DOCUMENTO DE AVANCE: PROTOTIPO No.2

Sistema de Ingreso por Reconocimiento Biométrico en la Universidad del Cauca

Universidad
del Cauca

Presentado Por:

Carlos Daniel Collazos Zambrano

Luis Miguel Ortiz Muñoz

Cristian Felipe Bolaños Ortega

Jefferson Danilo Noguera

Presentado a:

Mary Cristina Carrascal Reyes

Hermes Fabian Vargas Rosero

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones

Ingeniería Electrónica y Telecomunicaciones

Laboratorio 4 de electrónica.

Popayán, octubre de 2024

Índice

1. INTRODUCCIÓN

Este informe documenta el avance en el desarrollo del segundo prototipo del "Sistema de Ingreso por Reconocimiento Biométrico" implementado en la Universidad del Cauca. Este proyecto tiene como objetivo mejorar los procesos de control de acceso en las instalaciones de la institución mediante la adopción de tecnologías biométricas avanzadas. En la fase actual, se han mantenido los dispositivos del primer prototipo, que incluyen un lector de huellas dactilares y un microcontrolador con conexión Ethernet. La principal mejora en esta entrega se centra en la optimización de la captura de huellas dactilares, extracción de características, y la transmisión eficiente de los datos al servidor mediante Ethernet.

En el servidor, se ha integrado un modelo de machine learning para la clasificación de las huellas dactilares, permitiendo autenticar de manera precisa al usuario. Los datos de los usuarios están asociados a sus huellas a través de una interfaz web de gestión, que facilita el registro de usuarios por parte del administrador. Esta segunda fase del proyecto busca mejorar la precisión del proceso de captura, la transmisión de datos y el rendimiento del modelo de clasificación. A través de estas mejoras, el sistema avanza hacia una solución completa, eficiente y segura, que permitirá controlar el acceso de los usuarios a las instalaciones de la universidad de manera automatizada.

2. OBJETIVOS

Objetivo general: Desarrollar un sistema de acceso biométrico mediante huellas dactilares para mejorar el control de acceso a la Universidad del Cauca, integrando hardware de captura de huellas, procesamiento de datos mediante machine learning y una plataforma web para la gestión y autenticación de usuarios.

Objetivos específicos:

- Configurar e integrar un lector de huellas dactilares con un microcontrolador para la captura, procesamiento, y envío de templates.
- Desarrollar un modelo de clasificación mediante machine learning que permita la autenticación de usuarios a partir de los templates de huellas dactilares.
- Integrar la plataforma web para la gestión de usuarios y visualización de registros de acceso.
- Garantizar la seguridad y precisión en el proceso de autenticación mediante pruebas y optimización del modelo.

3. FASES DESARROLLADAS

3.1 Configuración y programación del lector de huellas dactilares

- **Lectura de huella dactilar y extracción de vector de características**
  - Logros: Haciendo uso de la librería AdafruitFingerprint que es compatible con el sensor, se captura y procesa la huella dactilar, al mismo tiempo se extrae un vector de características o template, el cual es una cadena hexadecimal que representa la informacion clave de la huella.
  - Desafíos: La librería obliga a realizar la captura de la huella en dos fases, por lo que era necesario colocar el dedo dos veces siempre que se quiera registrar o autenticar. Es preocupante la calidad de las huellas dactilares debido a la calidad del sensor al ser tan económico.

- **Transmisión de datos al servidor a través de ethernet**
  - Logros: Con el servidor ya implementado en la primera etapa, y acceso al microcontrolador WT32 ETH01, se lograron hacer las pruebas pertinentes para verificar el funcionamiento del servidor, enviando una solicitud POST en la cual se encuentra el template de la huella.
  - Desafíos: La falta de documentación del lector de huellas dactilares dificulta el desarrollo de proyecto, ya que retrasa la configuración de ciertas

funciones. También, la imposibilidad de hacer pruebas de conexión con ethernet en la red de la universidad, debido a que esta misma bloquea las direcciones MAC de dispositivos desconocidos.

3.2 Integración de aplicación web con la base de datos

- **Consultas a la base de datos desde la interfaz web**
  - Logros: Con la base de datos funcional en mongoDB atlas, se implementa la funcionalidad de observar los datos almacenados en tiempo real desde la aplicación web en la opción de gestionar usuarios.
  - Desafíos: La falta de conocimiento y familiaridad con mongoDB atlas y bases de datos noSQL, ya que anteriormente los desarrolladores no habían trabajado con estas herramientas.

- **Backend de registro de usuarios y huellas dactilares integrado con la base de datos**
  - Logros: Desde la aplicación web se podrá transmitir los datos de registro de los usuarios (nombre, id, rol, huella) a la base de datos a través del servidor.
  - Desafíos: La integración de la aplicación web con el servidor puede generar problemas de compatibilidad, también, ya que la huella dactilar proviene del microcontrolador, es necesario asociar la información del usuario con la huella dactilar de alguna forma.

3.3 Registro y autenticación de huellas dactilares usando ML

- **Descripción e implementación del modelo de ML**
  - Logros: Se toma como base el modelo Random Forest Classifier disponible en la biblioteca scikit-learn, para hacer la clasificación de los template de huella dactilar de una forma eficiente a través de árboles de decisión.
  - Desafíos: El formato hexadecimal del template de la huella genera conflictos para el tratamiento de datos por parte del modelo, por lo que resulta necesario convertir este formato a uno que sea compatible y de longitud fija como un vector numérico normalizado.

    Otro desafío importante en la clasificación de los templates por parte del modelo fue elegir un umbral de probabilidad óptimo para que la verificación de la huella sea optima, al ser un valor entre 0 y 1, elegir el valor que mejor se adecue a nuestra implementación resulta complicado.

4. AVANCES SEGUNDO PROTOTIPO

4.1 Alcance y requisitos planteados

Según el primer documento de diseño del proyecto se tiene que el alcance del segundo prototipo es el siguiente:

- **Programación y configuración del lector de huella dactilar:** Conectar y configurar el lector de huellas dactilares para capturar huellas dactilares y exportarlas en un formato manejable por el modelo de machine learning para su posterior almacenamiento en la base de datos.
- **Implementación del microcontrolador:** Programación para integrar el microcontrolador con el lector de huellas dactilares y exportar los datos de la huella en un formato manejable con el servidor y transferible vía ethernet.
- **Captura y registro de huellas dactilares reales:** Actualizar la lógica de captura y autenticación para utilizar datos reales de los lectores. También, permitir el registro de huellas dactilares reales usando el lector físico.

Y los requisitos esperados para este segundo prototipo son los siguientes:

- Permitir al administrador registrar los usuarios y sus huellas dactilares en el sistema, haciendo uso del lector de huellas real.
- Los usuarios podrán autenticarse luego de haber sido registrados en el sistema, el servidor deberá procesar esta autenticación.
- Visualización de registros y accesos reales en el sistema a través de la aplicación web.

4.2 Características del segundo prototipo

- **Integración del lector de huellas con el microcontrolador (Dispositivo de acceso)**
  Permite al usuario colocar su dedo en el lector, después de un instante se procesa la huella e imprime una cadena hexadecimal representando los puntos clave de la huella. Esto es posible gracias a la librería disponible en Arduino.

- **Dispositivo de acceso**
  El lector de huella integrado con el microcontrolador representan el dispositivo

de acceso completo, este dispositivo se conecta a la red cableada a través de ethernet, de esta forma se conecta al servidor para transmitir los datos de la huella dactilar obtenida.

- **Modelo de machine learning para la clasificación y autenticación de templates de huellas dactilares:**

  El modelo Random Forest es adecuado para la clasificación de templates de huellas dactilares debido a las siguientes razones:

  - ***Capacidad de Manejar Grandes Conjuntos de Datos:*** Los datos de huellas dactilares suelen ser de alta dimensionalidad, ya que las características (minucias, texturas, etc.) pueden extraerse de diferentes puntos. Random Forest es eficiente para manejar datos de muchas características.
  - ***Tolerancia a Ruido y Variabilidad:*** Las huellas dactilares pueden variar debido a factores como la presión, humedad o imperfecciones. Random Forest es robusto frente al ruido en los datos, lo que le permite mantener un buen rendimiento incluso si las huellas tienen pequeñas imperfecciones.
  - ***Generalización y Diversidad:*** Al usar múltiples árboles de decisión y muestras aleatorias de los datos, Random Forest tiende a generalizar mejor y es menos propenso a sobre ajustar, lo que es crucial para la autenticación, ya que las huellas pueden variar ligeramente entre escaneos.

4.3 Porcentaje de cumplimiento de requisitos del primer prototipo

- Permitir al administrador registrar los usuarios y sus huellas dactilares en el sistema, haciendo uso del lector de huellas real.
  Porcentaje de cumplimiento: 97%
  Con el lector de huellas ya disponible y la aplicación web funcionando, el administrador puede solicitar los datos del usuario, ingresarlos en la interfaz de gestión de usuarios y solicitar al microcontrolador que capture la huella dactilar y la envie en el menor tiempo posible. Faltan optimizar detalles con respecto a la integración completa con la base de datos en este proceso.

- Los usuarios podrán autenticarse luego de haber sido registrados en el sistema, el servidor deberá procesar esta autenticación.
  Porcentaje de cumplimiento: 85%
  El proceso de autenticación y clasificación de usuarios depende de la capacidad del modelo de machine learning para identificar los templates, el modelo debe ser entrenado con una cantidad de templates de huellas dactilares adecuado, por el momento no se cuenta con muchas huellas.

- Visualización de registros y accesos reales en el sistema a través de la aplicación web.
  Porcentaje de cumplimiento: 90%

  La aplicación web permite hacer consultas a la base de datos, por lo cuál me va mostrar las colecciones de usuarios, accesos y huellas en las que se hayan almacenado datos. Por el momento, solamente se cuenta con registro de usuarios, debido a que los accesos no serán implementados por completo sino hasta la tercera entrega.

5. EJECUCIÓN DEL PROYECTO

A continuación, se explica de forma detallada como se elaboraron las fases desarrolladas de este segundo prototipo. Cabe recalcar que la mayoría del código se representará por medio de diagramas con el fin de un mejor entendimiento. En caso de requerir la visualización completa de los códigos realizados, dirigirse al repositorio GitHub del proyecto en [1].

5.1 Configuración y programación del lector de huellas dactilares

La primera fase del proyecto consistio en la integracion del sensor de huellas dactilares con el microcontrolar y su correcta programacion para la captura y procesamiento de huellas. Se utilizó la librería AdafruitFingerprint para interactuar con el lector, facilitando la adquisición de los datos de la huella en forma de un template hexadecimal.

Conexión del hardware

El lector de huellas dactilares fue conectado al microcontrolador WT32 ETH01, aprovechando su capacidad para transmisión de datos a través de ethernet. El lector fue montado para obtener huellas dactilares de usuarios que serán procesadas posteriormente.

Captura de huella y extracción de características

Para la captura, el usuario debía colocar su dedo dos veces consecutivas en el sensor, debido a las limitaciones de la librería utilizada. Esto era admisible en registro, sin embargo, es ineficiente para una autenticación rápida, por lo que se modificó la librería para realizar la captura de la huella de forma más rápida, capturando las dos veces necesarias pero sin levantar el dedo y en un tiempo de aproximadamente 10 ms, lo que será inperceptible para el usuario.

Luego de la captura, se genera el template de la huella, este template es una representación hexadecimal de las características clave de la huella, que más adelante será transformado para ser procesado por el modelo de machine learning.

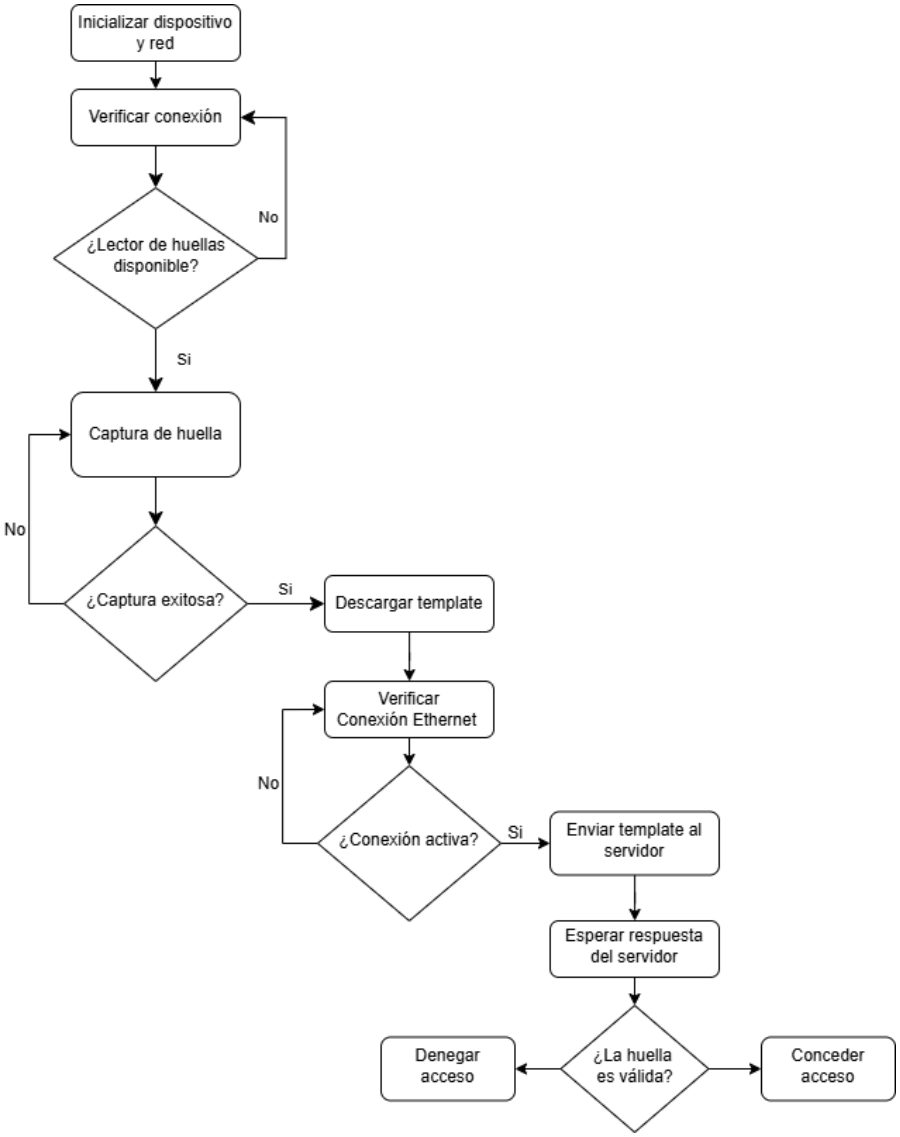

Figura 1 Diagrama de flujo código de autenticación para el sensor

A pesar de que el lector funciona adecuadamente, se enfrentaron problemas con la calidad de las huellas obtenidas, ya que el sensor utilizado es económico y no posee alta precisión. Sin embargo, se buscó optimizar el proceso de captura ajustando parámetros como el tiempo de lectura y la validación del template para asegurar un rendimiento aceptable.

5.2 Integración de la aplicación web con la base de datos

La segunda fase del proyecto involucró la implementación del sistema para gestionar los usuarios creado en el primer prototipo, que incluye el registro de usuarios y huellas dactilares en la base de datos.

Interfaz de gestión de usuarios

Haciendo uso de la interfaz de gestión de usuarios en la página web, se permite registrar a los usuarios ingresando información personal como: nombre, identificación y rol en la universidad.

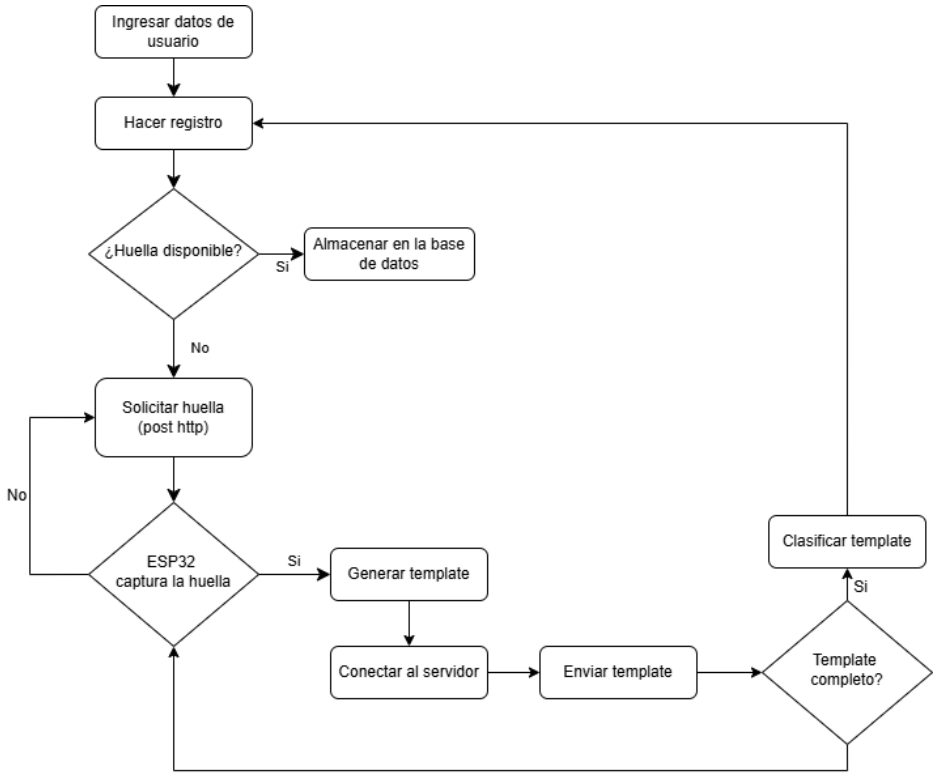

Figura 2 Diagrama de flujo proceso de registro en la página web

Conexión con la base de datos

Utilizando MongoDB Atlas como base de datos en la nube, se configuró la integración del backend para almacenar los templates de huellas dactilares y la información de los usuarios (nombre, ID, rol). Desde la interfaz web, el administrador puede consultar los datos de la base de usuarios, así como visualizar los vectores de características de las huellas dactilares.

Uno de los mayores retos fue la falta de familiaridad del equipo con bases de datos noSQL como MongoDB. Esto implicó un proceso de aprendizaje que

ralentizó la fase inicial, pero que se superó a medida que se avanzaba en la implementación del backend.

5.3 Diseño PCB del módulo de acceso

Para el diseño de la PCB que será implementada en el modulo de acceso, se uso la herramienta KiCad. Primero se realizaron las respectivas conexiones lógicas con el fin de obtener el circuito esquemático de la PCB como se observa a continuación:

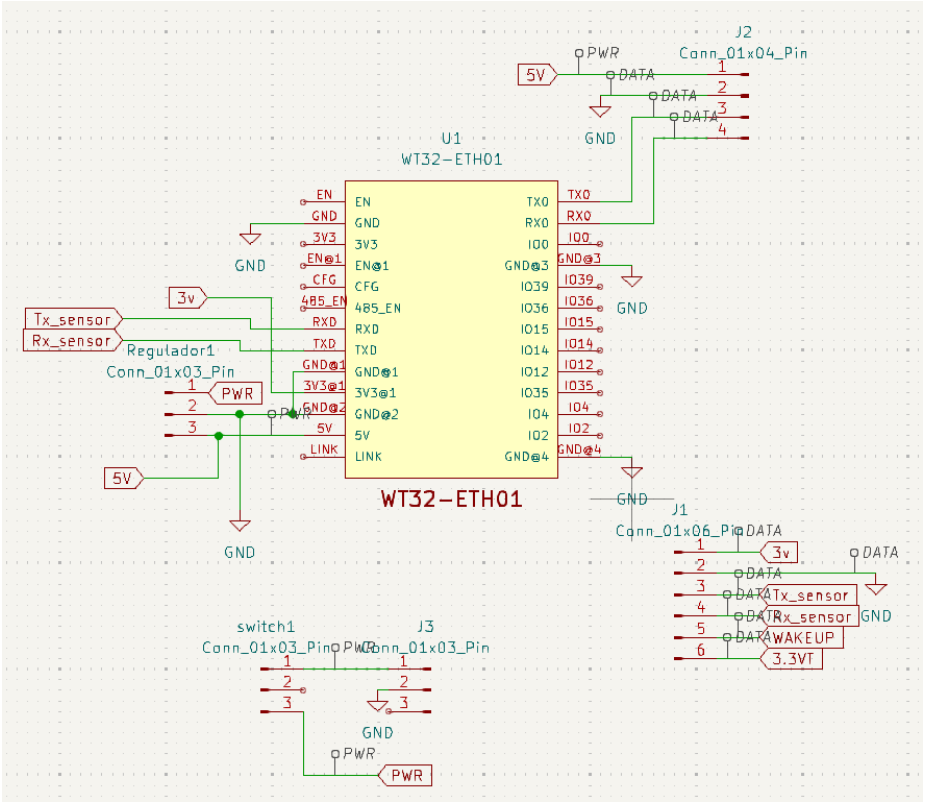

Figura 3. Circuito Esquemático del Módulo de Acceso

Seguido a esto de manera eficiente se hace el diseño de conexión de vías físicas de la PCB, teniendo siempre como objetivo economizar espacio para que la PCB ocupe el menor espacio posible.:

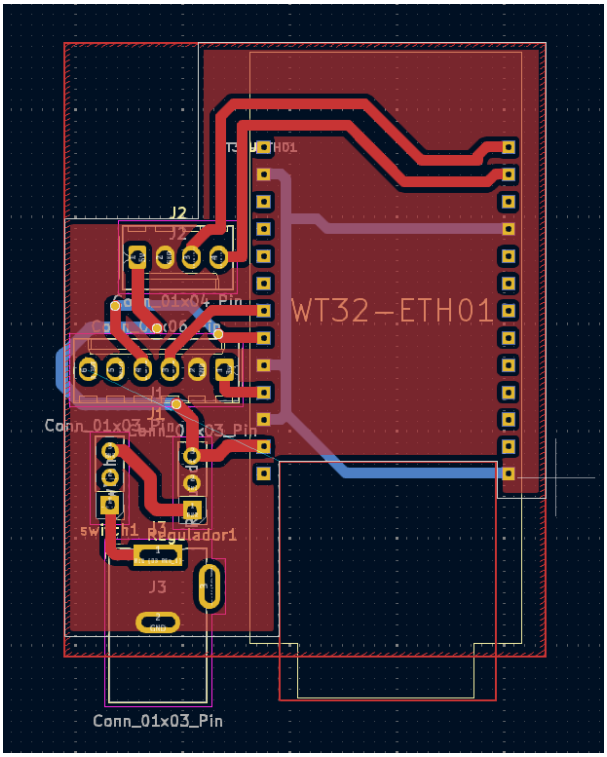

Figura 4. Diseño de vías físicas de la PCB de módulo de acceso.

5.4 Registro y autenticación de huellas dactilares usando ML

Para mejorar la seguridad del sistema de acceso y optimizar la clasificación de huellas dactilares, se implementó un modelo de machine learning para la autenticación.

Selección del modelo

Se optó por utilizar el algoritmo Random Forest Classifier de la biblioteca scikit-learn, ya que este modelo tiene la capacidad de manejar datos de alta dimensionalidad y es robusto ante el ruido y la variabilidad en los datos, características comunes en la captura de huellas dactilares.

Entrenamiento del modelo

El modelo fue entrenado utilizando una serie de templates de huellas dactilares. Para hacer esto posible, los templates en formato hexadecimal fueron convertidos a vectores numéricos, normalizando los datos para garantizar su compatibilidad con el algoritmo de clasificación.

Clasificación y autenticación

En el proceso de clasificación, se procesa la huella que será almacenada posteriormente, y el modelo la clasifica según las carácterisiticas claves del vector, con esta clasificación, la autenticación de las huellas será más eficiente debido a que se comparan las nuevas huellas capturadas contra las huellas almacenadas y clasificadas en la base de datos.

El modelo asigna una probabilidad a cada comparación, y si el umbral de coincidencia es superado, el usuario es autenticado. Ajustar este umbral ha sido un desafío importante, ya que un valor muy bajo podría permitir accesos incorrectos, mientras que un valor muy alto podría dificultar el ingreso de usuarios legítimos.

Aunque el modelo ha demostrado ser eficiente en la clasificación, es necesario ampliar el conjunto de datos de entrenamiento con más huellas para mejorar su precisión y reducir posibles errores. Además, el sistema ha sido diseñado para seguir optimizándose conforme se avancen las etapas del proyecto.

6. TROUBLESHOOTING

Problema: **Dificultades de conexión con la red de la universidad**

Descripción: La red de la universidad bloquea el tráfico ethernet para las direcciones MAC de dispositivos desconocidos o ajenos a la universidad, lo que impide realizar pruebas de conexión mediante ethernet desde el equipo de trabajo de los desarrolladores.

Solución: A largo plazo es imprescindible solicitar acceso a la red para realizar pruebas de conectividad y hacer el despliegue total del sistema. Debido al alcance del proyecto, por el momento se realizan pruebas en una red externa para evitar bloqueos, o se podría plantear el uso de los equipos disponibles en el laboratorio de electrónica.

Problema: Mala configuración de la conexión Ethernet

Descripción: Se tenía definido en el código de conexión al servidor que para inicializar la conexión Ethernet se debía simplemente inicializar el puerto Ethernet de la siguiente forma:

Eth.begin()

Esto se asumió por el hecho de que el microcontrolador WT32-ETH01 ya poseía icorporado el puerto ethernet. Sin embargo, la conexión a la red no fue exitoso.

Solución: Para solucionar este problema se debe inicializar el puerto Ethernet con sus respectivos pines de configuración de la siguiente forma:

```
#define ETH_ADDR        1
#define ETH_POWER_PIN   16
#define ETH_POWER_PIN_ALTERNATIVE 16
#define ETH_MDC_PIN     23
#define ETH_MDIO_PIN    18
#define ETH_TYPE        ETH_PHY_LAN8720
#define ETH_CLK_MODE     ETH_CLOCK_GPIO0_IN
void setup()
{
  // Comunicación con la PC
  Serial.begin(9600); // Serial0 para la comunicación por USB con el PC
  while (!Serial);
  delay(100);

  ETH.begin(ETH_TYPE, ETH_ADDR, ETH_MDC_PIN, ETH_MDIO_PIN, ETH_POWER_PIN,
ETH_CLK_MODE);
```

7. CONCLUSIONES Y PRÓXIMOS PASOS

El segundo prototipo ha logrado avances significativos en la configuración del lector de huellas dactilares y su integración con el microcontrolador y la base de datos. Se ha desarrollado un sistema funcional que permite la captura de huellas dactilares y su clasificación mediante un modelo de machine learning. Sin embargo, se han identificado áreas clave que requieren optimización, como la calidad del sensor de huellas y la integración completa con la base de datos y el servidor.

Para esta etapa fue importante tener acceso a los dispositivos reales (sensor y microcontrolador) ya que no contar con estos dispositivos generó un retraso en el desarrollo del proyecto.

Próximos pasos

- Optimización del modelo de ML: Continuar con el ajuste del umbral de probabilidad y mejorar el formato de los templates de huellas para una autenticación más precisa.

- Pruebas en la red de la universidad: Resolver los problemas de conexión con la red de la universidad y realizar pruebas exhaustivas en el entorno real.
- Desarrollo del tercer prototipo: Implementar la integración completa del sistema en el siguiente prototipo, mejorando la capacidad de visualización de accesos y permitiendo el registro en tiempo real de usuarios y autenticaciones.
- Mejora del diseño hardware: Realizar una impresión 3D que permita el mejor manejo al dispositivo de acceso y una presentación minimalista que sea atractiva como producto final.

8. BIBLIOGRAFIA

[1] “GitHub - crisbol123/Control-de-acceso”. GitHub. [En línea]. Disponible: https://github.com/crisbol123/Control-de-acceso

[2] MongoDB, Inc., "MongoDB Python Drivers Documentation," MongoDB. [Online]. Available: https://www.mongodb.com/docs/drivers/python-drivers/. [Accessed: Sep. 25, 2024].

[3] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

[4] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830.

[5] “Popular python frameworks for building apis: A comprehensive guide”. Oscar Aguado Web. [En línea]. Disponible: https://oscaraguadoweb.com/python/popular-python-frameworks-for-building-apis-a-comprehensive-guide/

[6] Adafruit Industries. (n.d.). Adafruit Fingerprint Sensor Library. Retrieved from https://github.com/adafruit/Adafruit-Fingerprint-Sensor-Library

[7] Quinlan, J. R. (1986). Induction of Decision Trees. Machine Learning, 1(1), 81-106.

[8] “GitHub - khoih-prog/webserver_wt32_eth01: Simple ethernet webserver, HTTP/HTTPS client wrapper library for WT32_ETH01 boards using LAN8720 ethernet. the webserver supports HTTP(S) GET and POST requests, provides argument parsing, handles one client at a time. it provides HTTP(S), MQTT(S) client and supports webserver serving from littlefs/spiffs. now supporting ESP32 core v2.0.0+”. GitHub. [En línea]. Disponible: https://github.com/khoih-prog/WebServer_WT32_ETH01

[9] Angular, "Getting started with Angular: Your first app," 2023. [Online]. Available: https://angular.io/start. [Accessed: 20-Sep-2024].