SISTEMA DE AUTOMATIZACIÓN PARA INGRESO BIOMÉTRICO A AMBIENTES DE FORMACIÓN Y MONITOREO DE OBSERVACIONES RESPECTO A LOS ACTIVOS

PROPUESTA PRELIMINAR

INTEGRANTES:

OMAR JORDAN JORDAN
GERALDINE TRIBALDO SALGADO
ASHLEY CATALINA USAMA
ERIK SANTIAGO CASTELLANOS
DARLING ANGULO

SEMILLERO DE INVESTIGACIÓN

ANÁLISIS Y DESARROLLO DE SOFTWARE

SUB-GRUPO 3: DISEÑO

SERVICIO NACIONAL DE APRENDIZAJE
CALI - COLOMBIA
2025

Contenido

A. Introducción

Antes de exponer la temática de investigación y desarrollo, es de anotarse que el presente documento supone un ejercicio preliminar de análisis y diseño, en el cuál se aborda una posible solución obtenida de la búsqueda y observación, empleando técnicas de ingeniería inversa. Por tanto, este documento es de rigurosidad y detalle básicos, pensado en dar apoyo a las labores generales del semillero de investigación.

La problemática a resolver se trata del control de acceso a ambientes de formación, ya que actualmente el SENA en la sede Salomia de Cali, tiene como logística el uso de llaves, manejadas desde el personal de aseo hasta los cargos administrativos, pasando por los instructores que son quienes más las usan y se ven afectados por la mala gestión. La solución propuesta aborda otras sub problemáticas como son, el registro eficiente de observaciones que se realizan cuando los activos se ven afectados negativamente, también el control de acceso puede medir la eficiencia con la que los docentes cumplen sus labores.

El sistema propuesto en el documento se apoya mayormente en UML, aunque usando una versión minimalista y ágil, verá requerimientos funcionales y no funcionales, historias de usuario, de los que surgen diseños de casos de uso, bases de datos y otros modelos.

Este diseño realizado con bajo detalle y aún con mucho camino investigativo por delante, demuestra un ejercicio de aplicación de diseño de software, suponiendo que a priori no hay otros sistemas a los que acoplarse. En la práctica existe una base de datos institucional, un sistema de asignación de horarios y en general muchas tecnologías hardware + software para acceso a puertas, todo listo para ser instalado y acoplado. Dichas tecnologías y sistemas existentes, deben investigarse a profundidad para mejorar la solución planteada.

B. Problema

En la institución SENA, más específicamente en la sede Salomia de Cali, se han identificado problemáticas en: la logística de acceso a ambientes de aprendizaje o trabajo, seguimiento del cumplimiento de labores de docencia, seguimiento de asistencia a clase para aprendices, necesidad de automatización en la obtención de registros de cambios en el estado de los activos, etc. Muchos de estos temas son debidos a la realización manual de los procesos.

El grupo semillero ADSO del CEAI encuentra una oportunidad para aplicar las tecnologías 4.0, renovando y valorizando el quehacer y el sentido de pertenencia al interior de la institución. De las problemáticas, el presente documento aborda dos, haciendo mayor énfasis en el control de acceso a los ambientes de aprendizaje.

La pregunta problema directriz es:

> *¿Cómo sería el sistema software / hardware idóneo para optimizar el acceso a los ambientes académicos del SENA, el cuál brinde a los instructores y otros funcionarios de la institución, confort y facilidad de uso con respecto al sistema actual?*

Otra pregunta problema para ampliar la solución con una funcionalidad secundaria es la siguiente:

> *¿Cómo se podría automatizar el registro de novedades observadas por los instructores, concernientes al estado de los activos al interior de los ambientes de aprendizaje, para poder dar una trazabilidad a las pérdidas y daños?*

Como análisis, hay que destacar una diferencia clara en las necesidades de un salón de clases versus un entorno con oficinas de trabajo individuales. En el segundo caso, cada persona con un puesto de trabajo tiene permiso de acceso a su salón o zona de trabajo, mientras en una clase, el instructor o profesor a cargo tiene acceso, pero muchas otras personas entrarán y saldrán durante la clase, no se puede poner al instructor en rol de portero o a otros a que abran la puerta cuando alguien “timbre”.

Se debe tener en cuenta que, registrar biométricamente a los aprendices no parece ser una buena idea, puesto que requiere una mayor logística al ingreso y a la salida de su etapa lectiva, sería más tedioso para las labores administrativas del sistema, podría entrar en conflicto con el temor de algunas personas a dar sus datos, y al no tener los aprendices un rol de permiso de acceso sin compañía de instructores, sería un registro innecesario, a menos que el ingreso del instructor inicie una especie de sesión en el salón, pero analizando, eso complica más el sistema sin generar necesariamente un beneficio equiparable.

Como propuesta, se idea un sistema interno, puede ser a modo de switch para dejar la puerta en modo permisivo, puesto que ya hay un instructor dentro y una clase en curso.

C. Marco Teórico

Sobre los sensores biométricos

En el artículo de Scientia et Technica [1] se hace un listado de técnicas biométricas con descripciones superficiales, en general biometría refiere a medir alguna de las características del cuerpo humano, sean físicas o comportamentales con el fin de identificar a los individuos. Algunos tipos de biometría son: la cara, la huella, la geometría de la mano, el iris, la voz, las venas, las orejas, el pulso cardiaco, la radiografía dental, el ADN, la forma de escribir a mano.

Para que un sistema biométrico sea aceptable, se requiere: **universalidad** (características presentes en todos los individuos), **unicidad** (muy pequeña probabilidad de dos personas con la misma información), **permanencia** (que la característica no cambie con el tiempo), **cuantificación** (debe poder medirse cuantitativamente).

Las técnicas de reconocimiento más usadas hasta la fecha del artículo son: El **reconocimiento de firmas**, que usa las características a la hora de firmar, aunque es poco fiable puesto que cada firma del mismo individuo suele variar mucho. El **reconocimiento facial**, mide puntos clave de zonas de la cara, depende en gran medida de la calidad del algoritmo más la claridad de la fotografía. El **reconocimiento de iris**, es uno de los más fiables dado que hay muchas características que extraer a comparación de otros métodos, los ojos son únicos y estables en el tiempo, pero el contra radica en el costo, ya que requiere cámaras y algoritmos muy especializados. El **reconocimiento de voz** es un buen método, porque extrae características únicas, aunque suele verse afectado por el ruido del medio y en algunos casos es imitable.

El **reconocimiento por huella dactilar**, según Narayan, L & G, Sonu & M, Soukhya [2] es un método que cumple con todos los requisitos de la biometría (listados arriba), tiene una buena relación calidad / precio, y es socio culturalmente aceptado. Se utilizan cámaras o superficies capacitivas para captar la huella (260x300 px), el proceso de adquisición es seguido por un pre-procesamiento que deja la imágen en la calidad ideal, a blanco y negro con las crestas y valles bien definidos. Continúa la aplicación de algoritmos que extraen los patrones, entre ellos: bifurcaciones, fin de crestas, ángulo de las líneas. Estos datos son los que, o bien se guardan en la base de datos (proceso de registro) o se comparan con un dato guardado (proceso de comparación), arrojando un resultado.

Existen dos conceptos importantes, el FAR (false aceptation rate) que porcentualmente mide qué tan probable es que se aceptada una comparación de huellas diferentes, y el FRR (false rejection rate) que ve la probabilidad de rechazar a la huella que sí debería aceptar. En el caso del sensor de ejemplo JM-101 [3] tenemos FAR < 0.001% y FRR < 1% dicho sensor es un sistema AFIS (automatic fingerprint identification system), significa que internamente ya trae todos los algoritmos que hacen la extracción de características, cuenta con memoria para 150 huellas, tiempo de identificación < 1s y búsqueda < 0.5s, con comunicación serial UART y USB. Este sensor es una opción para prototipos de aprendizaje o desarrollos caseros DIY (do it yourself).

Cuando usamos un login clásico correo + contraseña, estamos haciendo una comparativa 1:1, pues ya tenemos al usuario sólo es comparar sus credenciales de acceso, por otra parte, cuando tenemos sólo una credencial de acceso, digamos una huella, la comparativa es 1:N puesto que toca buscar la coincidencia en toda la base de datos. Esto es algo a tener en cuenta a la hora de diseñar, ya que se están comparando estructuras de datos complejas.

Cerraduras electromagnéticas para puertas

pendiente…

Controlador para una o varias puertas

pendiente…

Sensor de apertura de puerta

pendiente…

uninterruptible power supply (UPS)

pendiente…

D. Diagrama Conceptual

Hay tres espacios a distinguir: el data center que contiene en este caso al servidor, aunque podría estar a parte.

El salón de logística donde trabaja algún administrador, allí además del software (sea desktop o web), está un sensor biométrico para llevar a cabo la adquisición de registros, aunque en otra arquitectura esa adquisición se podría hacer con las mismas puertas).

Luego tenemos los salones de clase o en general, espacios que deben ser protegidos, allí están las puertas con sus sensores, controlador y actuadores, además, como se ve en el diagrama, existe un elemento de interacción extra que sería el smartphone del instructor o personal a cargo.

El smartphone se utiliza para llevar a cabo registros u observaciones, pero no interfiere con la apertura de las puertas, las mismas sólo pueden accederse con el sensor biométrico, sea huellero, facial, voz, iris, más opciones no biométricas: tarjeta, contraseña, Qr. Por otra parte, la salida se lleva a cabo sólo con un pulsador, y se propone un switch interno que permita el estado totalmente permisivo de la puerta (explicado en casos de uso), finalmente, la puerta cuenta con su indicador que alertará cuando está quede abierta o mostrará su estado actual.

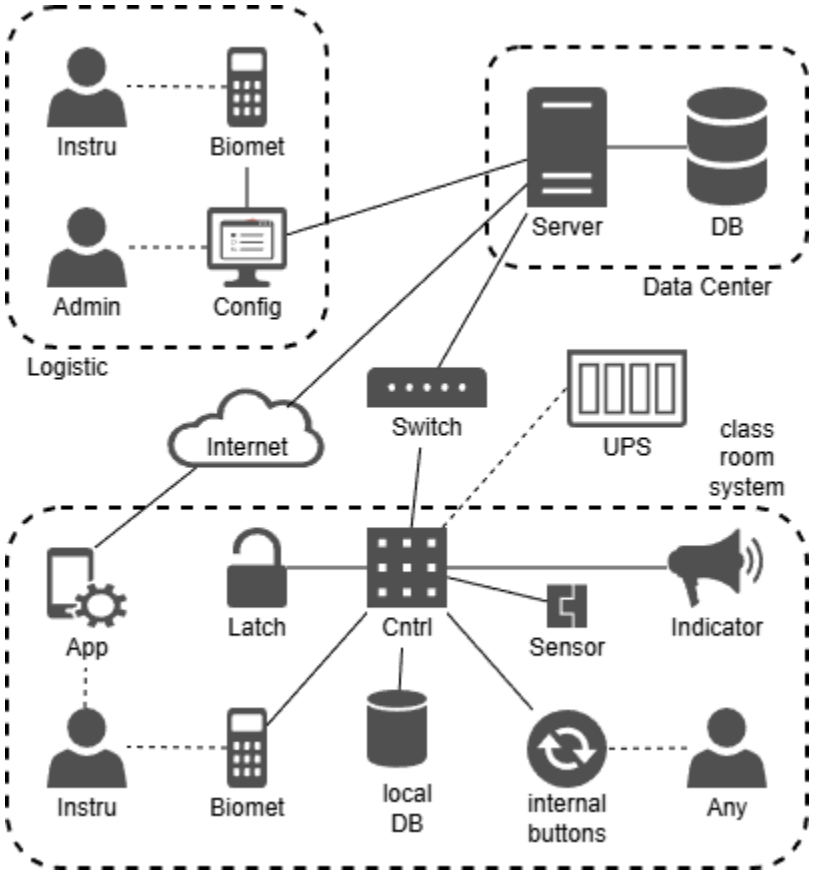

Figura D1

La conexión del sistema se divide en dos subsistemas, uno es la red interna institucional, la cuál no requiere un servicio de Internet o hosting, y con ello promete ser más barata y a la vez estable ante pérdidas del servicio de Internet a nivel de la institución.

El segundo subsistema es dependiente de Internet, sea institucional o por parte de los datos de cada usuario, este hace funcionar el sistema de registro de novedades, como se había explicado es un sistema paralelo a la seguridad de las puertas, funcionando entonces como cualquier otro servicio web.

Los sistemas de las puertas deben cumplir con el requerimiento no funcional de mantener su funcionamiento aún cuando no haya energía por unas horas y aún cuando el servidor no de respuesta. Esto se logra con bases de datos locales, a nivel de controlador, las cuáles se sincronizan con la base de datos principal, sosteniendo el estado de permisos y firmas biométricas ante la desconexión.

En los diagramas se ve que los controladores y por extensión todo el sistema de puerta, requiere conexión a una UPS (sistema de alimentación de emergencia con baterías), que va en paralelo al sistema de conexión de datos, por medio de switches. En principio es tentativo pensar en la conexión tipo PoE (power over ethernet), la cual entrega datos y energía en un mismo cable, pero esta puede verse limitada por la potencia entregada a los actuadores de las puertas, y otros dispositivos que se conecten a futuro (tentativa de acoplar un sistema de cámaras). Se requiere hacer investigación para indagar sobre la potencia requerida y los costos de mantenimiento de las UPSs.

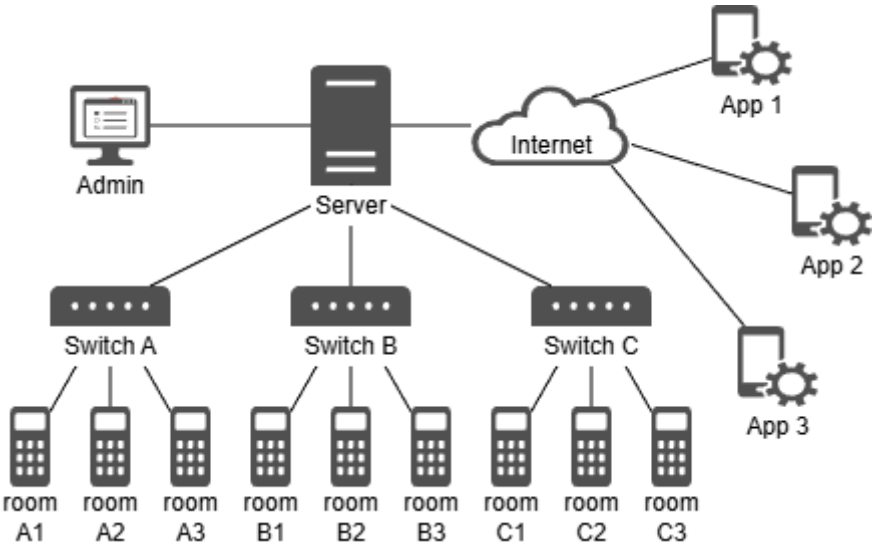

Figura D2

En el diagrama de ejemplo, vemos varios puntos de la red interna conectando cada uno a varios salones, imaginemos que cada punto podría ser un edificio dentro de la institución, con sus respectivos salones asociados. En caso de haber varias sedes, el sistema sólo debe añadir un tunelamiento de su red interna, probablemente las sedes ya utilicen WAN o MAN.

E. Usuarios Identificados

La tabla muestra una caracterización superficial de los usuarios del sistema, al interior de la institución SENA, estos usuarios pueden tener subdivisiones jerárquicas para cada tipo, pero se hace un análisis macro del ecosistema humano.

Tabla E1

Usuario	Perfil	Frustración	Deseo
Administrador	Persona responsable, usualmente de edad mayor a 30 años, con conocimiento en manejo de software de escritorio y gestión informática	Hacer un seguimiento a la formación de los aprendices y a la eficiencia de los instructores, es un proceso difícil muchas veces lleno de sesgos, falta de información o errores humanos como la complicidad y autosabotaje	Llevar un registro exacto y fiable del sistema educativo, poder ver cómo es el desempeño de los instructores y aprendices Poder hacer más fácil las largas tareas administrativas
Instructor	Trabajador con alta responsabilidad en la administración de su espacio de clases, usualmente adultos mayores a 30 años, sin conocimientos de tecnología más allá del celular, para el caso de asignaturas no afines con la tecnología, por ej, educación física, costura y demás, este grupo suele subdividirse en instructores de planta y contratistas	El uso de llaves para ingresar a los espacios es usualmente tedioso, puede ser lento, confuso, en unas ocasiones un instructor se las lleva a casa, o las extravía, o llega más tarde a clase respecto a otro que carece de llaves. En cuanto a la creación de reportes de novedades, no hay una forma eficiente de hacerlo, suelen duplicarse o no llevarse a cabo al final de cuentas Un gran problema es la impuntualidad de los aprendices, tanto al inicio de las formaciones como luego de los descansos, también muchos se van antes de	Poder ingresar fácilmente a los ambientes de formación, registrándo su llegada para dar validez a su responsabilidad Poder registrar eficiente y rápidamente las novedades o daños vistos en los ambientes, pues con esto se podría actuar más rápido y rastrear responsables, así cumplir el deseo de tener ambientes en mejor estado Ver que los aprendices llegan puntuales y pasan el tiempo de formación dentro del ambiente, sin escaparse, sin

		tiempo, es un problema con el que no se puede lidiar con el sistema actual de chequeo manual único	que el instructor se desgaste llamando a lista muchas veces o haciendo presión que crea conflictos sociales
Aprendiz	Mayormente jóvenes en la jornada diurna (en torno a 19 años), y adultos en la nocturna (en torno a 27 años), muchos no familiarizados con las tecnologías más allá del celular, tendencia en los jóvenes a comportamientos que ponen en riesgo los activos de la institución, por ej, a causa del juego físico	Tener que esperar a que los instructores encuentren las llaves o esperen a que otro con las llaves aparezca También se da el caso de instructores que llegan tarde, no cumplen el horario o simplemente no llegan enviando excusas a última hora, la institución no tiene buenas métricas para actuar en muchos de estos casos Muchas veces los ambientes tienen daños en sus mobiliarios o activos en general, lo que baja la moral a la hora de ver una clase	Tener una educación de calidad, con instructores puntuales, salones altamente disponibles y activos en buen estado Aunque no sea un deseo evidente, el hecho de los aprendices ser forzados por el sistema a ser puntuales, hace que reciban una mejor formación para el trabajo, lo cuál es algo que buscan, pese a que en la superficie sea normal no querer madrugar por deseo propio
Aseador	Trabajadores que limpian los espacios, tienen por contrato responsabilidad respecto al cuidado de los activos, poco involucrados con tecnología más allá del celular	Igualmente deben estar pendientes de las llaves, al entrar a tantos espacios diferentes, es toda una labor de memoria y organización personal, manejar tantas llaves	Poder ingresar a los ambientes de forma fácil, concentrando esfuerzos en la logística del material de aseo y no en moverse por todo el campus, llevando llaves de un lugar a otro
Funcionario	Trabajador de oficina, con conocimiento en manejo de software de escritorio, este usuario incluye profesiones como logística, contaduría, psicología, trabajo social, y afines	Las oficinas donde conviven varios funcionarios, sufren el mismo tema de las llaves, aunque en menor medida a los ambientes de formación, sería mucho más cómodo un	Sentir que está en una institución que avanza y se compromete con las nuevas tecnologías, al tener espacios de trabajo con entradas automatizadas

		acceso fácil pero seguro	
Vigilante	Trabajadores con entrenamiento en gestión de espacios y actividad humana, poco involucrados con tecnología más allá del celular	Usualmente no se involucran con los espacios internos de formación, podría presentarse algún caso en modalidad de emergencia	Respecto a los ambientes de formación, no hay muchos deseos, los vigilantes tienen sus otros deseos, especialmente en el trabajo en portería

Comportamientos emergentes de los usuarios respecto a diferentes sistemas de acceso:

- Usar algún dispositivo de acceso, como lo son tarjetas o Qr, de forma irresponsable, prestandola a aprendices o a otros funcionarios, dañando así la trazabilidad del historial.
- Si el acceso registrase aprendices, para medir su asistencia, estos podrían marcar entrada y salida a la vez que no entran al ambiente, capando clase.
- Los aprendices más indisciplinados podrían jugar con los sensores biométricos, acortando su vida útil o hasta dañándolos a propósito, por ese hecho, algunas instituciones usan cámaras de vigilancia en las entradas, para trazabilizar daños.

F. Requerimientos Funcionales

Los requerimientos funcionales (RF) describen las funcionalidades que debe tener el sistema, para cumplir con la solución a la problemática y dar a los usuarios un completo flujo de interacción.

Requerimientos para la gestión del sistema

1. El sistema debe permitir la creación y actualización de nuevos usuarios, asignándoles sus respectivos datos, entre los que se incluye su rol, centro, identificación y contacto.
2. El sistema debe permitir la gestión de métricas o credenciales de acceso para cada usuario, ya sean biométricas, contraseñas, tarjetas, etc.
3. El sistema debe permitir que se asigne un salón a un usuario específico, o también la revocación de dicha asignación, especificando su fecha inicial, final y horario.
4. El sistema debe permitir la gestión de salones, pudiendo añadir nuevos y especificar en qué zona y sede se ubican físicamente.
5. El sistema debe permitir la gestión de zonas, pudiendo añadir nuevas, estas zonas representarán agrupamientos físicos de salones.
6. El sistema debe permitir la gestión de sedes, pudiendo añadir nuevas, este proceso debe ser cuidadoso en cuanto a seguridad, para prevenir cambios drásticos por error.
7. El sistema debe permitir la gestión de centros, pudiendo añadir nuevos, este proceso debe ser cuidadoso en cuanto a seguridad, para prevenir cambios drásticos por error.
8. El sistema debe permitir la gestión de grupos de usuarios, pudiendo añadir nuevos.
9. El sistema debe permitir que se asigne o revoque uno o varios grupos a cada usuario.
10. El sistema debe permitir que se conecten grupos de usuarios a zonas de salones.
11. El sistema debe permitir hacer búsquedas filtradas de entidades, por ejemplo, usuarios, salones, sedes, grupos, centros, zonas.
12. El sistema debe permitir el login de usuarios, siendo cada tipo de usuario permitido sólo en la aplicación correspondiente a su rol, por ejemplo, los administradores son los únicos que pueden entrar a la aplicación administrativa.
13. El sistema debe permitir a los usuarios cerrar sesiones abiertas.
14. El sistema debe permitir que se den o revoquen permisos de administración, mediante un sistema de verificación presencial, puede ser basado en biometría.

Requerimientos para recopilación de información general.

15. El sistema debe permitir ver un historial de auditorías administrativas usando filtros.
16. El sistema debe permitir ver un historial de interacciones con las puertas de los salones, pudiendo filtrar los resultados, debe mostrar quién hizo dichas interacciones.
17. El sistema debe permitir ver un historial de cumplimiento de asignaciones, donde se muestre cómo se han cumplido las entradas de un usuario asignado a un salón en un horario específico.
18. El sistema debe permitir ver las asociaciones entre usuarios y salones, a modo de horarios y espacios asignados.

19. El sistema debe permitir ver las observaciones hechas por los usuarios, filtrando por salones o quién hizo la observación, también puede ser por fechas.

Requerimientos para la interacción directa con el salón y su puerta con ingreso automatizado

20. El sistema debe permitir el ingreso a los salones a través de las puertas aseguradas electromecánicamente, mediante un sistema de verificación de usuario y sus permisos.
21. El sistema debe permitir observar el estado de una puerta, mostrando indicadores visuales en caso de que esté bloqueada, abierta o con bloqueo parcial.
22. El sistema debe permitir un modo de bloqueo parcial (modo permisivo), donde la puerta pueda ser accedida por cualquier persona dado que alguien ya la ha desbloqueado y ha activado dicho modo.
23. El sistema debe permitir que cualquier usuario pueda salir de un salón bloqueado, sin necesidad de estar registrado o con permisos.

Requerimientos referentes al subsistema de realización de observaciones

24. El sistema debe permitir la realización de observaciones a modo de reporte, que incluya un texto y opcionalmente una imágen fotográfica.
25. El sistema debe permitir al usuario que hace observaciones, cambiar sus credenciales de acceso a la aplicación para dicho fin.

G. Requerimientos No Funcionales

Los RNF son las condiciones que debe cumplir el sistema a la hora de llevar a cabo los requerimientos, es el cómo debe ejecutarlos, por ej, velocidad, límites, seguridad, librerías, hardware.

1. El sistema debe ser capaz de funcionar autónomamente cuando se corta el fluido eléctrico por al menos 24 horas, suponiendo un uso no intensivo de las puertas.
2. El control de las puertas debe mantener información de acceso local cuando no tenga conexión al servidor con la database, al menos 200 registros de permisos.
3. El control de las puertas debe almacenar el historial de acciones cuando no hay conexión, para enviarlas posteriormente cuando se restablezca, al menos 1000 registros de sucesos.
4. El sistema debe ser independiente de la red de Internet, del proveedor ISP, al menos en cuanto a su funcionalidad de uso de puertas, para ser económico y activo cuando la red de Internet caiga.
5. El sistema debe exigir que las contraseñas tengan cierto grado de complejidad (mínimo 6 caracteres, con mayúsculas, minúsculas y números), además deben guardarse encriptadas.
6. El sistema debe auditar los cambios hechos por los usuarios administradores, pudiendo mostrar un historial detallado.
7. El sistema debe ser robusto ante ataques DDoS, inyección SQL, XSS y fuerza bruta.
8. El sistema debe garantizar una disponibilidad del 99% del tiempo mensual.
9. El sistema debe cargar datos bajo demanda en menos de 4 s.
10. El sistema que muestra los datos importantes, debe soportar al menos 10000 registros descargados y contenidos en memoria RAM y UI.
11. El sistema administrativo no debe tardar más de 4 s en hacer cambios a la DB.
12. El sistema debe soportar al menos 700 conexiones simultáneas para envío de observaciones.
13. El sistema debe guardar las imágenes de observaciones en formato JPG de máximo 2000 x 2000 pixeles.
14. El sistema debe tener unos indicadores lumínicos y quizá sonoros en las puertas, para mostrar el estado de las mismas, estos indicadores lumínicos deben ser perfectamente visibles en el día, aún con luz indirecta del sol, al menos a 10 m.
15. El sistema deberá soportar hasta 30 switches PoE con 25 puertas asociadas a cada uno, para un total de 750 puertas.
16. El sistema debe contar con baterías para UPS con tiempo de vida de al menos 4 años.
17. El sistema debe soportar al menos 10000 usuarios en la base de datos, aunque deberían ser más si a futuro los aprendices van a ser registrados.
18. El sistema debe ser escalable en tamaño y soporte, o al menos instalable en diferentes sedes o instituciones, como unidades independientes.
19. El sistema debe evitar que los usuarios queden atrapados por error dentro de una sala.
20. El tiempo para abrir la puerta tras verificar las credenciales, debe ser menor a 5s.

H. Historias de Usuario

Las historias de usuario (HU) se escriben de la forma "Como usuario, quiero hacer tal cosa, para tal objetivo de valor", esto para describir de una manera sencilla una pieza del software, focalizándose en la usabilidad u objetivo que le da valor al producto.

Historias del usuario administrativo.

1. **Como administrador**, quiero poder registrar, actualizar y eliminar usuarios en el sistema, especificando su rol, centro, datos de identificación y contacto, para dar cumplimiento posterior a las labores de asignación de permisos.
2. **Como administrador**, quiero poder gestionar las métricas o credenciales de acceso de un usuario, por ejemplo, registrar o actualizar su huella, alguna contraseña, etc, para permitirle usar los permisos de acceso.
3. **Como administrador**, quiero poder asignar y revocar un salón a un usuario, especificando una fecha de inicio, fin y un horario, para darle un control de acceso específico a áreas puntuales, usualmente asociadas a clases.
4. **Como administrador**, quiero poder registrar, actualizar y eliminar salones, especificando en qué zona o agrupamiento de salones se encuentra, y en qué sede se encuentran, para actualizar el sistema cada vez que se conecte una nueva puerta inteligente.
5. **Como administrador**, quiero poder registrar, actualizar y eliminar zonas o agrupamientos de salones, para tener un control estructural del sistema.
6. **Como administrador**, quiero poder registrar, actualizar y eliminar sedes, aunque lo último debería ser difícil de conseguir evitando catástrofes, para tener un control estructural del sistema.
7. **Como administrador**, quiero poder registrar, actualizar y eliminar centros, aunque lo último debería ser difícil de conseguir evitando catástrofes, para tener un control estructural del sistema.
8. **Como administrador**, quiero poder registrar, actualizar y eliminar grupos, es decir, clústers de usuarios, para manejar permisos de una forma óptima sin tener que darlos uno por uno.
9. **Como administrador**, quiero poder asociar o revocar grupos a usuarios, para de este modo darles permisos óptimamente, ya que se asociarán a los grupos.
10. **Como administrador**, quiero poder conectar grupos de usuarios con zonas de salones, para dar permisos masivamente y con eficiencia.
11. **Como administrador**, quiero poder hacer búsquedas de usuarios, salones y demás, observando los resultados en lista y pudiendo aplicar filtros, para encontrar el elemento sobre el cuál deseo llevar a cabo labores administrativas.
12. **Como administrador**, quiero poder loguearme en el sistema administrativo, utilizando las credenciales de acceso de igual forma que si ingresara a una puerta, para poder hacer mis labores y garantizar que el sistema es seguro.
13. **Como administrador**, quiero poder cerrar sesión en el sistema administrativo, sea manualmente o por tiempo de inactividad, para que nadie vaya a manipularlo.

14. **Como administrador**, quiero poder dar o revocar permisos de administrador (rol administrativo) a alguien que deba verificar sus credenciales en mi despacho, del mismo modo que se hace el proceso de registro de usuarios o mi login, para evitar por accidente, dar o quitar permisos a alguien mientras se hace un cambio de rol.

Historias de usuarios que recopilan o leen información del sistema

15. **Como funcionario**, quiero poder ver un historial de auditorías, donde pueda ver las acciones de los administradores, para así poder rastrear comportamientos indebidos, alertar sobre posibles fallos en la seguridad o revisar el buen cumplimiento de las labores administrativas.
16. **Como funcionario**, quiero poder ver un historial de las interacciones con las puertas, así como su estado actual, incluyendo quién hizo los accesos y tener los correspondientes filtros de búsqueda, para estar al tanto de la seguridad de la institución y rastrear fácilmente comportamientos sospechosos.
17. **Como funcionario**, quiero poder ver un historial de cumplimiento de asignaciones, es decir, de cumplimiento de clases o jornadas, con el correspondiente filtro de búsqueda, para así poder identificar fallos en la responsabilidad docente o el mal uso del campus.
18. **Como funcionario**, quiero poder ver los horarios a los cuáles está asociado un instructor con un salón, buscando por filtro, para así agilizar labores de identificación de espacios libres o verificar que las asignaciones sean adecuadas.
19. **Como funcionario**, quiero poder ver las observaciones hechas por los instructores, pudiendo hacer búsquedas por filtro, para así llevar a cabo acciones sobre las mismas y hacer revisiones de casos.

Historias de usuarios que interactúan con el sistema puerta / salón

20. **Como administrador**, quiero poder entrar fácilmente a cualquier salón, para hacer vigilancia de los mismos y verificar su funcionamiento
21. **Como funcionario**, quiero poder asegurarme que los salones tanto de clases como de administración, queden debidamente asegurados, para evitar que personal no autorizado ingrese causando potenciales daños a los activos y a la institución.
22. **Como instructor**, quiero poder entrar fácilmente al salón de clase que me fué asignado, en el horario que me fué asignado, para poder desarrollar mi clase sin contratiempos.
23. **Como instructor de planta**, quiero poder ingresar a cualquier salón de mi centro aún cuando no tenga una clase asignada ahí, para poder utilizar los espacios que estén disponibles y hacer labores logísticas.
24. **Como aseador**, quiero poder ingresar fácilmente a los salones que debo limpiar, para poder efectuar las labores de limpieza sin contratiempos.
25. **Como vigilante**, quiero poder ingresar a cualquier salón, para poder brindar apoyo de seguridad y resolver problemas de órden que puedan presentarse.
26. **Como funcionario no instructor**, quiero poder acceder fácilmente a los salones donde desarrollo mis actividades diarias o hago supervisión, para poder agilizar el proceso de ingreso y evitar encontrarme con espacios cerrados por accidente, ej, pérdida de llaves.

27. **Como instructor**, quiero permitir que los aprendices salgan y entren del salón, por ej, para ir al baño, una vez el salón está bajo mi mando, sin tener entonces que interactuar yo con la puerta, para evitar contratiempos interactuando innecesariamente con la misma, en otras palabras para evitar hacer las veces de portero estando en clase.
28. **Como usuario**, quiero poder identificar el estado de una puerta de forma fácil, por ej, con alguna iluminación, para saber si la puerta está bloqueada, abierta o desbloqueada.
29. **Como usuario**, quiero poder salir de un salón protegido con ingreso inteligente, aún cuando la puerta esté cerrada y el sistema de protección funcionando, para evitar quedarme encerrado indefinidamente.

Historias de usuarios que hacen reportes de novedades, o llamemoslas observaciones

30. **Como instructor**, quiero poder hacer reportes de novedades, dejando evidencia textual y fotográfica, por ej, cuando encuentre un equipo dañado a una hora y lugar exactos, para informar sobre el suceso a la institución y así evitar conflictos de culpabilidad hacia quienes encontraron el daño, al tiempo que se permite rastrear a los responsables.
31. **Como instructor**, quiero poder ingresar al sistema de observaciones usando mis credenciales de inicio de sesión, para poder hacer el envío de observaciones.
32. **Como instructor**, quiero poder cerrar sesión en mi aplicación de envío de observaciones, para evitar usos indebidos de mi cuenta.
33. **Como instructor**, quiero poder cambiar mis credenciales de la aplicación de observaciones, con verificación en dos pasos, para poder mantener mi cuenta segura o restablecerla si olvidé la contraseña.

I. Casos de Uso

Primero se muestran dos diagramas de clases, cabe aclarar que tanto los diagramas como tablas son minimalistas, enfocados en cubrir las funcionalidades básicas de un prototipo demostrativo, a continuación vemos la App para hacer observaciones y una puerta de salón.

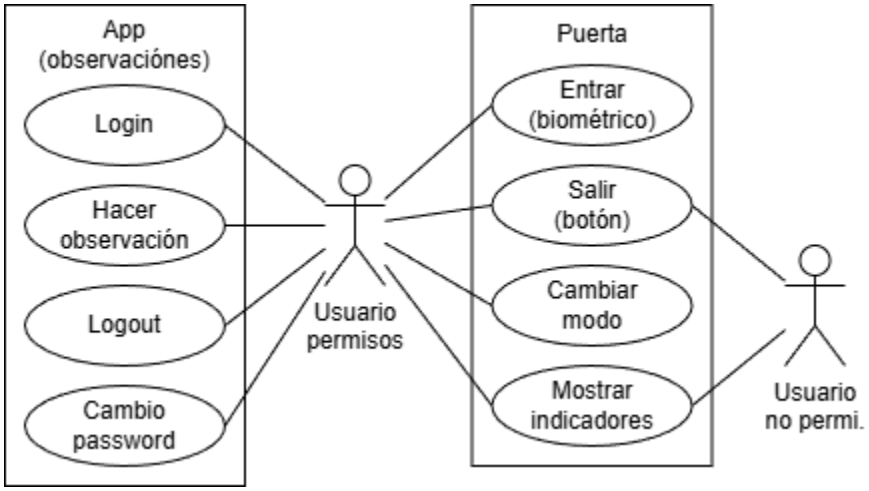

Figura I1

Como se ve en los requerimientos e historias de usuario, la aplicación administrativa, acá apodada configuración del sistema, es la que tiene más ítems por programar, pero en el siguiente diagrama se ha minimizado a los necesarios para explicar cómo un usuario es registrado y asignado a un salón. Esto se verá en detalle en las tablas de casos de uso.

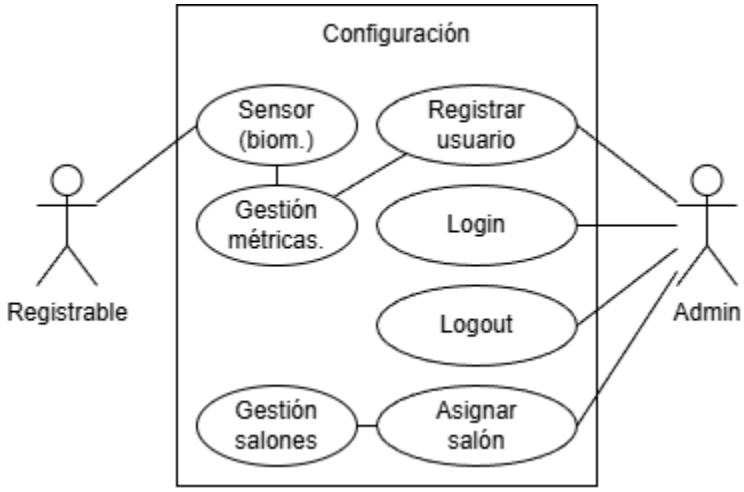

Figura I2

De los requerimientos funcionales, se estructuran secuencias de pasos que demuestren el funcionamiento real del sistema, sin ahondar en el mismo ya que todo se identifica desde la perspectiva de los usuarios o actores. En las siguientes secuencias se ha omitido la

comunicación de la puerta con el servidor para llevar a cabo el historial de cambios de estado, eso queda implícito y se ahondará en una documentación más amplia del proyecto. Cabe añadir que en esa funcionalidad, el sistema de almacenamiento local guarda los cambios que no se han logrado enviar, en caso de no haber conexión, para enviarlos luego.

CU-01	**Acceso permitido**
precondición	el instructor está registrado y con permiso de acceso a el salón que va a entrar, el sistema está energizado y con conexión
secuencia	1. **instructor**: está frente a la puerta que desea abrir y pone su huella 2. **puerta**: lee la huella y envía una solicitud de acceso al servidor 3. **servidor**: verifica si hay acceso y envía respuesta a la puerta 4. **puerta**: da la órden a su actuador de abrir y muestra el indicador 5. **instructor**: ve el indicador y empuja la puerta 6. **instructor**: una vez entra activa el modo permisivo 7. **instructor**: entran todos los aprendices y cierra la puerta 8. **puerta**: se bloquea, muestra su indicador en modo permisivo
excepción	- si la huella no es bien detectada debe reintentarlo - si no hubiese conexión al servidor, el sistema buscaría credenciales de acceso en su memoria local - el modo permisivo también podría activarse si ya la puerta se cerró y quedó bloqueada, pues este está dentro junto con el botón de salir

CU-02	**Salida y entrada de aprendiz a puerta permisiva**
precondición	el salón ha sido abierto previamente por un instructor, poniendo la puerta en modo permisivo, el aprendiz se halla adentro y desea ir al baño, el sistema está energizado y con conexión
secuencia	1. **aprendiz**: va a salir y estando frente a la puerta pulsa el botón interno 2. **puerta**: da la órden a su actuador de abrir y muestra el indicador 3. **aprendiz**: ve el indicador y empuja la puerta 4. **aprendiz**: sale y cierra la puerta 5. **puerta**: se bloquea, muestra su indicador en modo permisivo 6. **aprendiz**: regresa al salón, está frente a la puerta y pone su huella 7. **puerta**: ya que está en modo permisivo se abre, muestra indicador 8. **aprendiz**: ve el indicador y empuja la puerta 9. **aprendiz**: entra y cierra la puerta 10. **puerta**: se bloquea, muestra su indicador en modo permisivo
excepción	- puede que el sistema además de huella tenga un botón externo para estos casos de ingreso permisivo - si deja la puerta abierta mientras va al baño, está tras un tiempo debe emitir una alerta de haber quedado abierta, puede ser sonora, titilante, esto podría ser deshabilitar en caso de que se quiera dejar abierta para airear el salón - también puede ser posible que no se requiera cerrarla en caso de que

	tenga un sistema de cerrado automático mecánico (brazo resortado)

CU-03	**Finalización de clase con bloqueo de puerta**
precondición	el instructor y sus aprendices se encuentran dentro del salón de clase, lo que significa que la puerta está desbloqueada y además está puesta en modo permisivo, el sistema está energizado y con conexión
secuencia	1. **instructor**: va a salir y estando frente a la puerta pulsa el botón interno 2. **puerta**: da la órden a su actuador de abrir y muestra el indicador 3. **instructor**: ve el indicador y empuja la puerta 4. **instructor**: hace salir a todos sus aprendices 5. **instructor**: desactiva el modo permisivo 6. **instructor**: sale y cierra la puerta 7. **puerta**: se bloquea, muestra su indicador en modo bloqueo
excepción	- si un aprendiz trata de entrar de nuevo, la puerta alerta de acceso denegado, puede ser con sonido o titilante - si olvida quitar el modo permisivo, el indicador igual alertará al instructor o cualquier otro trabajador sobre el estado de la puerta

CU-04	**Envío de observaciones**
precondición	el instructor tiene instalada la App en su smartphone y previamente ha realizado su proceso de login, cuenta con conexión a Internet
secuencia	1. **instructor**: observa un daño en uno de los activos del salón 2. **instructor**: saca su smartphone y abre la aplicación de observaciones 3. **instructor**: pulsa la opción de hacer observaciones 4. **app**: muestra la interfaz de hacer observaciones 5. **instructor**: pulsa en foto y saca una foto del activo afectado 6. **app**: muestra la foto tomada 7. **instructor**: escribe un texto acorde a lo observado 8. **instructor**: escribe el ID del activo afectado 9. **instructor**: pulsa en enviar 10. **app**: envía los datos al servidor 11. **servidor**: recibe los datos y devuelve confirmación 12. **app**: muestra confirmación del envío 13. **instructor**: cierra la app
excepción	- en caso de no haber conexión, la app guarda la observación y trata de enviarla luego, mostrando notificación cuando lo logró - si no hay ID en el activo, ese campo es opcional - si no ve necesario tomar foto, ese campo es opcional

CU-05	**Ver historial de cumplimiento de horario**
precondición	debe haber registros previos del usuario buscado en el sistema, el cuál ha estado haciendo uso del salón que le fué asignado, además debe contarse con conexión a Internet, quien busca ya está navegando en la página
secuencia	1. **funcionario**: va al apartado de búsqueda de usuario 2. **página**: muestra las opciones de filtro y un resultado vacío 3. **funcionario**: escribe la información de identificación del instructor 4. **funcionario**: pulsa buscar 5. **página**: envía la solicitud de búsqueda al servidor 6. **servidor**: responde con los datos obtenidos 7. **página**: muestra en pantalla los salones asociados al instructor, listados por fechas de permisos asignados, más recientes primero 8. **funcionario**: elige qué salon quiere revisar 9. **página**: muestra el horario, la lista de accesos y el porcentaje de cumplimiento en cada uno, además de un porcentaje global para el periodo asignado
excepción	- si no hay datos del usuario buscado, el sistema dirá que no los hay

CU-06	**Creación de nuevo usuario**
precondición	el administrador debe tener su programa de gestión abierto, con conexión al sistema y debe estar previamente logueado en su cuenta, el software de escritorio debe correr en un computador que tenga correctamente conectado un sensor biométrico, el mismo debe estar ya reconocido por el software, el usuario (por ejemplo un instructor) que va a ser registrado, debe estar ahí presencialmente para poder tomarle sus datos biométricos
secuencia	1. **admin**: va al apartado de gestión de usuarios 2. **admin**: da clic en crear nuevo usuario 3. **desktop**: muestra el formulario para ingresar datos 4. **admin**: ingresa los datos del nuevo instructor 5. **admin**: le asigna una contraseña inicial, puede ser la misma cédula 6. **admin**: selecciona el rol y centro al que pertenece 7. **admin**: pulsa en crear 8. **desktop**: verifica los datos y los envía al servidor 9. **servidor**: verifica y guarda los datos, retorna una confirmación 10. **desktop**: muestra la confirmación y redirige al perfil creado 11. **admin**: pulsa en agregar nuevo método de identificación 12. **desktop**: muestra el formulario de método de identificación 13. **admin**: selecciona la opción de huella dactilar 14. **desktop**: se pone en modo escucha respecto al dispositivo huellero 15. **admin**: le pide al instructor que ponga el dedo en el huellero 16. **instructor**: pone el dedo 17. **desktop**: obtiene el dato biométrico y lo envía al servidor 18. **servidor**: acepta el dato biométrico y devuelve confirmación 19. **desktop**: vuelve al perfil, mostrando que se aceptó la huella

excepción	- si hay datos mal diligenciados el sistema advierte - si se cae la conexión, no se podrá hacer el trámite y el sistema lo dice - el paso de contraseña inicial, puede ser que el mismo instructor la escriba, para evitar que luego la tenga que cambiar en la App de observaciones(dos pasos) - si el sensor biométrico leyó mal, pedirá otro intento - si se quiere agregar al usuario a uno o varios grupos de usuarios, puede hacerse de una vez

CU-07	**Asignación de salón a usuario**
precondición	el administrador tiene acceso y login al sistema de configuración, el mismo debe tener conexión, debe haber al menos un usuario instructor y un salón registrados en la base de datos, para poder hacer una asignación directa
secuencia	1. **admin**: va al apartado de búsqueda de usuarios 2. **desktop**: muestra la lista de resultados vacía y los filtros 3. **admin**: ingresa algún identificador del instructor 4. **admin**: pulsa buscar 5. **desktop**: envía la petición de usuarios al servidor 6. **servidor**: entrega una lista de resultados 7. **desktop**: muestra la lista de resultados 8. **admin**: selecciona el resultado que desea, el instructor que buscaba 9. **desktop**: muestra el perfil del instructor 10. **admin**: pulsa en ver salones asignados 11. **desktop**: solicita al servidor los salones asignados al instructor 12. **servidor**: devuelve la lista de salones asignados 13. **desktop**: muestra la lista de salones asignados 14. **admin**: pulsa en hacer nueva asignación 15. **desktop**: muestra el formulario para la asignación 16. **desktop**: pide la lista de salones al servidor 17. **servidor**: entrega la lista de salones 18. **desktop**: pone la lista de salones en el buscador del formulario 19. **admin**: busca el salón deseado, quizá usando filtros 20. **admin**: coloca la fecha de inicio y fin de la asignación 21. **admin**: coloca el horario, seleccionando qué franjas de la semana serán asignadas, ej: martes mañana, jueves tarde, viernes noche 22. **admin**: pulsa en asignar 23. **desktop**: envía los datos al servidor para asignación 24. **servidor**: guarda la información y devuelve verificación 25. **desktop**: muestra el mensaje de éxito y vuelve a la lista de asignaciones de salones al usuario
excepción	- si hay datos mal diligenciados el sistema advierte - si se cae la conexión, no se podrá hacer el trámite y el sistema lo dice - si hay un usuario ya asignado a ese salón en ese horario, el sistema muestra la advertencia e interrumpe el proceso

J. Diagramas de Secuencias

Sirve para ordenar los flujos de datos entre componentes y la temporalidad con que se mueven, en el diagrama podemos apreciar el mecanismo de apertura y cierre con bloqueo permisivo, pudiendo ser cierre con bloqueo total, si inhibimos la acción del usuario “switch permisivo”.

Como se ha anotado en los casos de uso, no se incluirá la comunicación que permite guardar el estado de la puerta en el servidor, por propósitos de simpleza, también que, cuando el servidor se desconecta de la red, el mismo controlador de la puerta tratará de buscar los datos de permisos en su propia memoria.

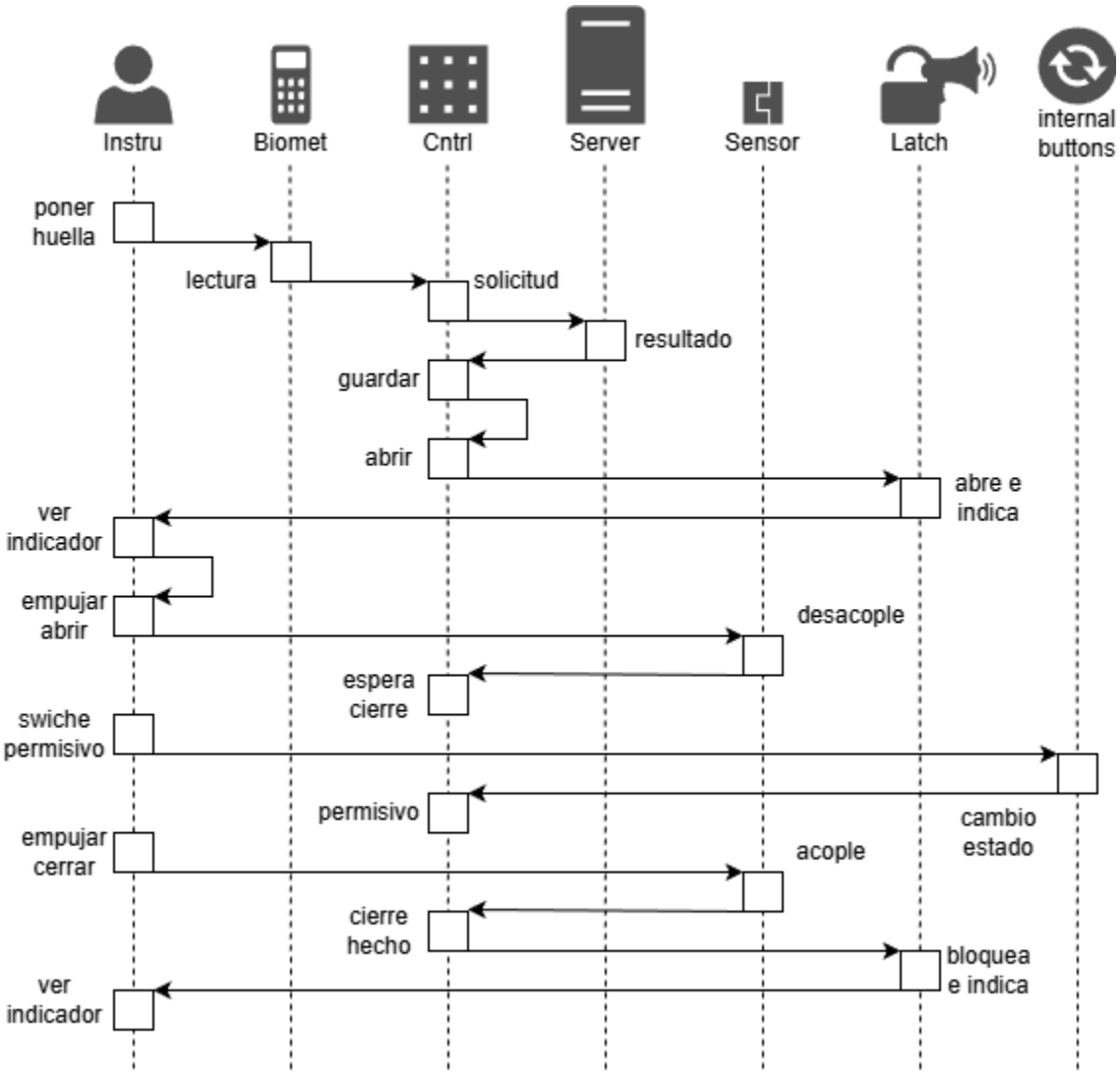

Figura J1

En el siguiente diagrama vemos el proceso para salir de un salón, no se requieren permisos y nuevamente, se ha dejado implícita la comunicación con el servidor para notificar historial. Si el

aprendiz desea reingresar, podrá hacerlo con la misma secuencia, usando el biomet como comando de entrada, pero esto sólo es posible si desde dentro el sistema se estableció como permisivo, así como se ve en la figura anterior.

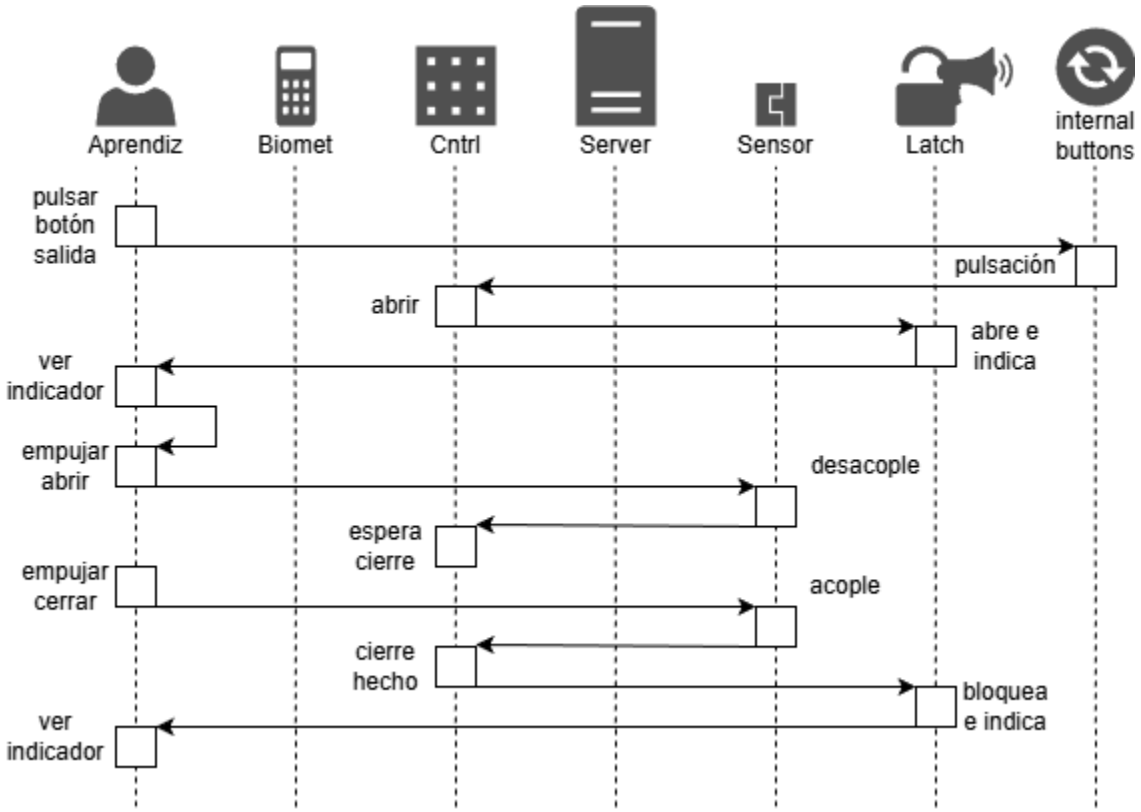

Figura J2

Las demás secuencias se observan claramente en los casos de uso, se han creado las dos asociadas al uso de la puerta, puesto que involucran muchos subsistemas, como lo son sensores o módulos.

K. Base de Datos

Este diagrama MER por simplicidad omite los atributos, las principales entidades son los usuarios y los salones, ambos tienen en común la asociación de centros, un instructor de un centro puede usar un salón de otro centro, es una dinámica común en el SENA. Luego la asignación de usuarios a salones es la forma en que se da permiso 1:1, además da la posibilidad de seguimiento, para ver el cumplimiento de horarios por parte de un instructor.

Las historias o historial, es la relación que hay de usuarios a salones cuando interactúan, entonces cada acceso o bloqueo genera un registro histórico. Por otro lado, la relación de permitir o permiso que hay entre grupos de usuarios y zonas de salones, es lo que genera permisos de entrada N:M. Notemos que los usuarios se pueden asociar a varios grupos, mientras las áreas sólo a una zona, esto último podría cambiarse si se considera necesario.

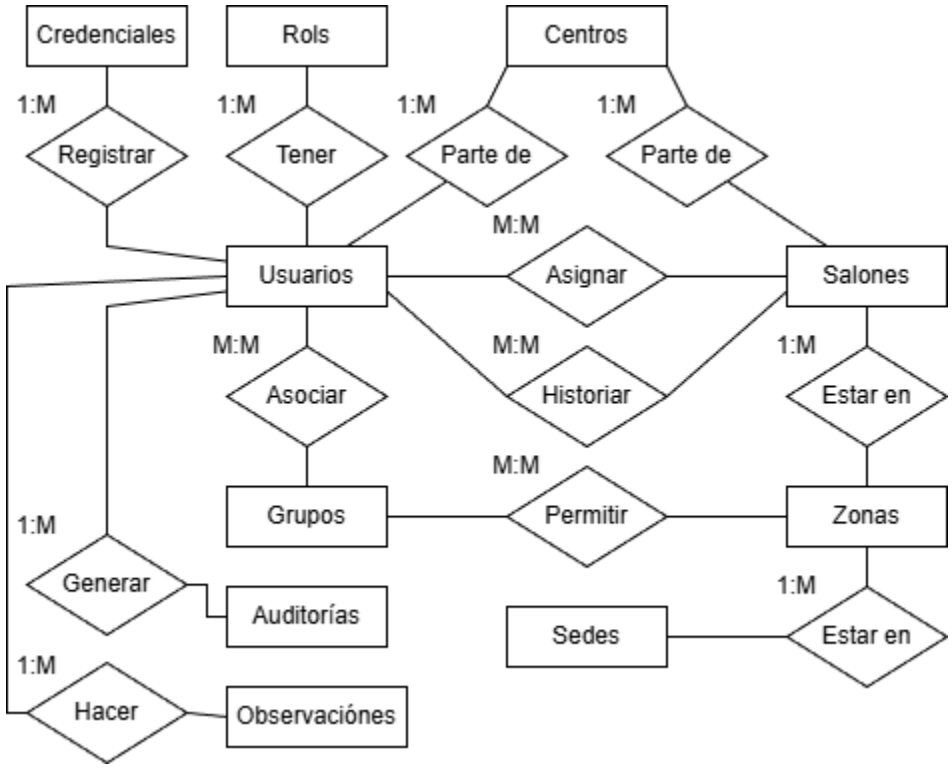

Figura K1

Los roles de usuario y la asociación de usuarios y salones a centros, no generan por sí mismos permisos, los permisos como ya se explicó van por otro camino, esto es para evitar tener múltiples caminos en los permisos, se centralizan en grupos vs zonas, para evitar más lazos o bucles en la database, Así es más eficiente tener rol: instructor_planta, y grupo: instructor_adso, ¿hay conflictos con la normalización? no, el instructor_planta es un rol global omnipresente en el SENA a nivel nacional, si se busca una atomicidad, grupos debería conectarse a roles, pero un usuario puede tener muchos grupos, entonces podría haber conflictos con múltiples roles institucionales por usuario. Resumiendo, grupos da libertad

creativa al administrador para ordenar a los usuarios independientemente de qué roles y a qué centros pertenezcan, sin perder de foco el rol y centro exacto al que pertenece el usuario.

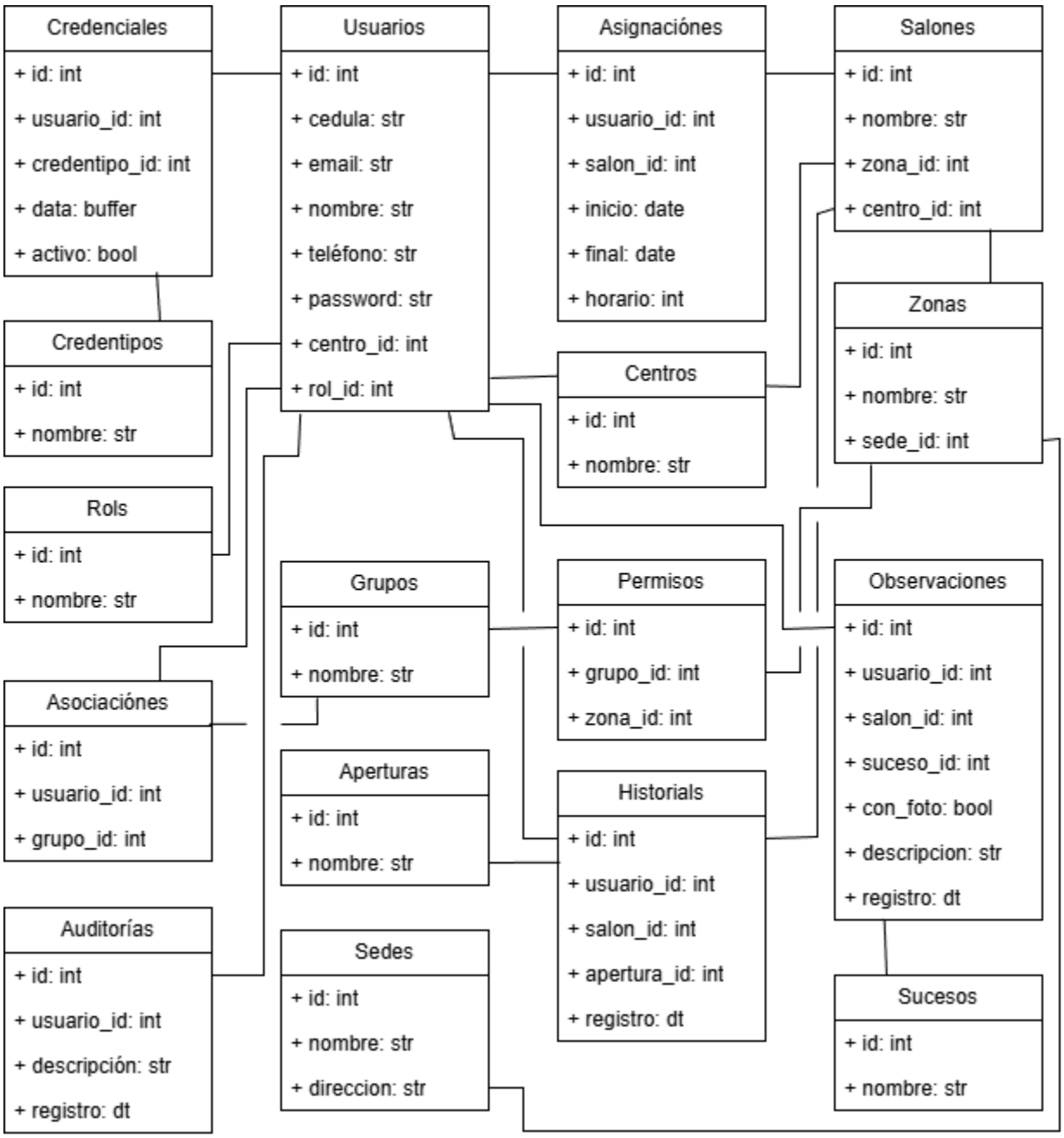

Figura K2

El modelo relacional, genera unas tablas extra, y muestra los atributos, todo en el marco de los requerimientos. La tabla credentipos lista las posibles credenciales, por ej, huella, iris, tarjeta, código, Qr, facial, etc. Mientras sucesos lista etiquetas para el tipo de observación, y aperturas lista estados de la puerta: bloqueada, desbloqueada, abierta, etc.

L. Montaje Arduino

El fin de un semillero de investigación, es promover la adquisición de conocimiento por parte de los aprendices, en esa línea de pensamiento y a sabiendas de que estos sistemas automatizados ya existen, y se venden bajo estándares de alto rendimiento, se plantea a modo experimental el montaje de una maqueta a escala, impulsada por la barata y ampliamente usada en enseñanza, tecnología de Arduino.

El siguiente montaje, réplica en el simulador **TinkerCad** [4], el diagrama de salón con puerta inteligente visto en la Figura D1, se ha puesto un motor eléctrico como representación del inductor del electroimán que fija la puerta, un teclado matricial como representación del huellero biométrico, un suiche (derecha) como sensor de puerta abierta, otro (izquierda) como activador del modo permisivo. Un LED RGB para mostrar el estado de la puerta y un pulsador para poder abrir desde dentro. Este montaje podría hacerse en una puerta de madera de balso para un entendimiento más lúdico del sistema, y su conexión serial con software para la GUI.

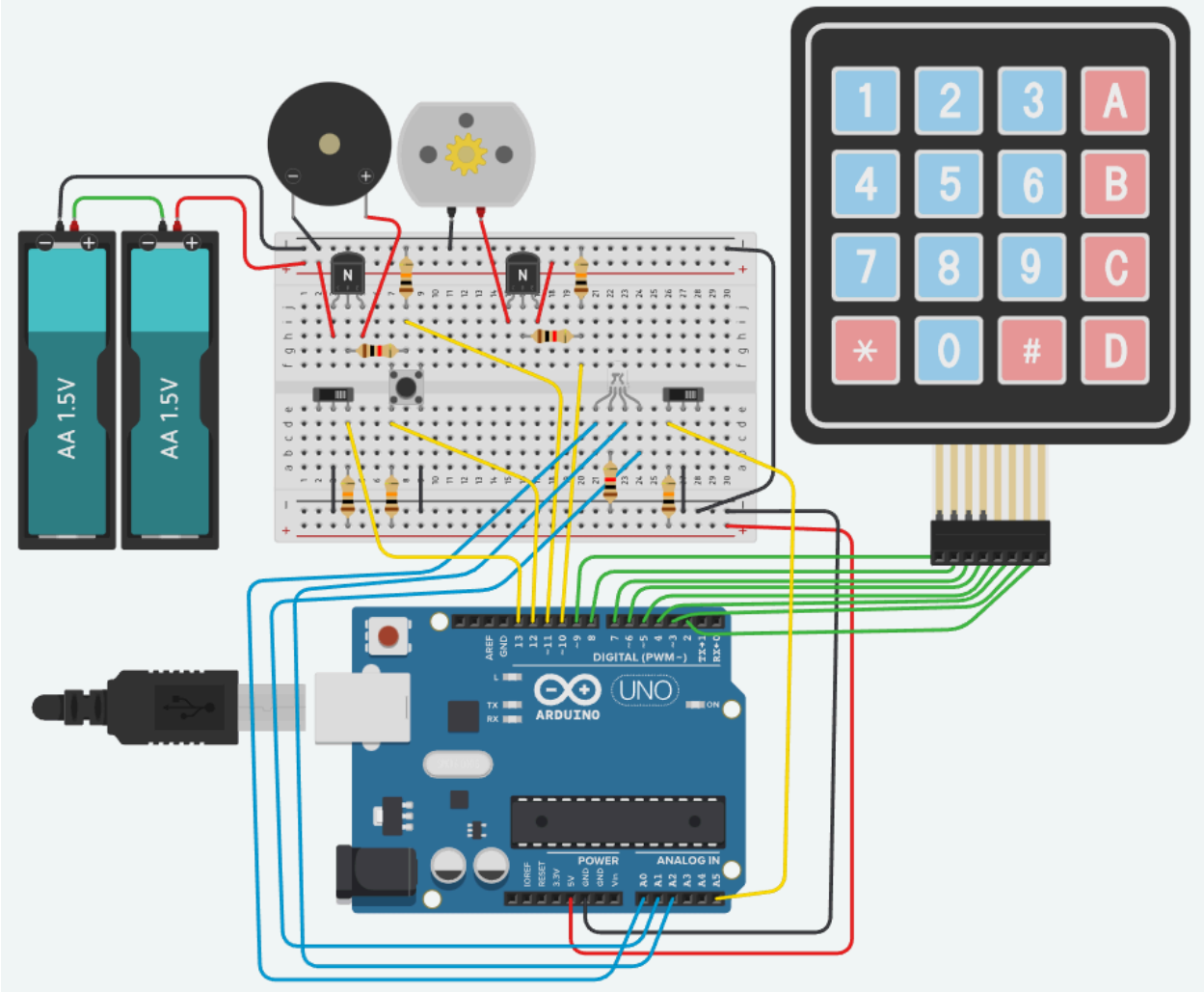

Figura L1

Para estructurar mejor el funcionamiento del sistema, se ha utilizado el concepto de **máquina de estados finitos (FSM)**, en la siguiente imágen observamos cómo funciona.

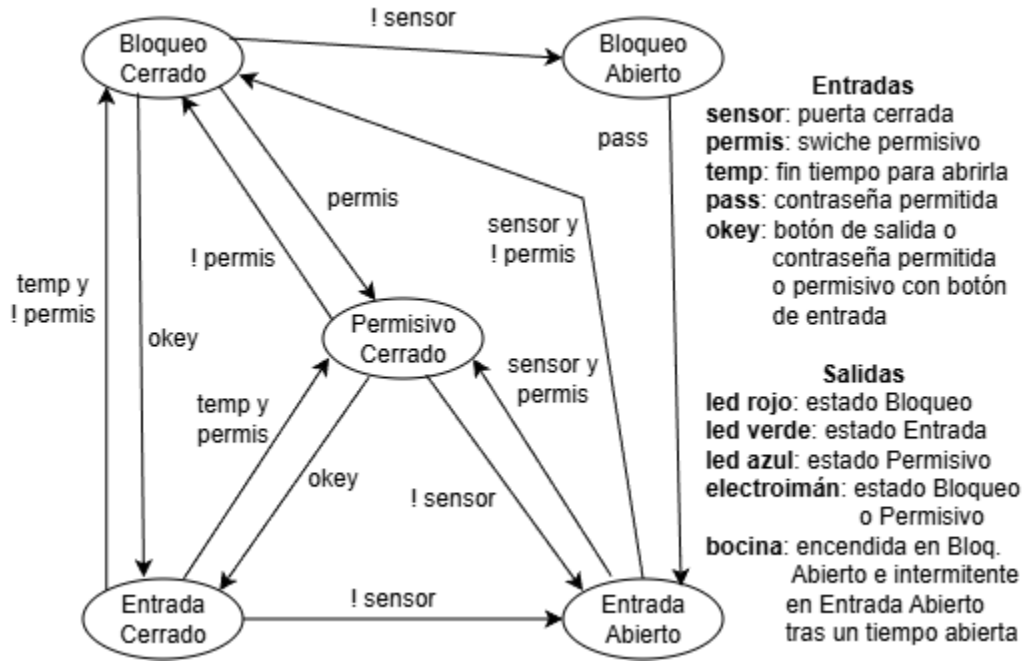

Figura L2

A continuación el código. Está limitado a almacenar en memoria un sólo password, y su interfaz con el PC sólo permite modificar dicha clave, claramente es trabajo a futuro expandir el ejemplo para que obtenga datos de una DB solicitándolos al servidor (ejemplo, Python con XAMPP), y sólo usando su memoria cuando no hay respuesta.

El código funciona implementando la máquina de estados, es además configurable mediante macros, las funciones **stateLoop**(), **stateChange**(), **stateOutput**() manejan la lógica del estado, la cuál permite hacer cambios y finalmente setear las salidas. Para obtener fácilmente las entradas, se llama a **getInputs**() quien no sólo lee los sensores sino que procesa lógica interna de acceso, como la vista en la figura L2.

La función **setCodigo**() lee la información que es enviada desde el PC vía comunicación serial, mientras **stepTeclado**() obtiene las pulsaciones del teclado matricial y las va guardando en un string. Por otra parte **openString**() y **saveString**() se encargan del almacenamiento en la memoria EEPROM, la cuál es no volátil.

La ejecución de Arduino consta de dos funciones principalmente, **setup**() y **loop**() por sus nombres es notorio que la primera se ejecuta una vez al inicio y la segunda se ejecuta a máxima velocidad como un bucle infinito. En está última se ha usado el concepto de **deltaTime**() el cuál permite mover contadores en tiempo real, evitando llamar funciones que causen bloqueos y retrasos significativos al hardware.

```
// ejemplo Arduino para maqueta de puerta con password

// biblioteca para manejo de teclado matricial y memoria
#include <Keypad.h>
#include <EEPROM.h>

/*
Entradas:
- swiche para modo permisivo (dentro del salon)
- pulsador para abrir desde dentro del salon
- sensor para saber si esta cerrada o abierta

Salidad:
- electroiman de la puerta
- bocina para alarmar
- led rojo: bloqueo
- led verde: apertura
- led azul: modo permisivo
*/

// asignacion de pines fisicos
#define BTN_PERMIS 13
#define BTN_SALIDA 12
#define BTN_SENSOR A5
#define ACT_PUERTA 10
#define ACT_SONIDO 11
#define LED_ROJO A0
#define LED_VERDE A1
#define LED_AZUL A2

// para navegar en el array de la funcion que lee inputs
#define INP_PERMIS 0
#define INP_SALIDA 1
#define INP_SENSOR 2
#define INP_PASS 3

// talla de los password
#define SIZE_PASSWORD 4
```

```
// especio en memoria EEPROM donde se guardara el password
#define DIR_MEMORY 6

// constantes de tiempos, contadores, temporizadores
#define TIME_LIMIT_ENTRAR 3 // 5s
#define TIME_START_PITIDO 6 // 15s
#define TIME_OSCILA_PITIDO 0.2 // 0.5s

// constantes de la maquina de estados finitos
#define EST_BLQ_CLS 0
#define EST_BLQ_OPN 1
#define EST_PER_CLS 2
#define EST_ENT_CLS 3
#define EST_ENT_OPN 4

// variable de estado para la maquina de estados
char state = EST_ENT_OPN;

// variables de temporizadores, contadores
float time_limit_entrar = 0;
float time_start_pitido = 0;
float time_oscila_pitido = 0;

// configuracion del teclado matricial
char keys[4][4] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};
byte pinesF[4] = {9, 8, 7, 6};
byte pinesC[4] = {5, 4, 3, 2};
Keypad teclado = Keypad(makeKeymap(keys),
  pinesF, pinesC, 4, 4);
// guarda el ultimo caracter ingresado
char tecla;

// variable para leer el teclado y obtener el password
String codigo = "";
```

```
// string que guarda el password de acceso
String password;

// variable para calculo del delta de tiempo
unsigned long previo;

void setup() {
  // comienzo modo de transmision serial
  Serial.begin(9600);
  // asignacion de modo input/output de pines
  pinMode(ACT_PUERTA, OUTPUT);
  pinMode(ACT_SONIDO, OUTPUT);
  pinMode(LED_ROJO, OUTPUT);
  pinMode(LED_VERDE, OUTPUT);
  pinMode(LED_AZUL, OUTPUT);
  pinMode(BTN_PERMIS, INPUT);
  pinMode(BTN_SALIDA, INPUT);
  pinMode(BTN_SENSOR, INPUT);
  // cargar el password de la memoria
  password = openString(DIR_MEMORY);
  // enviar mensaje inicial
  Serial.println("...");
  Serial.println("***Bienvenid@ a Puerta Segura***");
  Serial.print("digite y envie un password de ");
  Serial.print(SIZE_PASSWORD);
  Serial.println(" digitos");
  Serial.println("puede cambiarlo en cualquier momento");
  if (password != "") {
  Serial.print("Password: ");
    Serial.println(password);
  }
  Serial.println("...");
  // inicializar el estado inicial
  stateChange(state);
  // inicializar la variable para delta de tiempo
  previo = millis();
}
```

```
void loop() {
  // ejecuta muchas veces la maquina de estados
  float dt = deltaTime();
  stateLoop(dt);
  stateOutput(dt);
  // verifica si hay una nueva paswords desde el PC
  setCodigo();
  // trata de construir un password desde el teclado
  stepTeclado();
}

void stateLoop(float dt) {
  // primero obtiene los inputs del sistema
  bool inputs[4];
  getInputs(inputs);
  // depura los estados, cada uno hace algo diferente
  switch (state) {

    case EST_BLQ_CLS:
      // bloqueada y cerrada, puede ser abierta legalmente
      if (inputs[INP_SALIDA]) {
        stateChange(EST_ENT_CLS);
      }
      // pasar a modo permisivo desde dentro
      else if (inputs[INP_PERMIS]) {
        stateChange(EST_PER_CLS);
      }
      // o en el peor de los casos ser forzada, dispara alarma
      else if (!inputs[INP_SENSOR]) {
        stateChange(EST_BLQ_OPN);
      }
      break;

    case EST_BLQ_OPN:
      // modo alarma, debe usarse el password para silenciarla
      if (inputs[INP_PASS]) {
        stateChange(EST_ENT_OPN);
      }
      break;
```

```
case EST_PER_CLS:
  // en modo permisivo puede ser abierta
  if (inputs[INP_SALIDA]) {
    stateChange(EST_ENT_CLS);
  }
  // ser desactivado su modo permisivo
  else if (!inputs[INP_PERMIS]) {
    stateChange(EST_BLQ_CLS);
  }
  // o ser abierta forzadamente pero legal a la vez
  else if (!inputs[INP_SENSOR]) {
    stateChange(EST_ENT_OPN);
  }
  break;

case EST_ENT_CLS:
  // temporizador para poder ser abierta manualmente
  time_limit_entrar = min(time_limit_entrar + dt,
                          TIME_LIMIT_ENTRAR);
  // fue abierta manualmente
  if (!inputs[INP_SENSOR]) {
    stateChange(EST_ENT_OPN);
  }
  else if (time_limit_entrar == TIME_LIMIT_ENTRAR) {
    // sino, volvera a un estado de cierre
    if (inputs[INP_PERMIS]) {
      stateChange(EST_PER_CLS);
    }
    else {
      stateChange(EST_BLQ_CLS);
    }
  }
  break;

case EST_ENT_OPN:
  // temporizador para empezar a pitar por apertura
  time_start_pitido = min(time_start_pitido + dt,
                          TIME_START_PITIDO);
```

```
      if (inputs[INP_SENSOR]) {
        // de ser cerrada volvera a un estado de cierre
        if (inputs[INP_PERMIS]) {
          stateChange(EST_PER_CLS);
        }
        else {
          stateChange(EST_BLQ_CLS);
        }
      }
      break;
  }
}

void stateChange(char new_estate) {
  // los cambios de estado se mandan al PC
  state = new_estate;
  switch (state) {
    case EST_BLQ_CLS:
      Serial.println("Bloqueo Close");
      break;
    case EST_BLQ_OPN:
      Serial.println("Bloqueo Open");
      break;
    case EST_PER_CLS:
      Serial.println("Permisivo Close");
      break;
    case EST_ENT_CLS:
      Serial.println("Entrar Close");
      // resetear el temporizador de espera a la apertura
      time_limit_entrar = 0;
      break;
    case EST_ENT_OPN:
      Serial.println("Entrar Open");
      // resetear los temp. para pitido por apertura demorada
      time_start_pitido = 0;
      time_oscila_pitido = 0;
      break;
  }
}
```

```
void stateOutput(float dt) {
  // coloca el color de los leds segun el estado
  digitalWrite(LED_ROJO,
               state == EST_BLQ_CLS ||
               state == EST_BLQ_OPN);
  digitalWrite(LED_VERDE,
               state == EST_ENT_CLS ||
               state == EST_ENT_OPN);
  digitalWrite(LED_AZUL,
               state == EST_PER_CLS);
  // activa el electroiman segun el estado
  digitalWrite(ACT_PUERTA,
               state == EST_BLQ_CLS ||
               state == EST_BLQ_OPN ||
               state == EST_PER_CLS);
  // pone a sonar la alarma si hubo ingreso forzado
  if (state == EST_BLQ_OPN) {
    analogWrite(ACT_SONIDO, 50);
  }
  // sonara un pitido si ha pasado tiempo abierta
  else if (state == EST_ENT_OPN) {
    if (time_start_pitido == TIME_START_PITIDO) {
      // aca se calcula el pitido, usa dt para no usar delay
      time_oscila_pitido += dt;
      if (time_oscila_pitido > TIME_OSCILA_PITIDO) {
        time_oscila_pitido -= 2 * TIME_OSCILA_PITIDO;
      }
      // coloca el estado de la bocina como tal
      if (time_oscila_pitido > 0) {
        analogWrite(ACT_SONIDO, 50);
      }
      else {
        analogWrite(ACT_SONIDO, 0);
      }
    }
    else {
      analogWrite(ACT_SONIDO, 0);
    }
  }
```

```
  // desactiva la alarma en otros estados
  else {
    analogWrite(ACT_SONIDO, 0);
  }
}

float deltaTime() {
  // calcula el delta de tiempo para codigos no bloqueantes
  unsigned long actual = millis();
  float dt = (actual - previo) / 1000.0;
  previo = actual;
  return dt;
}

void getInputs(bool inputs[4]) {
  // lectura de los sensores
  inputs[INP_PERMIS] = digitalRead(BTN_PERMIS);
  inputs[INP_SALIDA] = !digitalRead(BTN_SALIDA);
  inputs[INP_SENSOR] = digitalRead(BTN_SENSOR);
  inputs[INP_PASS] = false;
  // verifica el ingreso con password
  if (codigo == password && password != "") {
    codigo = "";
    inputs[INP_SALIDA] = true;
    inputs[INP_PASS] = true;
  }
  // en modo permisivo puede ingresar con cualquier tecla
  else if (tecla && state == EST_PER_CLS) {
    inputs[INP_SALIDA] = true;
  }
}

void stepTeclado() {
  // captura las teclas pulsadas y construye password
  tecla = teclado.getKey();
  if (tecla) {
    // agregar la tecla al password ingresado
    if (codigo.length() >= SIZE_PASSWORD) {
      codigo = codigo.substring(1);
```

```
    }
    codigo += tecla;
  }
}

void setCodigo() {
  // revisa si llegan mensajes del PC para password nuevo
  if (Serial.available()) {
    String input = Serial.readStringUntil('\n');
    // solo digitos de la cadena
    String digitos = "";
    for (char c: input) {
      if (c >= '0' && c <= '9') {
        digitos += c;
      }
    }
    // asignar solo la cadena tiene la talla deseada
    if (digitos.length() == SIZE_PASSWORD) {
      password = digitos;
      Serial.print("Password: ");
      Serial.println(password);
    }
    // sino resetea el password
    else {
      password = "";
      Serial.println("Password: vacío");
    }
    // guardar password en memoria no volatil
    saveString(DIR_MEMORY, password);
  }
}

String openString(int address) {
  int len = EEPROM.read(address);
  String result = "";
  for (int i = 0; i < len; i++) {
    result += char(EEPROM.read(address + 1 + i));
  }
  return result;
```

```
}

void saveString(int address, String data) {
  int len = data.length();
  EEPROM.write(address, len);
  for (int i = 0; i < len; i++) {
    EEPROM.write(address + 1 + i, data[i]);
  }
}
```

En la siguiente figura, el diagrama circuital para hacer la conexión de los componentes.

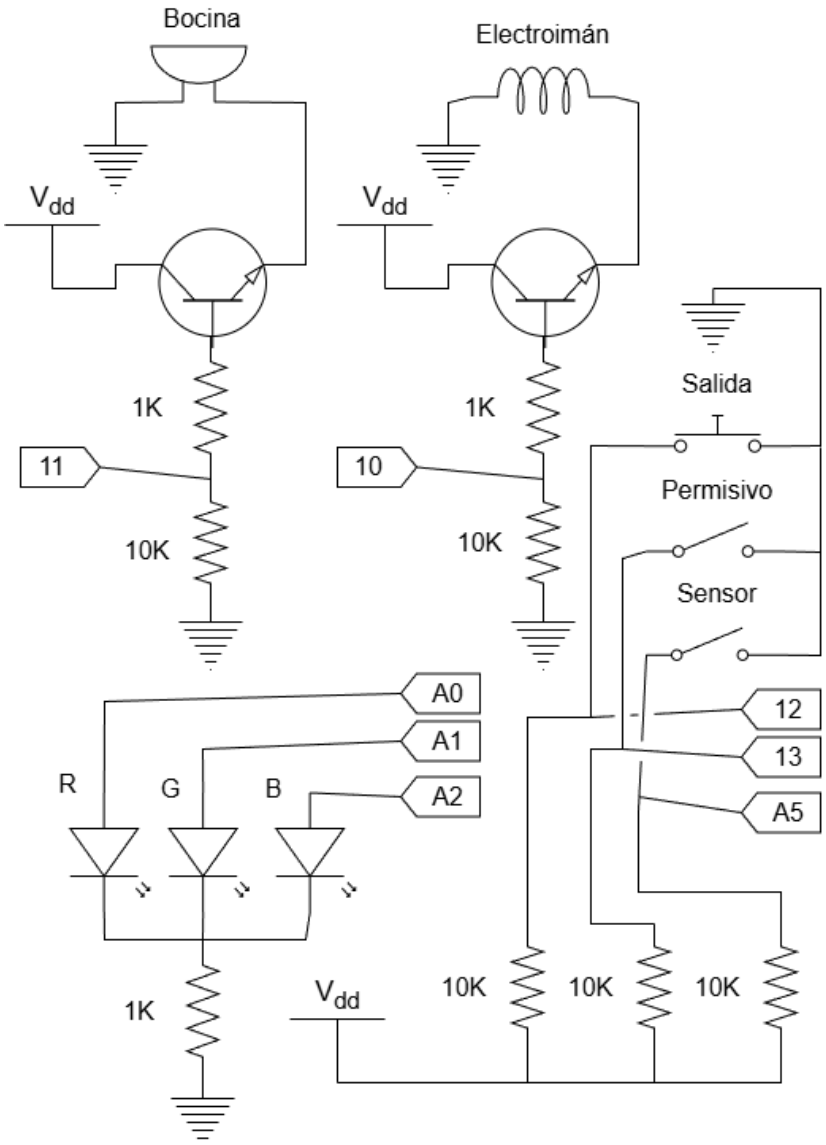

Figura L3

M. Referencias

[1] Scientia et Technica Año XVII, No 46, Diciembre 2010. Universidad Tecnológica de Pereira.

[2] Narayan, L & G, Sonu & M, Soukhya. (2020). Fingerprint Recognition and its Advanced Features. International Journal of Engineering Research and. V9. 10.17577/IJERTV9IS040393.

[3] JM-101 Optical FingerprintModule User manual, Version 1.8, Jan 2017

[4] https://www.tinkercad.com/things/jIXyXmYoftg-puertasegura