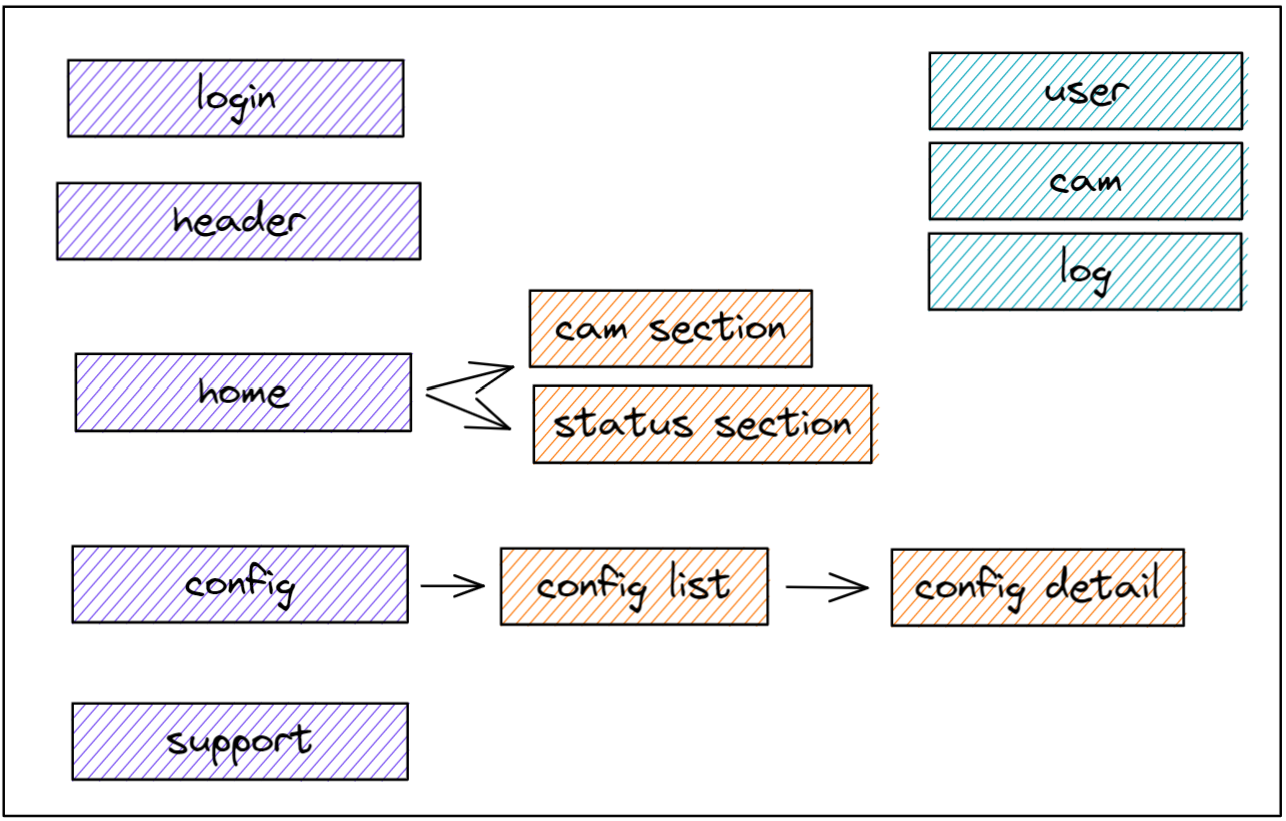
Aquí se pueden apreciar tres secciones principales, la barra de navegación en la parte superior de la imagen, la sección de cámaras y las actualizaciones y notificaciones que llegan al sistema.

Posteriormente, fue planteado por el cliente cambiar las actualizaciones por una sección específica para el estado de los miembros de WAKI, como se puede observar en la Figura 17, siendo verde aquellos miembros que están en el recinto, y en rojo aquellos que no.



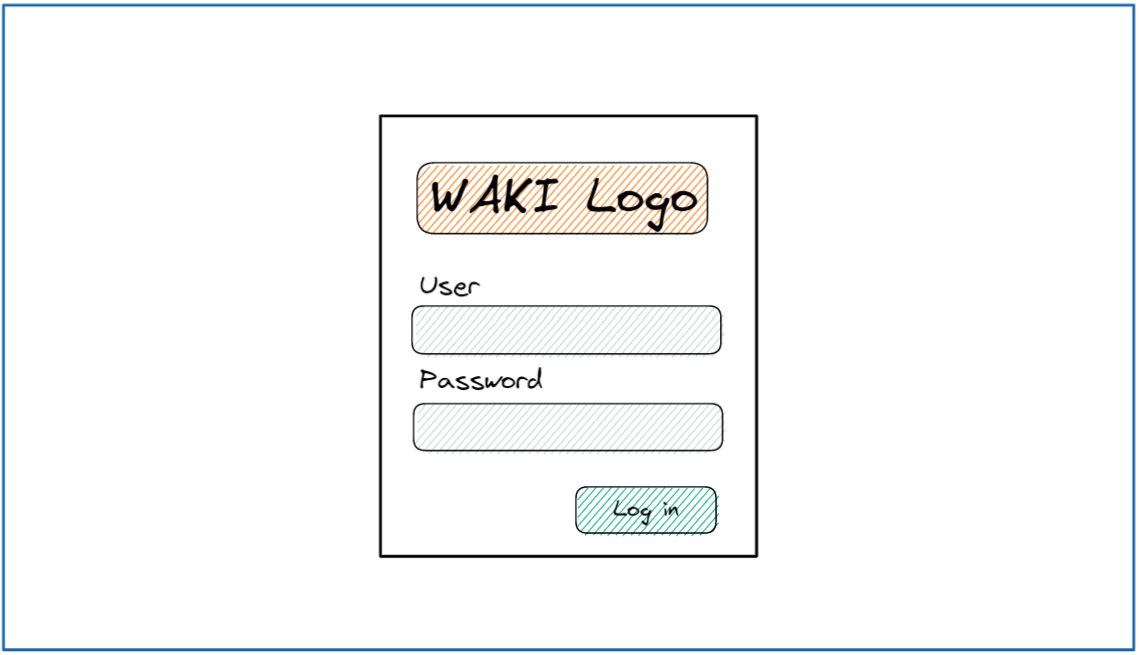
### Figura 17. Boceto de WAKI estado.

A partir de dichos bocetos, se comenzó a estudiar el framework Angular, el cual permitió un acercamiento a través de componentes, como se observa a continuación en la Figura 18. Donde se puede ver de color morado los componentes padres que sirven como vistas principales, de color naranja se tienen los componentes hijos que corresponden a secciones de dichas vistas. Y por otro lado, de color celeste se tienen las clases de objetos a utilizar, en este caso Usuario, Cámara y Registro.

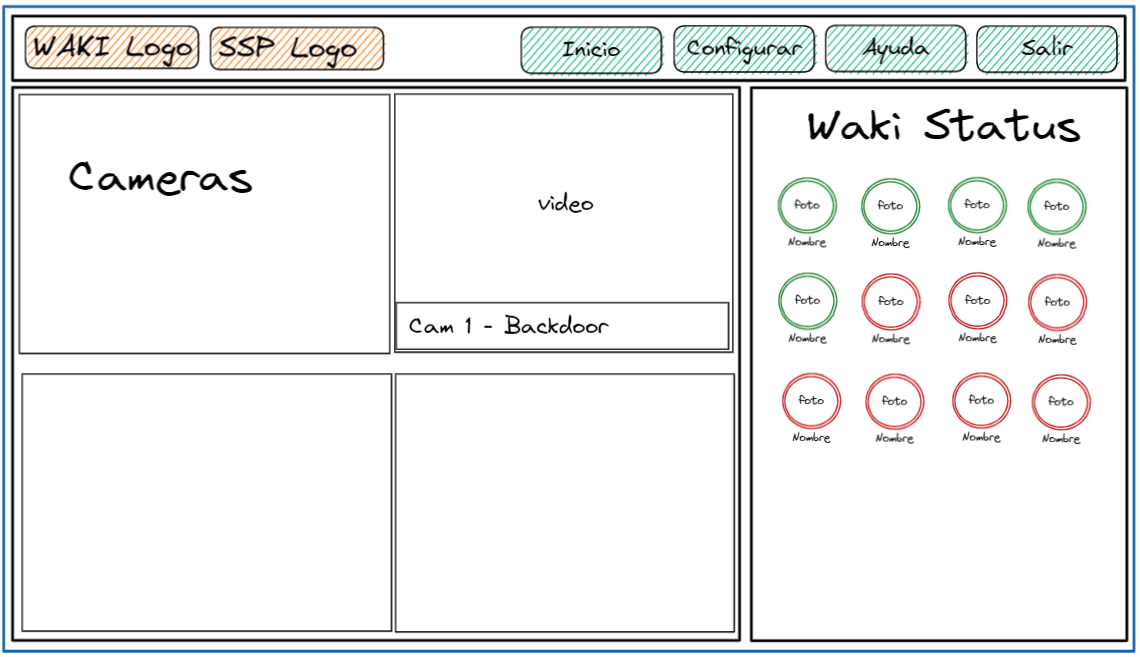


### Figura 18. Componentes en Angular

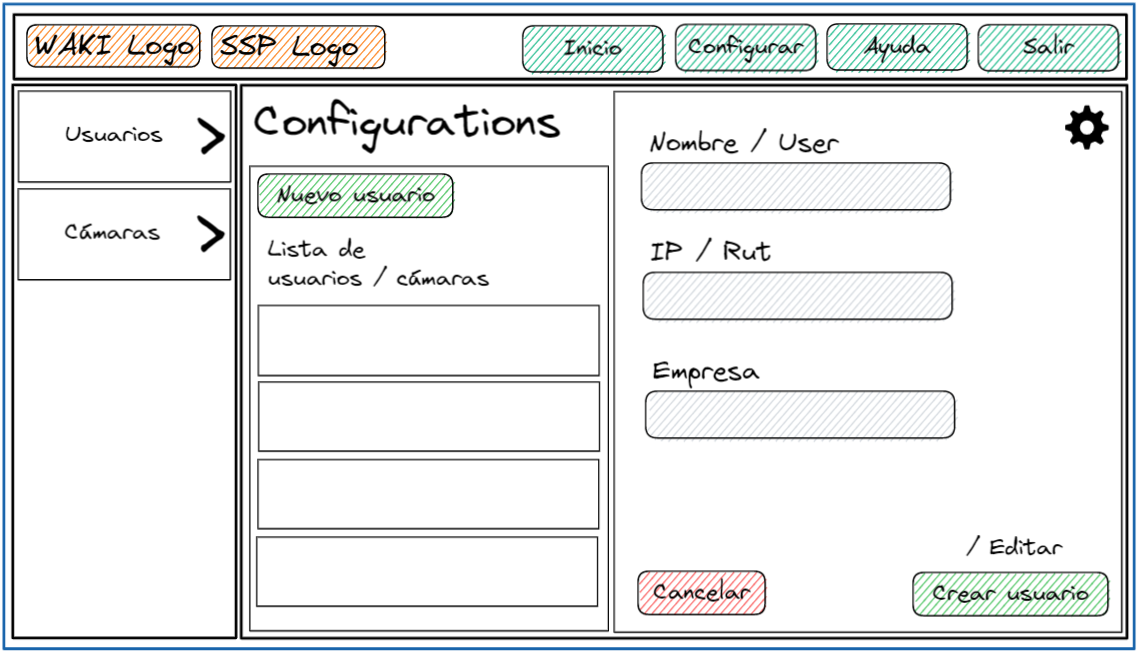
Dichos componentes actúan como piezas del sistema, como pueden ser las vistas de Login, Inicio, Configurar y Ayuda, que se observa en las Figuras 19, 20, 21 y 22. O secciones más pequeñas como la sección de status.



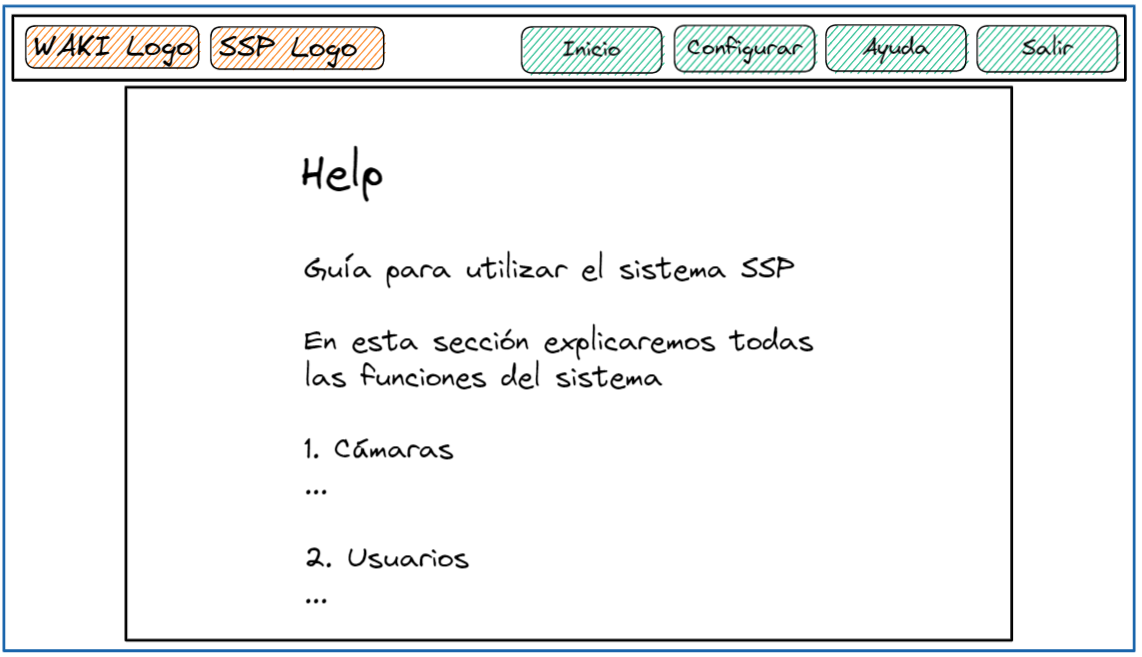
*Figura 19. Bosquejo de vista Login.*



### Figura 20. Bosquejo de vista Inicio.



*Figura 21. Bosquejo de vista Configurar.*

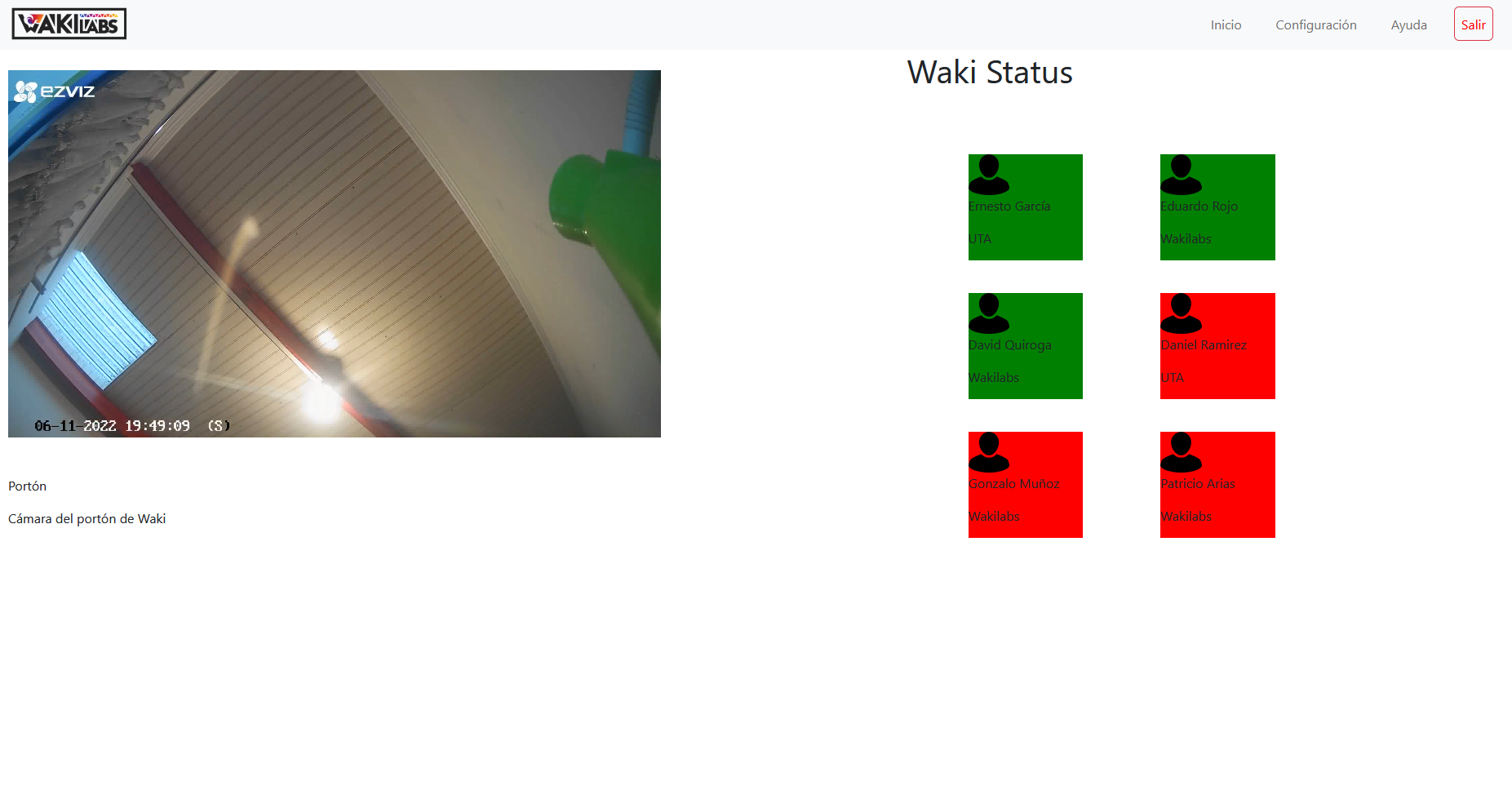


### Figura 22. Bosquejo de vista Ayuda.

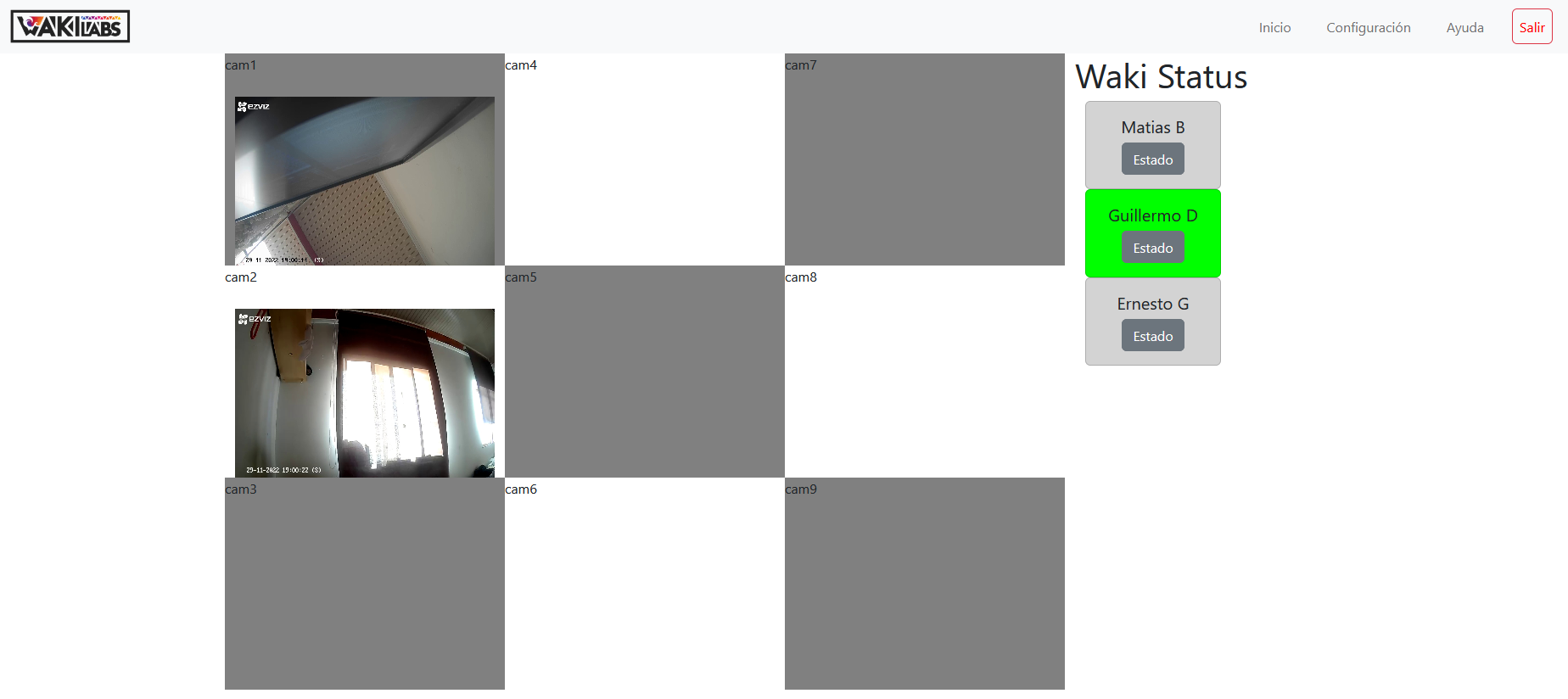
A partir de los bosquejos mostrados anteriormente, se realizó una implementación inicial de dichas vistas en el sistema, que se muestran a continuación.

### Vista Inicio

En las Figuras 23 y 24 se observa la vista Inicio, donde en la parte izquierda se transmite video de la cámara IP en tiempo real, y en la parte derecha se tiene la sección “status”, que contiene el estado de cada miembro en WAKI.



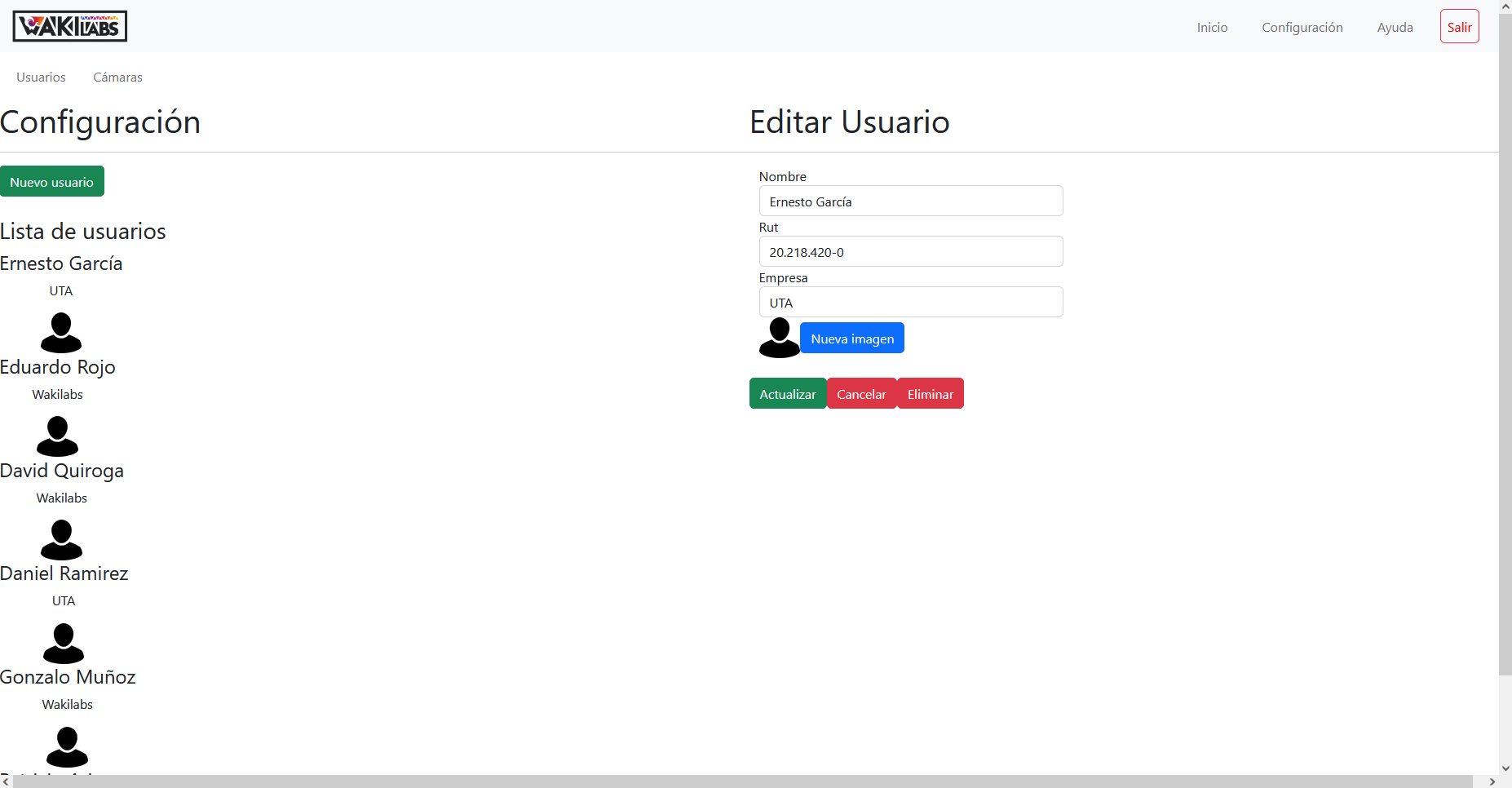
### Figura 23. Implementación inicial de vista Inicio.

**

### Figura 24. Implementación avanzada de vista Inicio.

### Vista Configurar

En la Figura 25 se observa la vista configuración, donde en la parte izquierda tiene la selección de usuarios o cámaras, y en la parte derecha se tiene la modificación de estos, ya sea crear, modificar o eliminar.



### Figura 25. Implementación inicial de vista Configurar.

## Backend

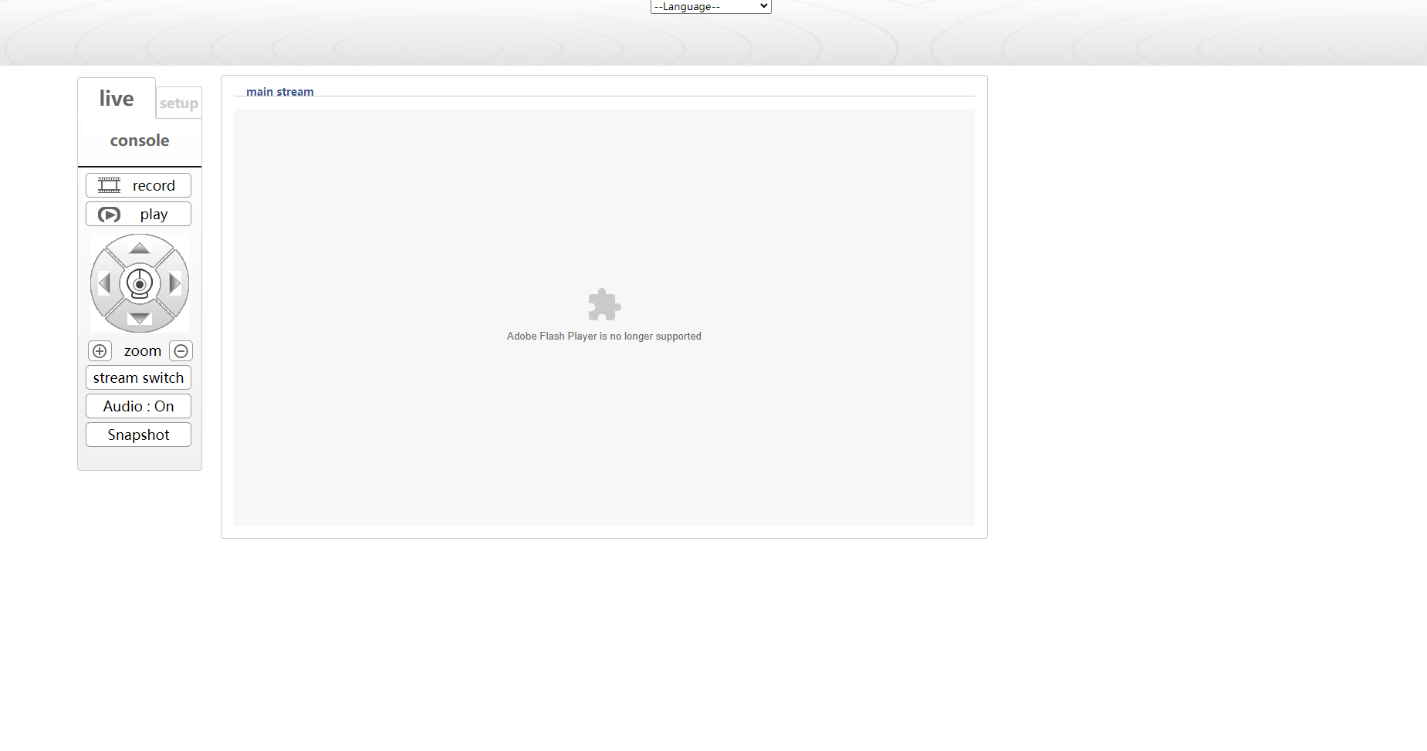
Para el desarrollo del Backend se tuvo que realizar una extensiva investigación para conocer y entender las herramientas a utilizar, y hacer que cumplan los requerimientos del proyecto.

Inicialmente se estuvo trabajando con las cámaras IP de marca VTech de WAKI, como se observa en la Figura 26.



### Figura 26. Cámaras VTech.

Pero se tuvo complicaciones para utilizarlas, debido a la poca o nula documentación encontrada en línea, además de su funcionamiento obsoleto que dificulta el uso apropiado de estas, como se observa en la Figura 27, donde la interfaz de usuario no funciona, ya que el reproductor utilizado por esta ya no es soportado por los buscadores.



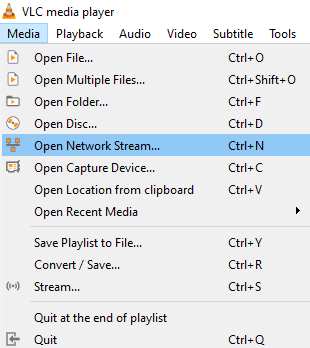
### Figura 27. Interfaz de la cámara.

Debido a esto, el equipo de WAKI Labs permitió comprar nuevas cámaras IP de marca EZVIZ, como se muestra en la Figura 28, con las cuales se comenzó a trabajar.

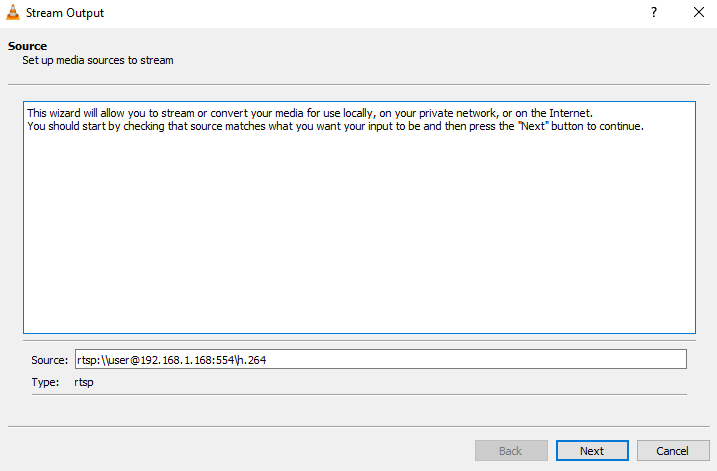


### Figura 28. Cámaras EZVIZ.

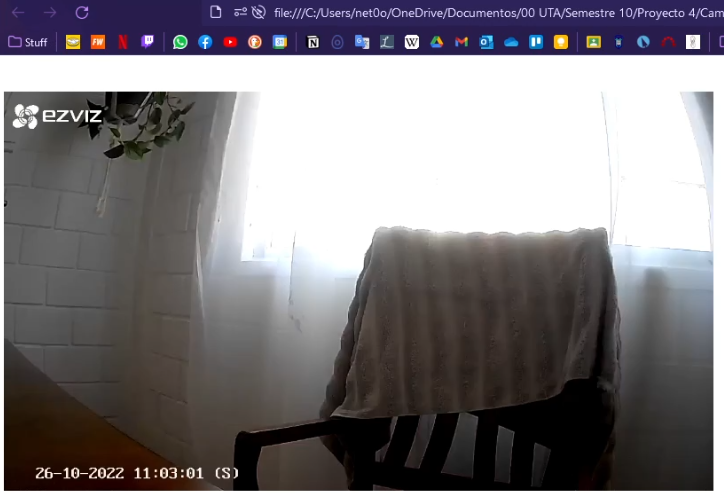
Luego de la investigación realizada acerca del protocolo RTSP, y junto a las nuevas cámaras Ezviz adquiridas, se logró transmitir el video de la cámara, utilizando el software VLC, a través de la función “Network Stream”, como se observa en las Figuras 29, 30 y 31.



*Figura 29. Software VLC.*



### Figura 30. Transmisión de video RTSP en VLC.



### Figura 31. Transmisión de video RTSP.

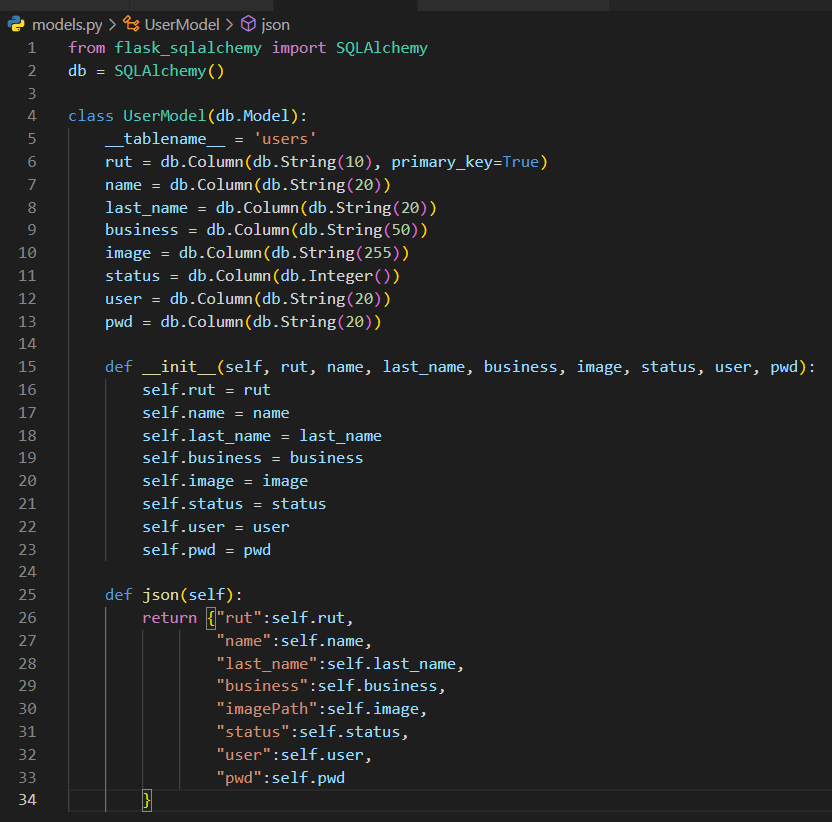
Finalmente, se comentó con el equipo acerca del funcionamiento del “status” para cada usuario, y se acordó que el usuario debe mostrar un código QR a la cámara, para que su estado cambie en el sistema, por lo que se comenzó a estudiar cómo leer códigos QR con OpenCV en Python, y en la Figura 32 se puede observar la lectura de códigos QR ya implementada, a través de la cámara, donde se observa un texto en azul, que aparece en la esquina superior izquierda, mostrando el texto leído, en este caso un rut.



### Figura 32. Lectura de código QR.

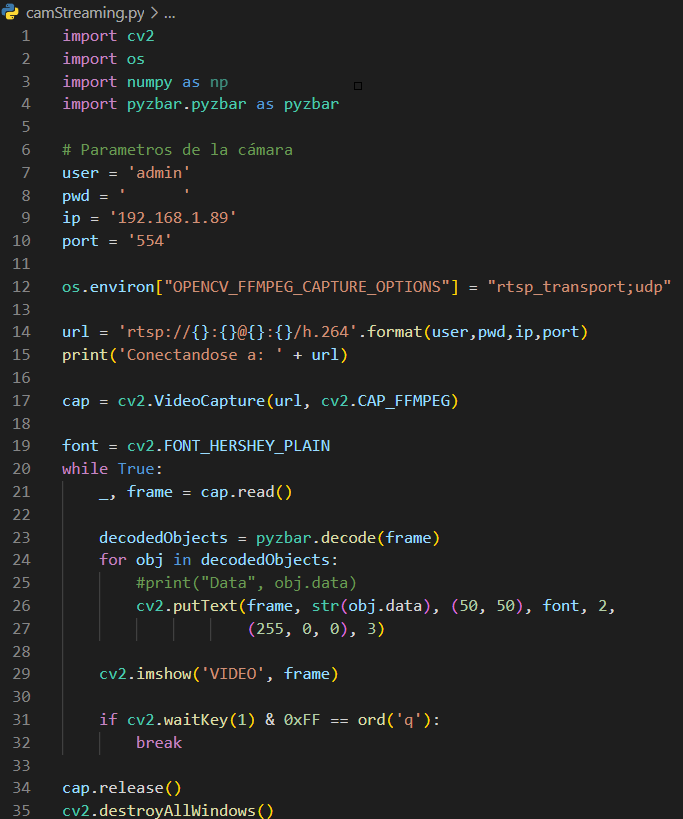
Dado que el Backend fue comenzado recientemente, no se tiene mucho implementado de este, pero de momento se logró:

* La creación de modelos para la base de datos, utilizando la herramienta SQLAlchemy que tiene Flask, como se observa en la Figura 33.



### Figura 33. Modelos para base de datos.

* La transmisión de video utilizando la herramienta OpenCV y Pyzbar para escanear un código QR, como se observa en la Figura 34.



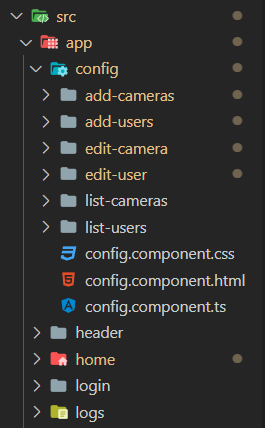
### Figura 34. Escáner de código QR.

# Implementación final del sistema

Luego de las etapas de análisis del problema, diseño del sistema, y estudio de las herramientas necesarias a aprender para llevar a cabo la codificación del sistema, se lograron implementar las funcionalidades principales del sistema, tanto en el Frontend como en el Backend, como se muestra a continuación.

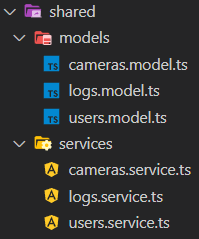
## Frontend

En primer lugar, para el Frontend se consiguió crear un sistema estructurado a partir de los componentes que provee Angular, como se puede apreciar en la Figura 35.



### Figura 35. Estructura de los componentes del Frontend en Angular.

Así como los servicios y modelos creados, que se muestran en la Figura 36, que fueron necesarios para conseguir realizar una conexión entre el Frontend y el Backend, haciendo que este sistema de API REST funcione sin problemas.

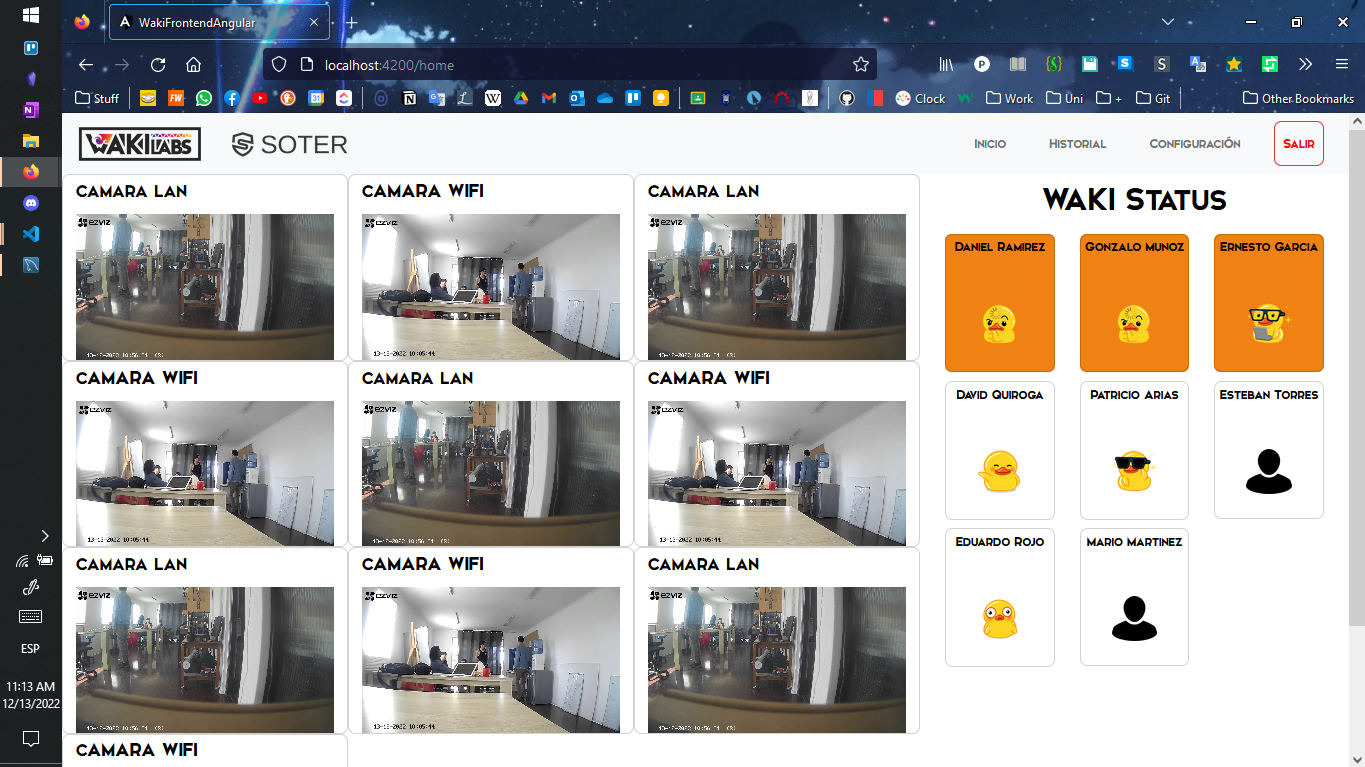


### Figura 36. Estructura de los modelos y servicios del Frontend en Angular.

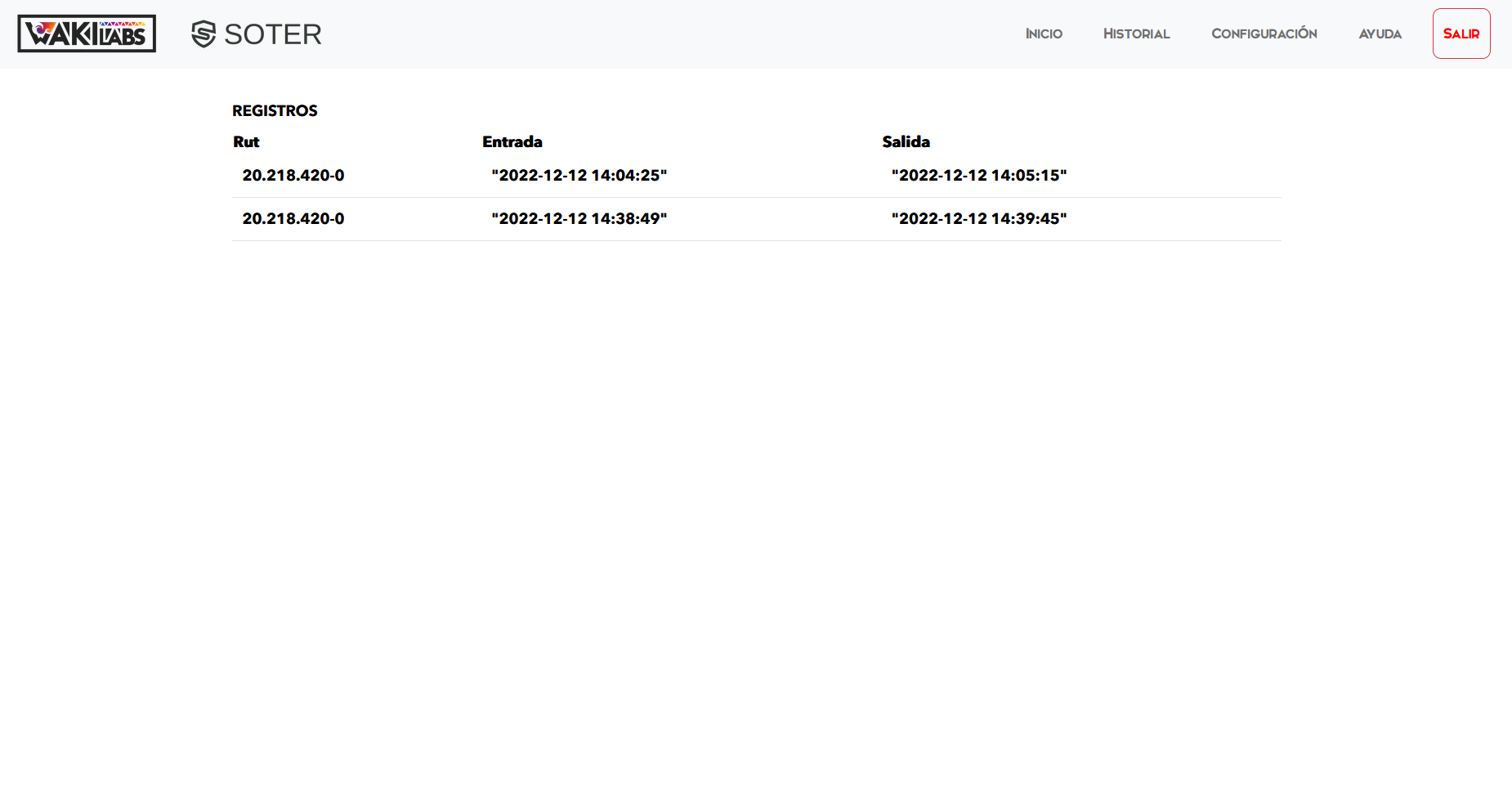
En las siguientes Figuras 37, 38, 39, 40, 41, 42, 43 y 44 se puede observar el Frontend del sistema SOTER finalizado.



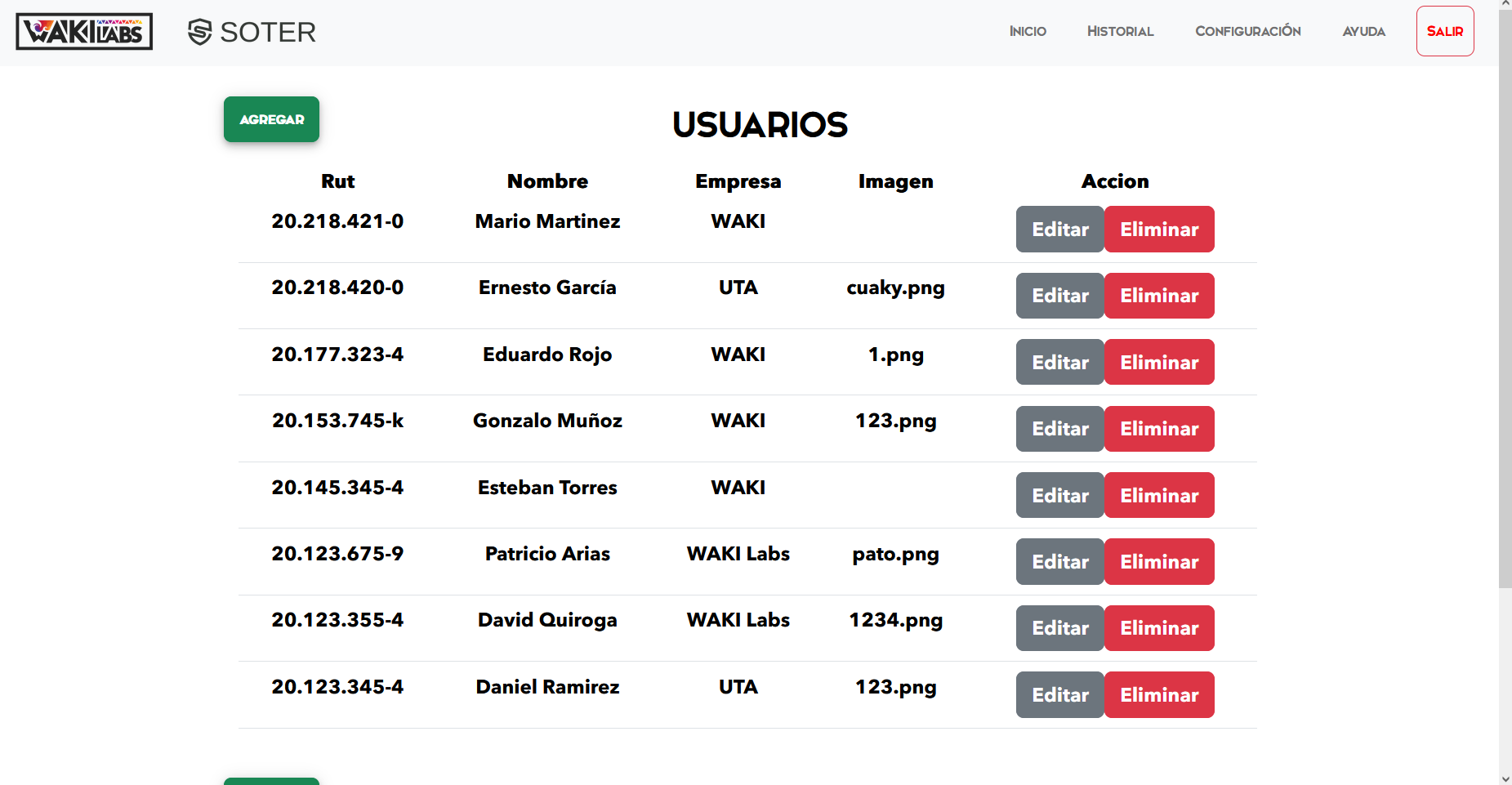
### Figura 37. Vista de login.



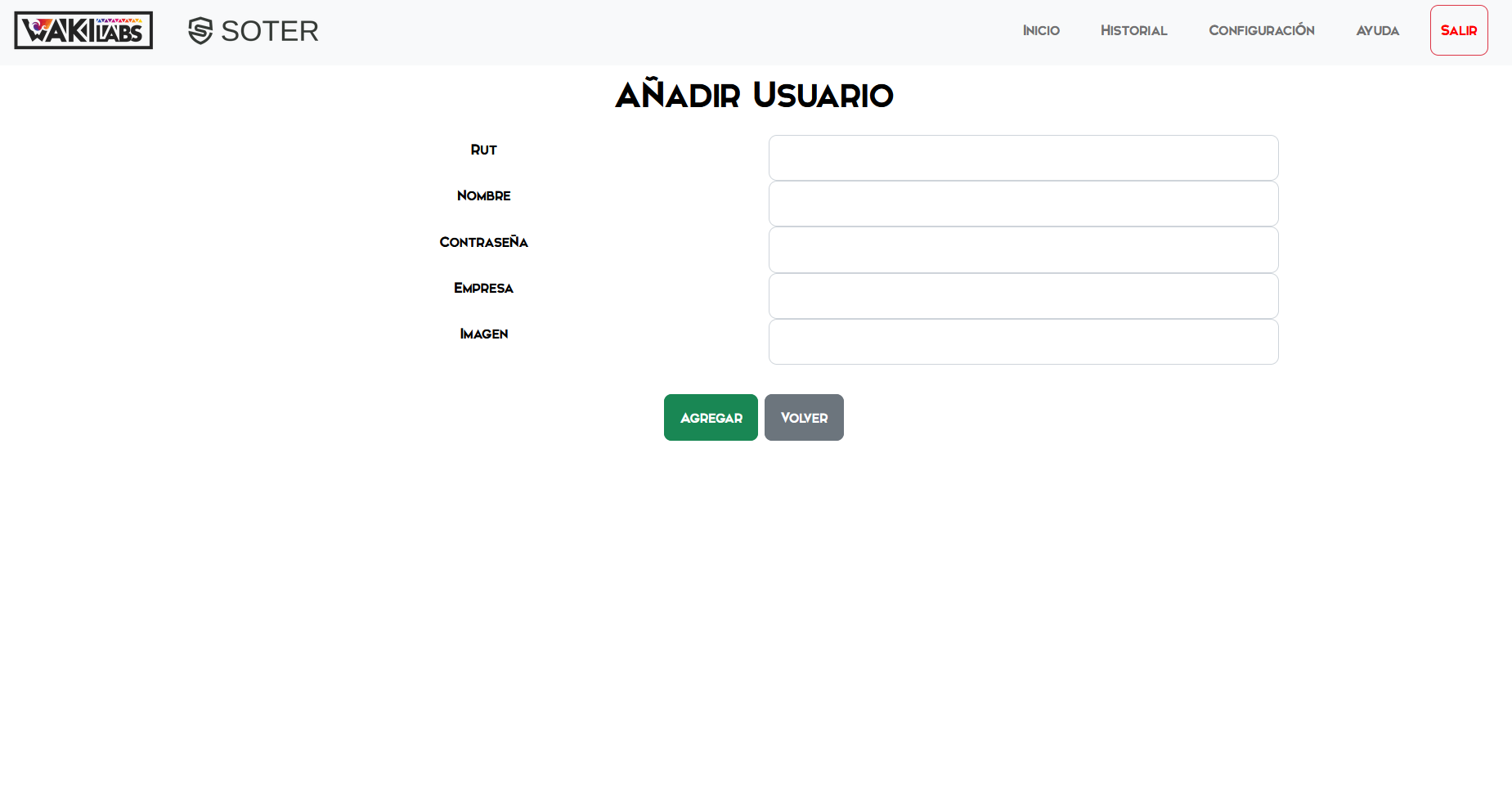
*Figura 38. Vista de login.*

**

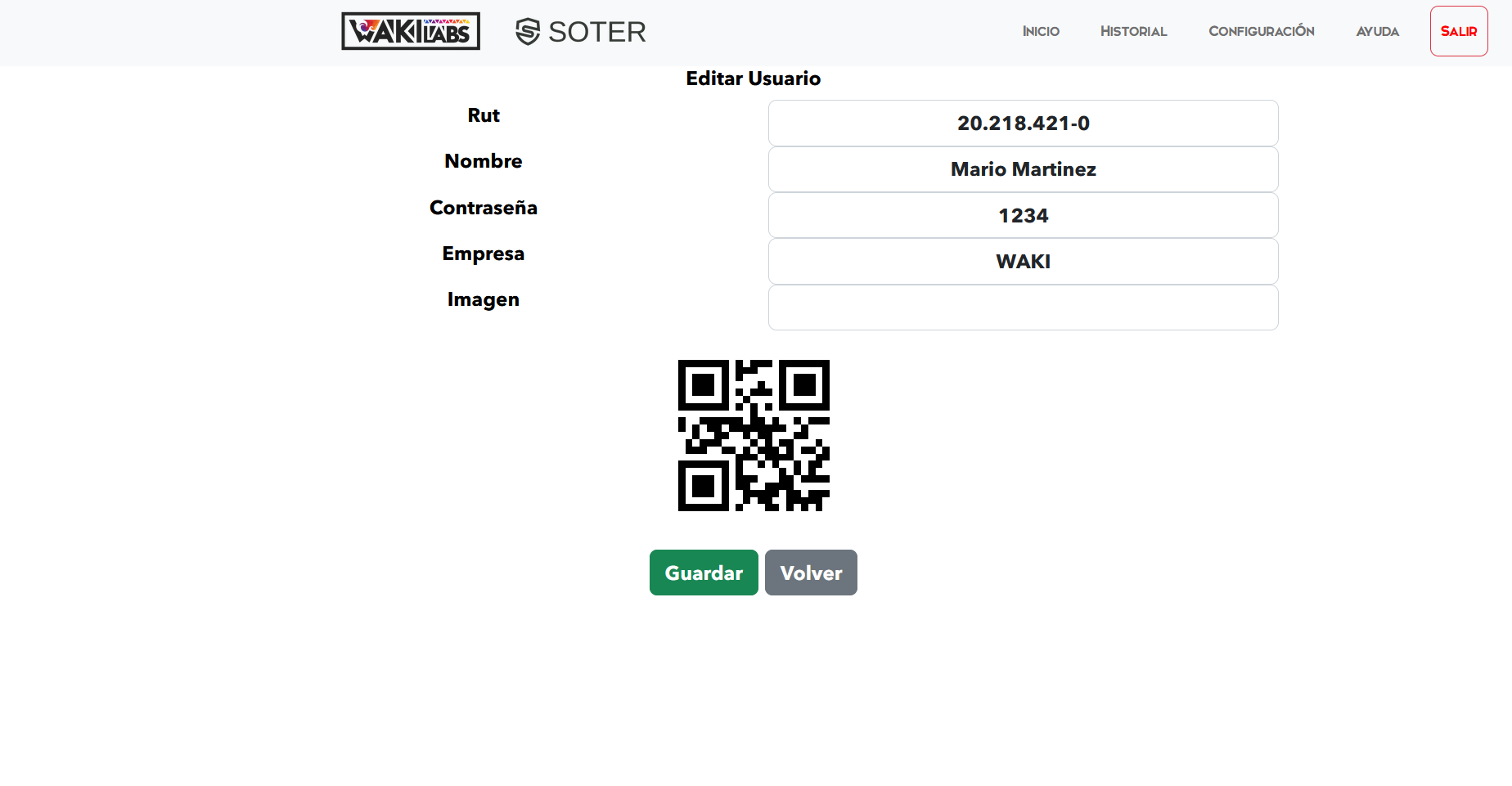
*Figura 39. Vista de historial.*

**

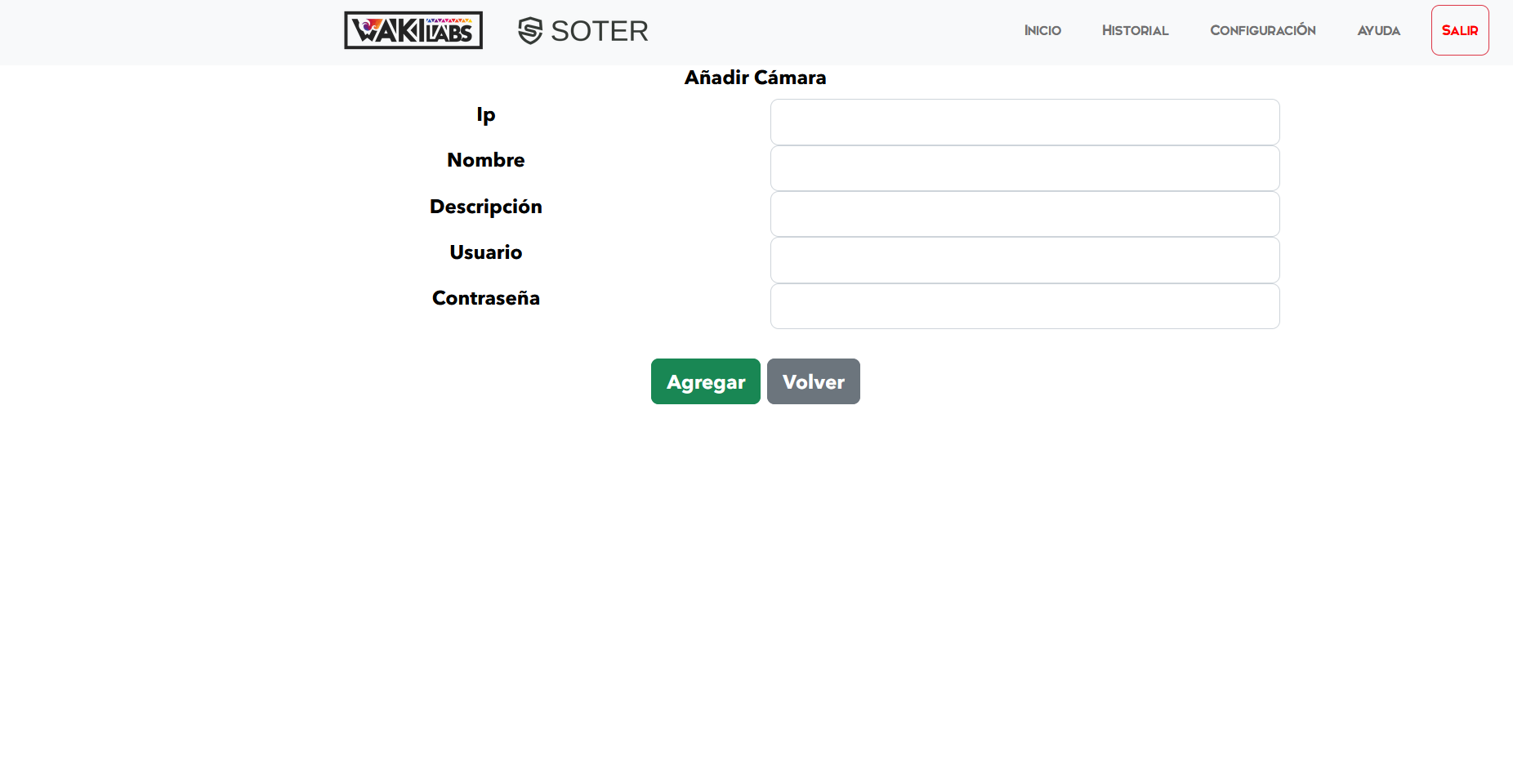
*Figura 40. Vista de configuración.*

**

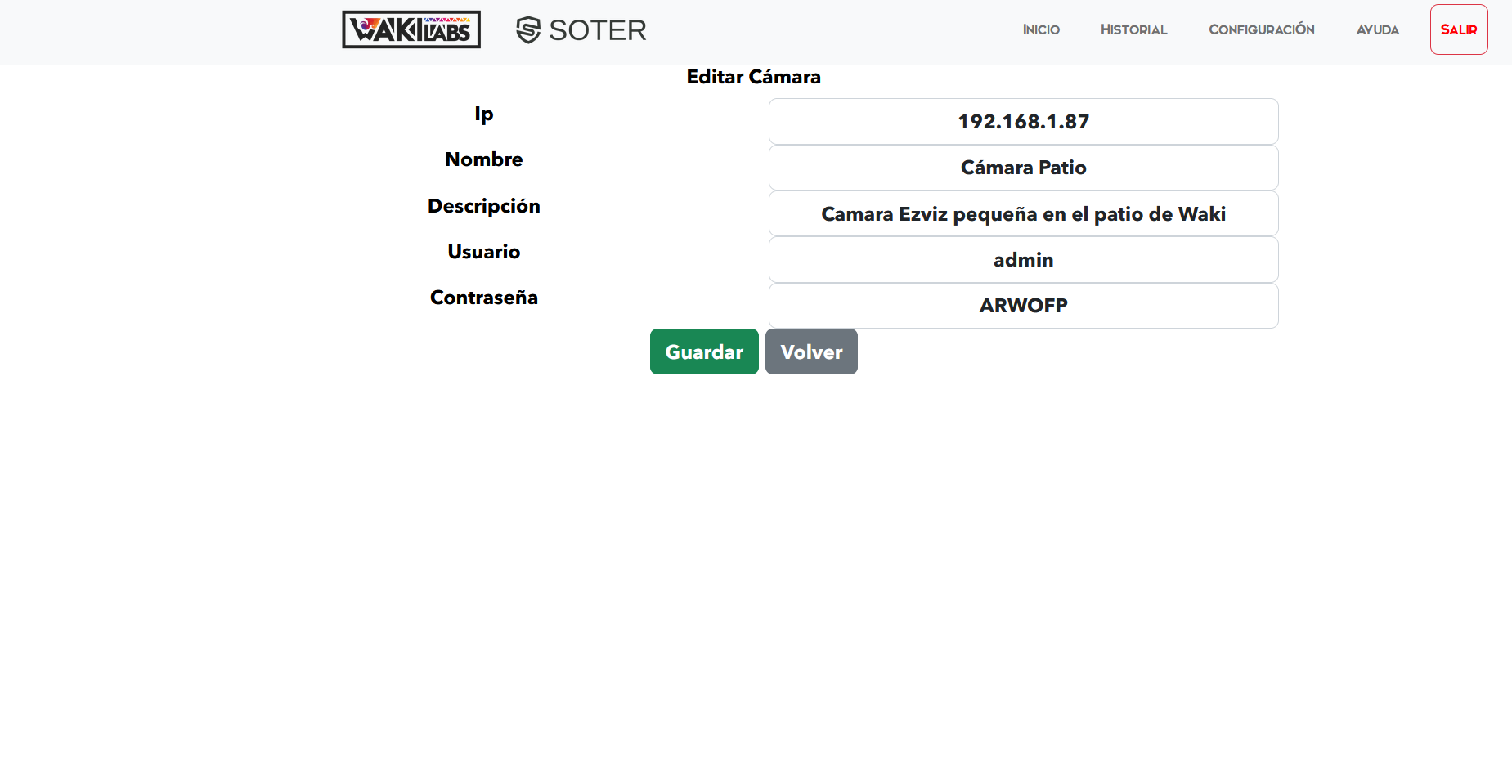
*Figura 41. Vista de añadir usuario.*

**

*Figura 42. Vista de editar usuario.*

**

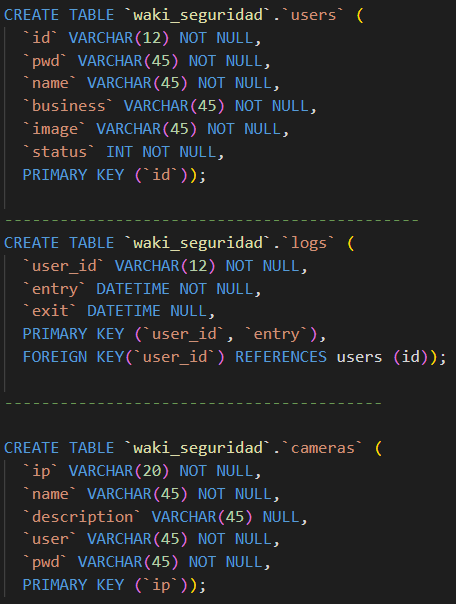
*Figura 43. Vista de añadir cámara.*

**

### Figura 44. Vista de editar cámara.

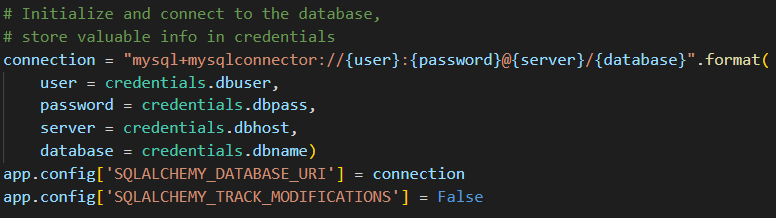
## Backend

Por otro lado, en el backend se trabajó en la creación y conexión a la base de datos MySQL, como se observa en la siguiente Figura 45, en la cual se crean las tablas necesarias para el sistema.



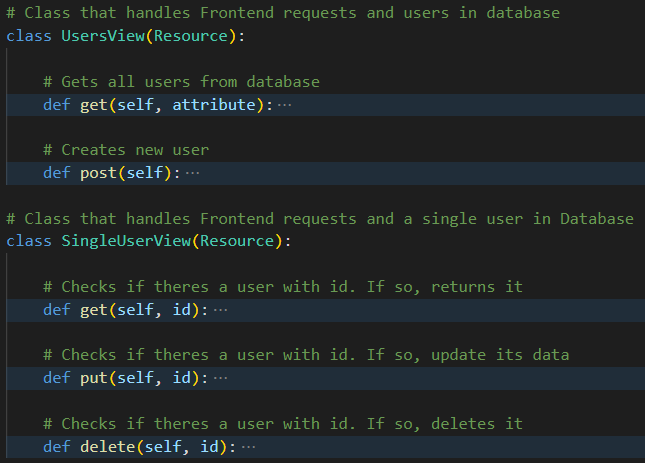
### Figura 45. Creación de tablas SQL.

Luego en la Figura 46 se puede observar la conexión al servidor local de MySQL, utilizando credenciales almacenadas en un archivo privado, para mantener la seguridad del sistema.

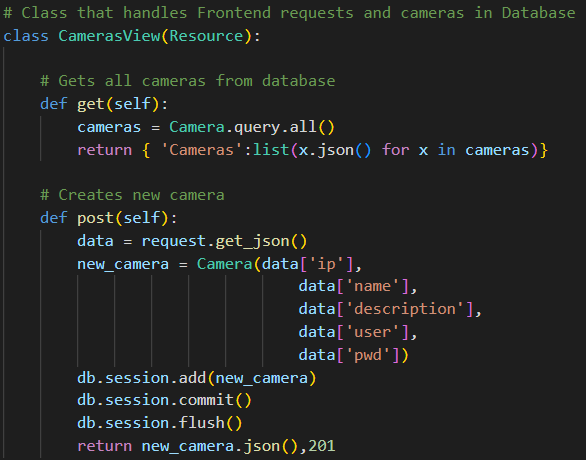


### Figura 46. Conexión a servidor MySQL.

Luego fueron creadas las respectivas clases, con respecto a las tablas creadas anteriormente, para manejar las peticiones de tipo GET y POST, que se envían desde el Frontend, como se puede apreciar en las Figuras 47 y 48.



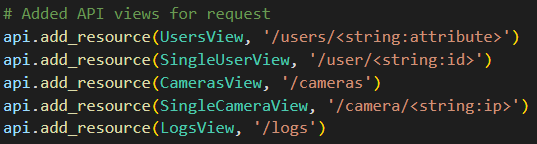
### Figura 47. Clases para recibir peticiones.



### Figura 48. Clase Cámara que recibe peticiones GET y POST.

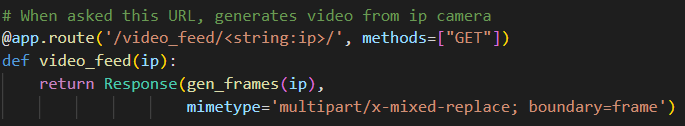
A través de estas clases, se accede a la base de datos y se modifica la información enviada desde el Frontend, como ocurre al Agregar o Editar Usuario.

Dichas clases son luego agregadas a la API utilizada, que se encarga de manejar dichas peticiones a url específicas como “/cameras” o “/logs”, como se muestra a continuación en la Figura 49.



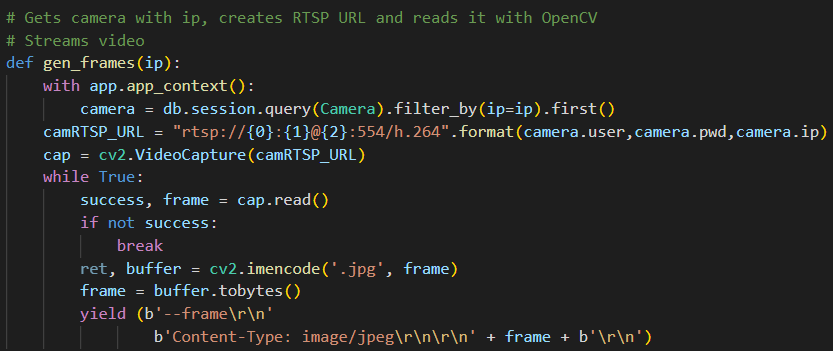
### Figura 49. Agregando clases a la API.

Por otro lado, se tiene el manejo de los múltiples videos de las cámaras IP, a través de la función “@app.route” que genera el video correspondiente de la cámara indicada, utilizando la función “gen\_frames”, que se ve en la Figura 50.



### Figura 50. App route para video de cámaras IP.

Dicha función permite transmitir video de múltiples cámaras al mismo tiempo, debido a que puede trabajar en paralelo, cuando ocurren múltiples llamadas a la ruta, como se puede apreciar en la Figura 51. En resumen, la función se encarga de crear la url RTSP de la cámara IP, a partir de sus datos almacenados en la base de datos, para que luego el video sea capturado por OpenCV y retornado a quien hizo la petición en el Frontend.

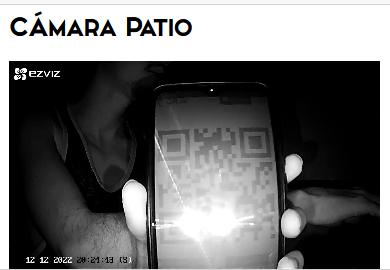
**

### Figura 51. Manejo de múltiples cámaras IP.

# Pruebas del sistema

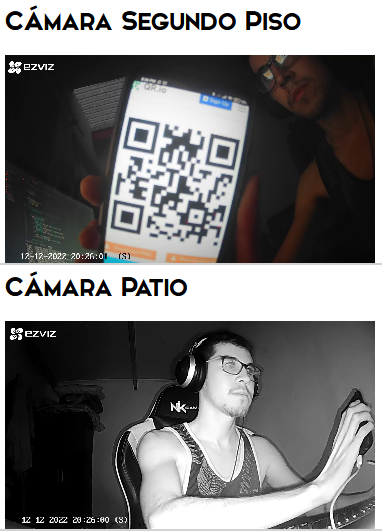
Tal como pudo ser observado en la sección anterior, de implementaciones finales, el sistema se encuentra en un estado bastante completo, aunque todavía faltan detalles específicos a resolver.

Uno de estos problemas aparece al momento de anochecer, cuando la cámara utilizada para leer códigos QR, entra en modo nocturno, como se muestra en la Figura 52.

**

### Figura 52. Cámara IP nocturna leyendo QR de noche.

Mientras que dicho problema no ocurre con la segunda cámara, como se muestra en la Figura 53.

**

### Figura 53. Cámara IP leyendo QR de noche.

# Conclusiones

En esta etapa final del proyecto SOTER, se realizó un trabajo arduo para conseguir implementar el sistema a tiempo, dadas las restricciones y fechas límites para este. Esto en parte a que las primeras etapas del proyecto fueron más lentas y se enfocaron en aprender nuevas herramientas y diseñar el sistema, mientras que esta etapa se concentró en desarrollar toda la codificación necesaria para hacer funcionar el sistema.

El trabajo con la empresa WAKI Labs continuó sin mayores problemas, dado a que no hubo ningún cambio repentino de requerimientos, como en etapas anteriores. Además, se contactó oportunamente a los miembros del equipo cuando se tenían dudas y se necesitaba ayuda con algunas tecnologías.

Cabe destacar la extensa investigación, sin la cual este sistema no podría haber sido llevado a cabo, y a todo el estudio realizado para aprender herramientas nuevas como Flask, Angular e incluso los protocolos RTSP.

Por último, se destaca que el proyecto pudo ser completado en su mayor parte, con ciertos detalles puntuales siendo trabajados durante un breve tiempo extra, según lo acordado con la empresa, con el fin de entregar un producto bueno y estable.

Obs: Excelente informe, tiene todo lo que exigible, hay problemas de redacción menor

Algunas figuras o diagramas no explicados

Muy buen trabajo

# Referencias

[1] Sitio oficial de WAKI Labs

[https://WAKIlabs.cl/site/](https://wakilabs.cl/site/)

[2] Sitio oficial de Angular

<https://angular.io/start>

[3] Servicio web para empotrar video en vivo de cámara IP en un sitio web

<https://www.ipcamlive.com/faqs>

[4] Uso de OpenCV para el manejo de Cámaras IP

<https://youtu.be/EkqhIeSZGN8>

[5] Manejo de múltiples cámaras con Flask y OpenCV

<https://stackoverflow.com/questions/59554042/handle-multiple-cameras-using-flask-and-opencv>

[6] Generación de códigos QR en Angular

<https://remotestack.io/angular-qr-code-generator-tutorial-with-example/>

[6] Escaneo de códigos QR con OpenCV

<https://youtu.be/-4MPtERPq2E>

[7] Crear aplicación con Python-Flask y Angular

<https://youtu.be/IdhtcdCbTVk>

<https://youtu.be/MaiAeGVwVU4>

[8] Aplicación con Angular, Python-Flask y SQLAlchemy

<https://auth0.com/blog/using-python-flask-and-angular-to-build-modern-apps-part-1/>

[9] Instalación y uso de MySQL Server y Workbench

<https://youtu.be/Sv2vBT3dtvQ>

[10] Conexión entre Flask y SQLAlchemy

<https://youtu.be/4bIB86LRX7I>

[11] Repositorios de GitHub en privado

<https://github.com/Sutams/waki-frontend-angular>

<https://github.com/Sutams/waki-backend-flask>