FACULTAD
DE INGENIERIA
Universidad de Buenos Aires

CARRERA DE ESPECIALIZACIÓN EN
INTERNET DE LAS COSAS

MEMORIA DEL TRABAJO FINAL

Sistema de gestión de alertas y tareas de procesos de planta - Control de acceso

Autor:
Lionel Gutierrez

Director:
Gustavo Ramoscelli (UNS)

Jurados:
José Alamos (HAW Hamburg)
Leandro Lanzieri Rodriguez (UTN FRA/HAW Hamburg)
Leopoldo Zimperz (UBA)

Este trabajo fue realizado en la ciudad de Villa Mercedes,
entre mayo de 2020 y marzo de 2021.

Resumen

La presente memoria describe el diseño e implementación de un sistema de control de acceso de personal de terceros a una locación industrial. El sistema garantiza que solo aquellas personas que tienen en regla los requisitos legales y médicos solicitados accedan, evitando que la empresa sea responsable ante posibles accidentes o incidentes de dicho personal. El trabajo desarrollado es la primera etapa de un proyecto integral de gestión de alertas y procesos para la empresa Tenaris Metalmecánica, sobre el cual se agregarán a futuro nuevos casos de uso.

Para la elaboración del trabajo se aplicaron conocimientos adquiridos a lo largo de la carrera, principalmente los referidos a gestión de proyectos, desarrollo de aplicaciones web y multiplataforma, protocolos de Internet y seguridad en IoT. Además, se integraron tecnologías de base de datos relacionales y no relacionales y se aplicaron varias técnicas de testing.

Agradecimientos

AGREGAR AGRADECIMIENTO

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción general

Poner párrafo introductorio.

1.1. Estado del arte

Introducción, propósito y estado del arte de IoT y solución propuesta.

1.1.1. Tecnología IoT

Introducción a las soluciones IoT, posibilidades que brinda para sensado y control de diferentes procesos, ventajas de la tecnología (economía, simplicidad).

1.1.2. Control de acceso

Sistemas de control de acceso que existen en el mercado. Diferencias con el sistema propuesto (valor agregado de la propuesta contra soluciones existentes).

1.2. Motivación

Razones por el cual se desea desarrollar el sistema. Justificaciones. Necesidad del cliente.

1.3. Objetivos y alcance

Objetivos y alcance del trabajo.

Capítulo 2

Introducción específica

Poner párrafo introductorio.

2.1. Protocolos de comunicación

Descripción de los protocolo de comunicación (Wi-Fi/HTTP) utilizados para IoT.

2.1.1. Tecnología de comunicación Wi-Fi

Descripción de tecnología Wi-Fi.

2.1.2. Protocolo HTTP

Descripción de protocolo HTTP.

2.2. Componentes de hardware utilizados

Descripción de los componentes de hardware utilizados: ESP32, lector de tarjetas, cerradura electrónica.

2.2.1. Módulo ESP32

Descripción del módulo.

2.2.2. Módulo RFID RC522

Descripción del módulo.

2.2.3. Cerradura electrónica

Descripción de cerradura.

2.3. Tecnologías de software aplicadas

Descripción de las tecnologías de software utilizadas.

2.3.1. Node.JS

2.3.2. Ionic

2.3.3. PostgreSQL

2.3.4. MongoDB

2.3.5. Docker

2.3.6. Postman

2.4. Software de control de versiones

Descripción del software de control de versiones.

2.4.1. GitFlow

Descripción de la herramienta.

2.5. Requerimientos

Requerimientos del proyecto, tanto funcionales, no funcionales, de documentación y de validación. Enumeración de los mismos.

2.5.1. Requerimientos funcionales

1.1 El sistema debe permitir el sensado de datos de distintas fuentes y procesos de planta.

1.2 El sistema deberá generar alertas a usuarios finales ante problemas detectados del sensado o situaciones límites/problemas potenciales.

1.3 El sistema deberá generar tareas de corrección y prevención con un circuito de estados que permita trazar el origen del problema y la solución asociada.

1.4 El sistema debe permitir hacer un seguimiento de la cantidad de alarmas diarias y mensuales generadas.

1.5 El sistema debe permitir hacer un seguimiento de la cantidad de tareas diarias y mensuales generadas A su vez, se deberá poder ver la cantidad de tareas cerradas, en curso y su antigüedad en días.

1.6 El sistema debe permitir gestionar usuarios. La gestión de usuarios incluye dar de alta nuevos usuarios, gestionar la recuperación y cambio de clave de los mismos. Dicho usuario se utilizará para acceder y utilizar el sistema.

2.5.2. Requerimientos no funcionales

2.1 El sistema deberá ser escalable, de forma de poder agregar más módulos actuadores y de sensado para los procesos de planta a futuro.

2.2 El sistema deberá ser recuperable ante problemas de hardware o software, de forma de asegurar la disponibilidad y no corrupción de la información, cumpliendo con la política de resguardo de datos de la empresa.

2.3 El sistema deberá poder operarse aún ante cortes puntuales de energía en algunas áreas, esto es, ante corte que no sean generales de toda la planta. Para ello se deberá contar con una política de suministro alternativo de energía para los servidores donde se ejecute el software.

2.5.3. Requerimientos de documentación

3.1 Se debe generar una Memoria Técnica con la documentación de ingeniería detallada.

3.2 Se debe generar un documento de casos de prueba.

3.3 Se debe generar un documento de la Infraestructura del sistema y de la configuración por ambiente y pasaje entre ambientes.

3.4 Se deberá generar la documentación del sistema y del proyecto en el sistema de aprobación y documentación TPA de la empresa.

2.5.4. Requerimientos de validación

4.1 Se deberá tener una matriz de trazabilidad entre los casos de uso y los casos de prueba, validando el cumplimiento de cada uno y con la aprobación final del auspiciante.

Capítulo 3

Diseño e implementación

Poner párrafo introductorio.

3.1. Arquitectura del sistema

Arquitectura del sistema. Separación en módulos/subsistemas. Protocolos de comunicación entre los módulos y escalabilidad.

3.2. Detalle de módulos de hardware

Detalle de los módulos del sistema y protocolos utilizados para la comunicación entre los mismos.

3.2.1. Módulo sensor

Detalle de implementación del módulo de sensado/ingreso.

3.2.2. Módulo actuador

Detalle de implementación del módulo de actuación.

3.3. Detalle de módulos de software

3.3.1. Módulo de backend

Detalle de implementación backend del sistema. Explicar por un lado la API Rest del sistema y el detalle de implementación del middleware de autenticación (API de autenticación).

3.3.2. Módulo de frontend

Detalle de implementación frontend del sistema.

3.4. Interfaz con sistema de documentación

Detalle de interfaz con sistema de documentación. Mock implementado para testeo integral del sistema.

Capítulo 4

Ensayos y resultados

En el presente capítulo se describen las pruebas realizadas sobre el sistema desarrollado y se muestran los resultados obtenidos.

4.1. Detalle de pruebas realizadas

Para planificar y gestionar todo el proceso de pruebas se desarrolló un *Master Test Plan* [1], donde se especificaron los objetivos de las pruebas, los responsables, la estrategia general y la estrategia por niveles de prueba.

Los objetivos de las pruebas realizadas sobre el software y hardware fueron:

- Determinar si el sistema cumple con los requerimientos especificados en la sección 2.5.
- Reportar las diferencias entre lo observado y el comportamiento deseado.
- Dejar evidencias y documentación para probar las siguientes versiones del software y solucionar cualquier *bug* detectado.

Para dar cuenta del proceso, en la tabla 4.1, se muestra como se organizaron los tipos de prueba realizadas.

Para lograr una trazabilidad entre los requerimientos y los casos de test se implementó, por un lado, una matriz de trazabilidad entre los requerimientos y los casos de uso definidos para el sistema, y por el otro, una matriz de trazabilidad entre los casos de uso y los casos de test. Esto quedó registrado en el documento de Casos de Uso y Casos de Test del sistema [3].

TABLA 4.1. Tipos de pruebas realizadas sobre el sistema.

Tipo de Prueba	Descripción y objetivo	Herramientas utilizadas/Modo de prueba
Pruebas unitarias	Este tipo de pruebas permitió verificar de forma separada cada uno de los módulos de hardware y software del sistema. Para aquellos que tenían interfaces con otros módulos se utilizaron *mocks* [2], de forma de simular el comportamiento de los mismos sin necesidad de tenerlos implementados.	Postman/Newman. *mocks* de módulos en caso de ser necesario.
Pruebas de sistema	Este tipo de prueba permitió verificar el sistema de manera integral, asegurando la correcta comunicación e interrelación de los módulos.	Pruebas en ambiente de desarrollo sobre el sistema, con los módulos implementados.
Pruebas de aceptación	Este tipo de pruebas permitió validar el sistema de manera integral junto al cliente. De esta forma se aseguró no solo que el sistema se comporte según lo especificado y desarrollado, sino que el usuario final corrobore que el sistema actúa según sus necesidades y requisitos.	Pruebas en ambiente de desarrollo sobre el sistema, con los módulos implementados.

4.1.1. Herramientas utilizadas

Para la realización de las pruebas unitarias se utilizó Postman, la cual permitió centralizar y gestionar todas las pruebas desde una única herramienta. Además, se utilizó Newman para automatizar la ejecución de las mismas.

En la figura 4.1 se muestra la herramienta Postman, junto a la organización de las pruebas unitarias, separadas por módulo y funcionalidad a testear.

Por su parte, Newman permite importar los conjuntos de tests definidos en Postman, y mediante un script, ejecutar los mismos y ver los resultados obtenidos. De esta manera podemos correr todos los tests automáticamente, de forma rápida y simple, sin necesidad de hacerlo manualmente.

Para llevar adelante dicho procedimiento, desde Postman se exportaron, tanto la colección de tests, como el ambiente de trabajo (*environment* [4]). Esto generó dos archivos que se utilizaron para configurar la ejecución automática desde Newman. Luego, desarrollamos un script en el backend del trabajo para poder correr la colección con la herramienta de *npm*.

En la figura 4.2 se muestra el script generado y la configuración del mismo en el archivo package.json del módulo de backend.

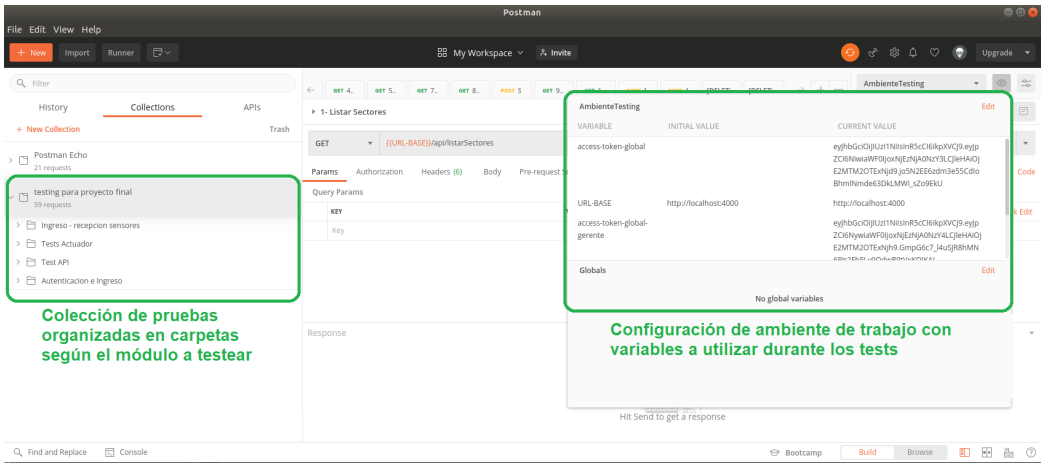

FIGURA 4.1. Herramienta Postman junto a su configuración básica.

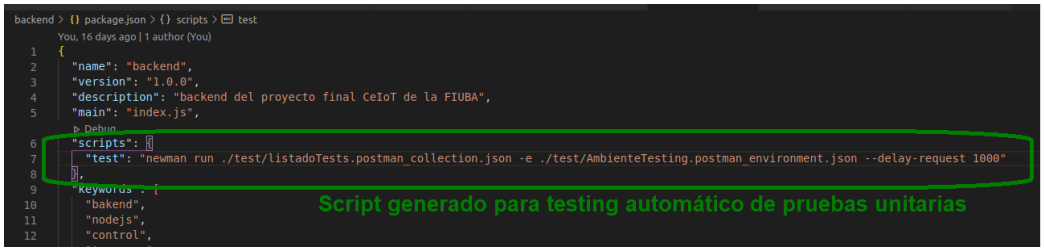

FIGURA 4.2. Configuración del script para testing automático de pruebas unitarias.

Por otra parte, en la figura 4.3 se muestra la ejecución del script desde la consola de Visual Studio Code y los resultados obtenidos.

4.1.2. Mocks implementados

Para aquellos módulos que tenían dependencias (interfaces) con otros se utilizaron *mocks*, lo que permitió testearlos, simulando el comportamiento de los dependientes, sin necesidad de tenerlos implementados.

Se desarrollaron dos *mocks* en el trabajo:

1. mockActuador: este *mock* permitió simular la operatoria del módulo actuador del sistema. El mismo se implementó como una aplicación web en Node.JS que respondía a las peticiones realizadas desde el backend del sistema.

2. mockSistemaTerceros: este *mock* permitió simular la interfaz entre el sistema desarrollado y el sistema de documentación de terceros. El mismo se implementó como una aplicación web en Node.JS,

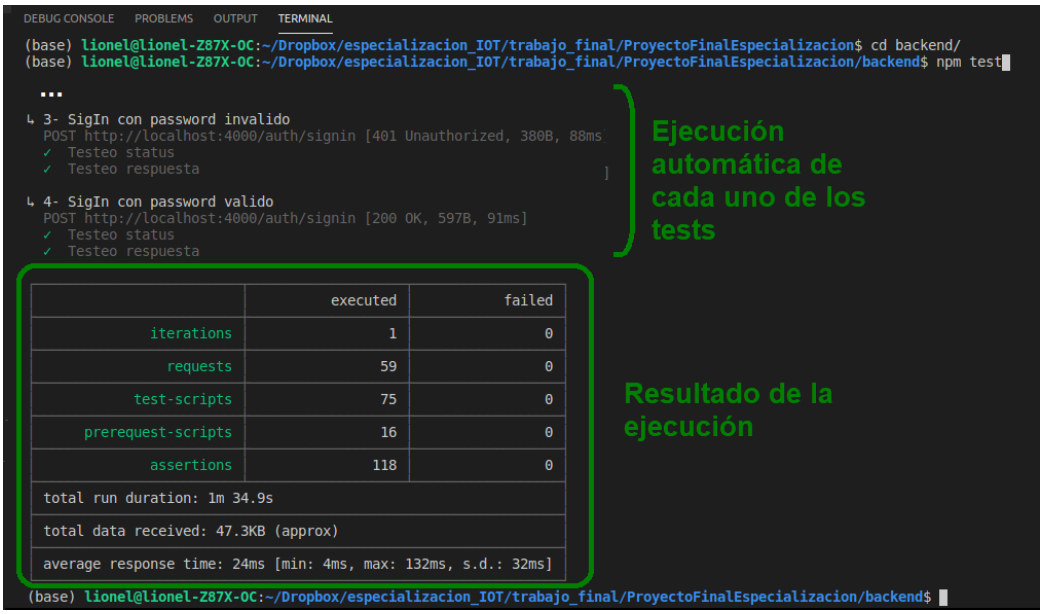

FIGURA 4.3. Ejecución y resultados de ejecución del script automático de testing.

4.2. Pruebas unitarias

En esta sección se detalla el conjunto de pruebas unitarias realizadas sobre cada uno de los módulos del sistema.

4.2.1. Testing del módulo sensor

Al momento de testear el módulo sensor, ya se contaba con el módulo del backend desarrollado, con lo cual no fue necesario implementar un *mock*.

Para proceder con las pruebas, se configuraron algunas tarjetas RFID con diferentes valores que simularon cada uno de los escenarios posibles. Se utilizó con tal propósito un programa escrito en Arduino IDE, que permitió asignar el valor a la tarjeta a través del monitor serie. Luego, se alimentó el módulo y se procedió a testear cada caso, acercando una a una las tarjetas ya configuradas para observar los resultados obtenidos.

En la tabla 4.2 se detalla el conjunto de casos de tests más relevantes implementados para probar el módulo sensor y sus resultados. [1]

En la figura 4.4 se muestra el caso de test 1 donde la tarjeta no tiene valor configurado. En la primera imagen se ve el momento en que se acerca la tarjeta y en la segunda imagen se ve el momento en que el módulo responde al usuario.

[1] Para tener un detalle completo de los test remitirse al documento de Casos de Uso y Casos de Test del sistema [3].

TABLA 4.2. Casos de test del módulo sensor.

Caso test	Escenario a testear	Resultados
1	Tarjeta RFID sin valor configurado.	El led de PinTarjNoLeida se prende durante dos segundos y se apaga.
2	Valor de tarjeta RFID recibido correctamente por el backend.	El led PinOkSistema se prende durante dos segundos y se apaga.
3	El módulo sensor no registrado en el sistema.	El led PinNoOkSistemaCodigo se prende durante dos segundos y se apaga.
4	El módulo sensor no está activo en el sistema.	El led PinNoOkSistemaCodigo se prende durante dos segundos y se apaga.

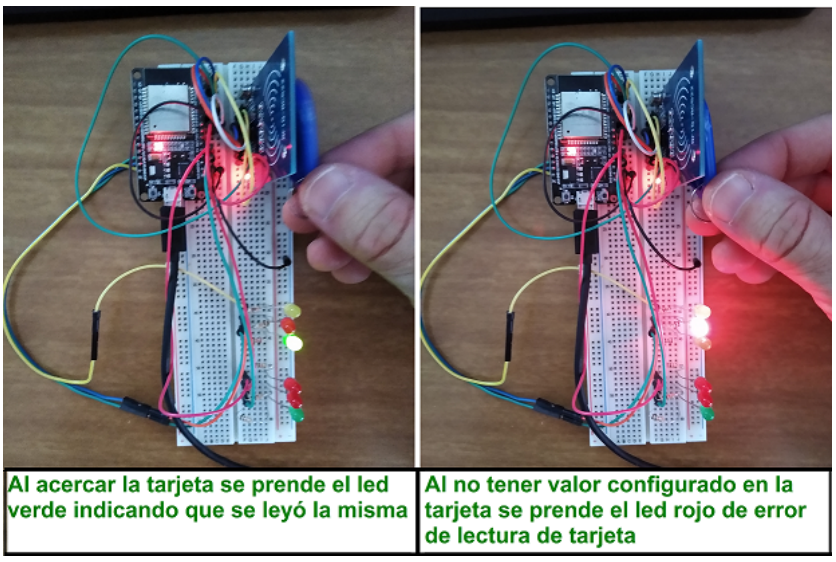

FIGURA 4.4. Procedimiento de prueba para el caso de test 1 con tarjeta sin valor configurado.

En la figura 4.5 se muestra el caso de test 2 donde la tarjeta es válida, y el sistema se comunica correctamente con el módulo del backend. En la primera imagen se ve el momento en que se acerca la tarjeta y en la segunda imagen se ve el momento en que el módulo responde al usuario.

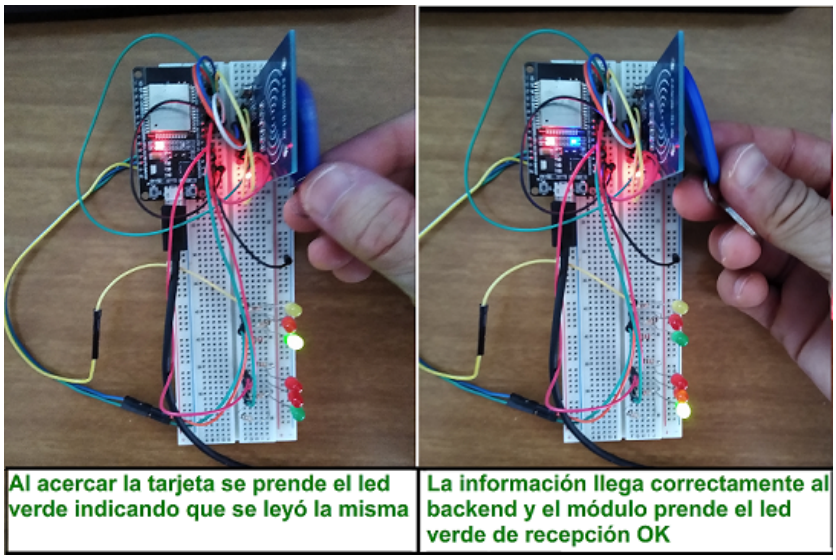

FIGURA 4.5. Procedimiento de prueba para el caso de test 2 con tarjeta con valor configurado y recibido correctamente por el backend.

4.2.2. Testing del módulo actuador

Para testear el módulo actuador se utilizó Postman, dado que el módulo expone una conexión HTTP POST. Dentro de Postman generamos una carpeta llamada Test Actuador, en la cual almacenamos todos los casos de test. En el *body* se envían los valores como un objeto JSON.

En la figura 4.6 se muestra la configuración de Postman para el testeo del módulo actuador.

La prueba implicó testear cada uno de los casos generados, ejecutándolos desde Postman. A fin de automatizar los tests, y no depender de un control manual de la respuesta obtenida luego de cada ejecución, se configuraron para cada test las condiciones esperadas o aserciones, utilizando la sección Tests de la herramienta. Algunas de las aserciones definidas fueron que el código de estado HTTP sea el esperado y/o que la respuesta contenga cierto dato o cierto objeto JSON con determinados valores. Al ejecutar el test, si los resultados eran los previstos, Postman indicaba en el apartado Test Results que los mismos eran correctos.

En la figura 4.7 se muestra un ejemplo de configuración de la sección Tests y la ejecución del caso de test.

En la tabla 4.3 se detalla el conjunto de casos de tests más relevantes implementados para probar el módulo actuador y sus resultados. Dentro del escenario a testear entre paréntesis se hace referencia al nombre del test en Postman. [2]

[2]Para tener un detalle completo de los test remitirse al documento de Casos de Uso y Casos de Test del sistema [3].

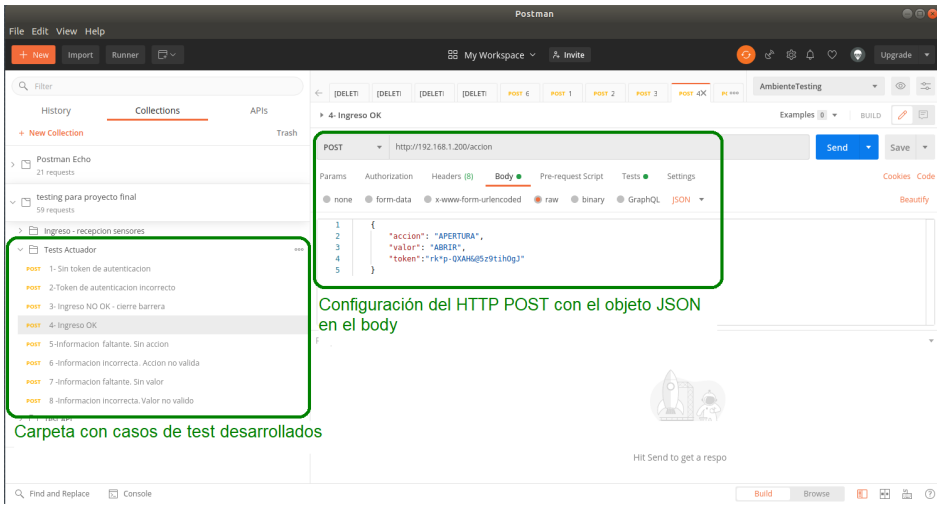

FIGURA 4.6. Configuración en Postman para el testeo del módulo actuador.

TABLA 4.3. Casos de test del módulo actuador.

Caso test	Escenario a testear	Resultados
1	Habilitar ingreso a la planta (Ingreso OK).	El led IngresoOk parpadea durante 4 segundos y se activa la cerradura.
2	Inhabilitar ingreso a la planta (Ingreso NO OK – cierre barrera).	El led IngresoNOOk se prende durante dos segundos y se apaga.
3	El pedido HTTP POST no cuenta con el campo de acción a realizar (Información faltante – Acción no válida).	El led Error parpadea dos veces.
4	El pedido HTTP POST no cuenta con token de autenticación (Sin token de autenticación).	El led Error parpadea una vez.

En la sección 4.4 se muestra el detalle de los casos de ingreso habilitado e inhabilitado junto a una figura del módulo actuador. Remitirse a dicha sección para más detalles.

4.2.3. Testing del módulo de backend

Para testear el módulo de backend se utilizó Postman, dado que el módulo expone una API Rest al frontend, una API de autenticación y un *endpoint* [5] para recibir los datos de los módulos sensores. Dentro de Postman generamos una carpeta llamada Test API para testear la API expuesta al frontend, una carpeta llamada Autenticación e ingreso para testear la API de autenticación y una carpeta llamada Ingreso – recepción sensores para testear los ingresos generados desde el módulo sensor.

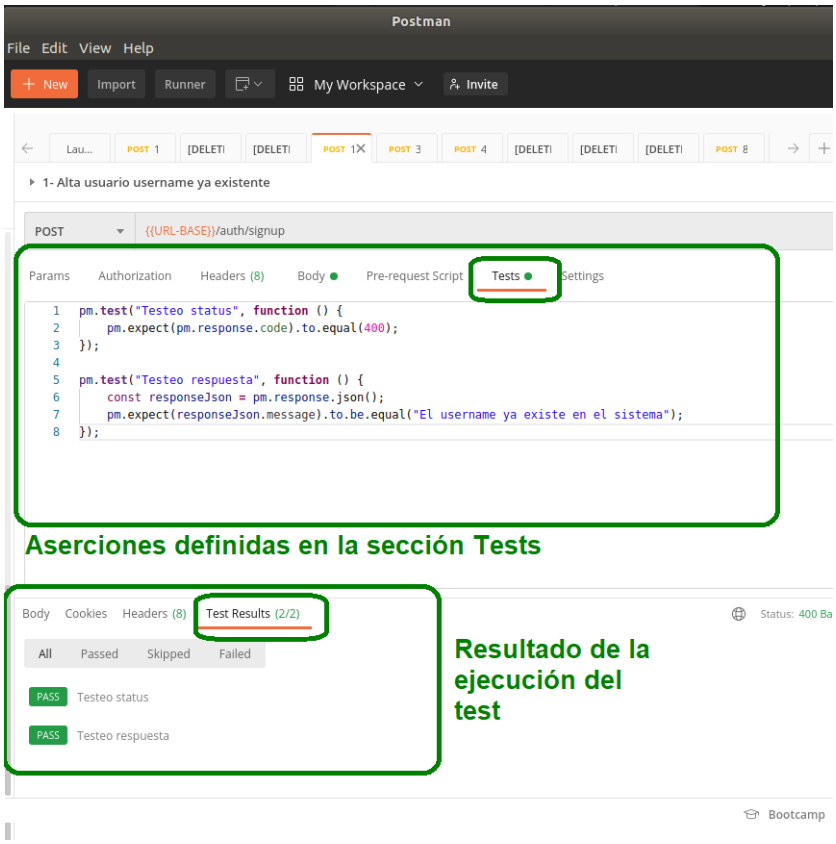

FIGURA 4.7. Configuración de sección Tests y resultados de la ejecución del test.

Dentro de la carpeta de Test API generamos cuatro subcarpetas para testear los casos sin autenticación y con autenticación para los diferentes perfiles de usuario. Dentro de la carpeta de Autenticación e ingreso, se colocaron los tests para el alta de usuario y el ingreso al sistema. Y dentro de la capeta Ingreso – recepción sensores, los tests para los ingresos generados desde el módulo sensor.

En la figura 4.8 se muestra las carpetas configuradas en Postman para el testeo de los casos de test de backend.

Para testear cada caso, se procedió a configurar en Postman la URL de cada *endpoint*, junto a los parámetros requeridos. Para los pedidos HTTP GET se configuró directamente la URL con los parámetros, y para los HTTP POST se completó el *body* con el objeto JSON, junto a los valores requeridos según cada caso.

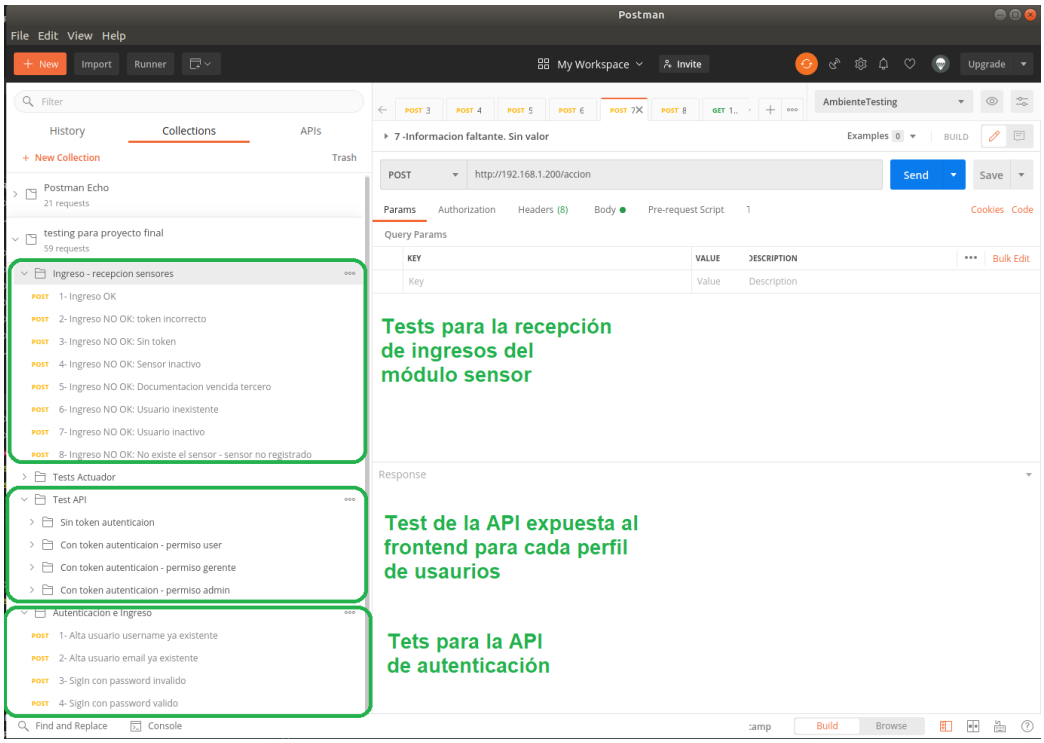

FIGURA 4.8. Configuración en Postman para el testeo del módulo de backend.

Para los casos de test con autenticación definidos dentro de la carpeta Test API, se procedió a utilizar los *environments* y las variables globales de Postman. Un *environment* permite definir ambientes de testing, dentro de los cuales podemos definir variables globales que pueden utilizarse en varios tests. Un test puede obtener un valor de la respuesta HTTP y guardarlo en una variable global, para luego ser utilizado por otro. De este modo, definimos un *environment* llamado AmbienteTesting y dentro del mismo configuramos las siguientes variables globales:

- URL-BASE: contiene la URL base del backend a partir del cual se define cada *endpoint*. Con la misma configuramos cada HTTP POST o GET, de modo que si cambia la URL de nuestro backend, con solo modificar esta variable quedan ajustados todos los *endpoints*.
- Access-token-global: utilizado para los *endpoints* que son accesibles para cualquier usuario del sistema. Permite guardar el token de acceso del rol usuario.
- Access-token-global-gerente: utilizado para los *endpoints* que son accesibles solo por el perfil de gerente de la aplicación. Permite guardar el token de acceso de rol gerente.
- Access-token-global-admin: utilizado para los *endpoints* que son accesibles solo por el perfil de administrador de la aplicación. Permite guardar el token de acceso para el rol de administrador.

Los tokens de acceso se configuran dentro de la sección header del test, con clave x-access-token y con el valor del token.

En la figura 4.9 se muestra las variables globales definidas en Postman.

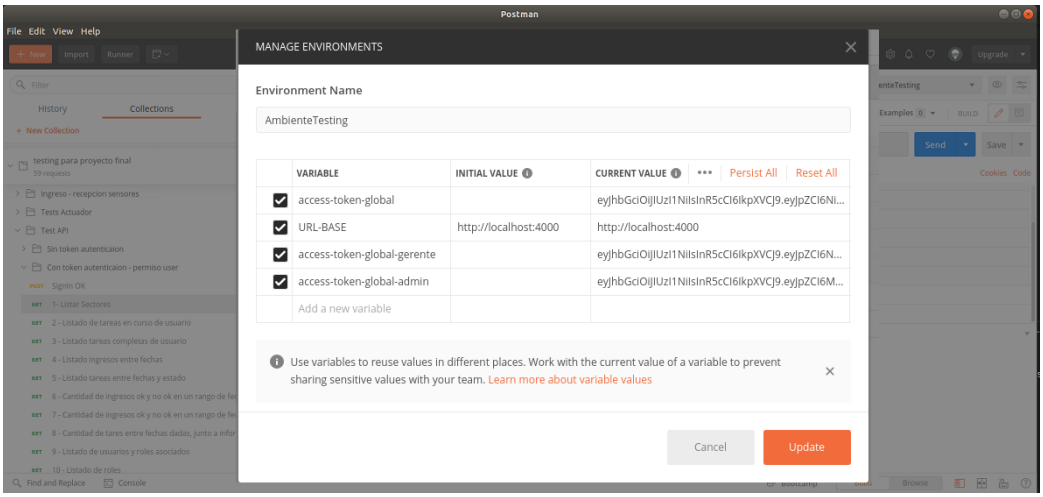

FIGURA 4.9. Variables globales definidas en Postman.

Tests de carpeta Test API

Dentro de la carpeta Test API tenemos cuatro subcarpetas:

- Sin token autenticación: testea los *endpoints* sin un token de autenticación.
- Con token autenticación – permiso user: testea los *endpoints* que son accesibles para cualquier usuario del sistema. En esta carpeta se define un test inicial llamado SigIn OK, que se utilizó para simular la autenticación de un usuario normal y completar la variable global access-token-global que fue utilizada por el resto de los tests de la carpeta.
- Con token autenticación – permiso gerente: testea los *endpoints* que son accesibles para el usuario con rol gerente. En esta carpeta se define un test inicial llamado SigIn OK, que se utilizó para simular la autenticación de un usuario gerente y completar la variable global access-token-global-gerente que fue utilizada por el resto de los tests de la carpeta.
- Con token autenticación – permiso admin: testea los *endpoints* que son accesibles para el usuario con rol administrador. En esta carpeta se define un test inicial llamado SigIn OK, que se utilizó para simular la autenticación de un usuario administrador y completar la variable global access-token-global-admin que fue utilizada por el resto de los tests de la carpeta.

En la figura 4.10 se muestran los tests de la subcarpeta Con token autenticación – permiso gerente, junto al test inicial de SignIn OK y al resto de los tests.

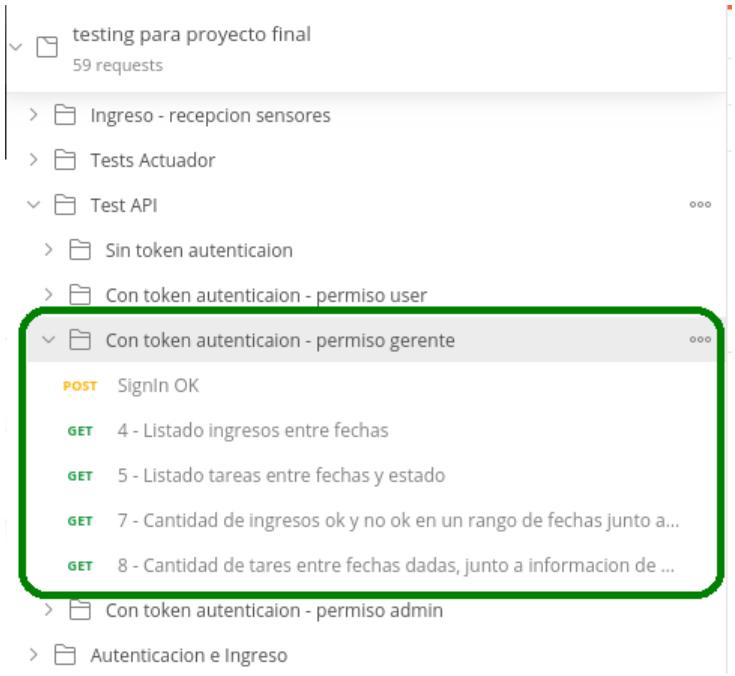

FIGURA 4.10. Tests de la subcarpeta Con token autenticación – permiso gerente.

Detalle de tests y resultados

En la tabla 4.4 se detalla el conjunto de casos de tests más relevantes implementados para probar la recepción de los datos de los módulos sensores y sus resultados. En la columna escenario a testear se hace referencia, entre paréntesis, al número del test en Postman. [3]

TABLA 4.4. Casos de test del módulo backend. Testing de recepción de los datos de los módulos sensores.

Subcarpeta Postman	**Escenario a testear**	**Resultados**
Ingreso – recepción sensores.	Ingreso con valor de tarjeta registrado en el sistema y módulo sensor activo (1).	Se confirma recepción con HTTP código 200.
	El módulo sensor no está registrado en el sistema (8).	Se rechaza el ingreso con mensaje de error de sensor inexistente.
	El módulo sensor no está activo en el sistema (4).	Se rechaza el ingreso con mensaje de error de sensor inactivo.

En la tabla 4.5 se detalla el conjunto de casos de tests más relevantes implementados para probar la API Rest expuesta al frontend y sus resultados. En la columna

[3]Para tener un detalle completo de los test remitirse al documento de Casos de Uso y Casos de Test del sistema [3].

escenario a testear se hace referencia, entre paréntesis, al número del test en Postman. [4]

TABLA 4.5. Casos de test del módulo backend. Testing de la API Rest expuesta al frontend.

Subcarpeta Postman	**Escenario a testear**	**Resultados**
Test API/Sin token autenticación.	Listar sectores de la empresa (1).	Se acepta el pedido y se devuelve el listado de sectores de la empresa.
	Resto de los *endpoints*.	Se rechaza el pedido por falta de token de autenticación con un HTTP 403.
Test API/Con token autenticación – permiso user.	Listado de tareas en curso del usuario (2).	Se acepta el pedido y se devuelve el listado de tareas en curso del usuario.
	Listado de tareas completas del usuario (3).	Se acepta el pedido y se devuelve el listado de tareas completas del usuario.
	Cantidad ingresos a la empresa por día (7).	Se rechaza el pedido por no tener rol gerente o administrador con un HTTP 403.
Test API/Con token autenticación – permiso gerente.	Cantidad ingresos a la empresa por día (7).	Se acepta el pedido y se devuelve el listado de ingresos OK y no OK para las fechas indicadas.
	Listado detalle de tareas en curso entre rango de fechas (5).	Se acepta el pedido y se devuelve el listado de tareas y sub-tareas para las fechas indicadas.
Test API/Con token autenticación – permiso admin.	Listado de usuarios y roles (9).	Se acepta el pedido y se devuelve el listado de usuarios y roles de cada usuario.
	Cambiar estado de rol de usuario (12).	Se acepta el pedido y ajusta el estado del rol para el usuario indicado.

[4] Para tener un detalle completo de los test remitirse al documento de Casos de Uso y Casos de Test del sistema [3].

En la tabla 4.6 se detalla el conjunto de casos de tests más relevantes implementados para probar la API de autenticación y sus resultados. En la columna escenario a testear se hace referencia, entre paréntesis, al número del test en Postman.

TABLA 4.6. Casos de test del módulo backend. Testing de la API de autenticación.

Subcarpeta Postman	**Escenario a testear**	**Resultados**
Autenticación e ingreso.	Alta de usuario con username, email y password correcto (5).	Se confirma el alta del usuario con HTTP código 200.
	Alta de usuario con username o email repetido (1 y2).	Se rechaza el alta. Respuesta HTTP con código 400 y mensaje de error.
	Inicio de sesión con username o password incorrecto (3 y 4).	Se rechaza el inicio de sesión. Respuesta HTTP con código 401 y mensaje de error.

4.3. Pruebas de sistema

En esta sección se detalla el conjunto de pruebas integrales realizadas sobre el sistema con todos sus módulos. Este test fue ejecutado por el desarrollador en el ambiente de pruebas, sin el cliente.

Una vez implementados todos los módulos del sistema, fueron realizadas las siguientes tareas:

1. Se crearon usuarios con cada uno de los roles en la base de datos.
2. Se alimentó el módulo sensor y el módulo actuador.
3. Se corrió el módulo de backend, el *mock* del sistema de terceros y el módulo de frontend. Con el propósito de automatizar este paso, se desarrolló un script para ejecutar y levantar los módulos antes mencionados, mediante el uso de contenedores *Docker* y de la herramienta *docker-compose*.

Con los módulos en ejecución, se procedió a *loguearse* en el sistema. En primer lugar, con un rol de usuario normal, luego con el rol de usuario gerente, posteriormente con el rol vigilante y por último con el rol administrador.

En la tabla 4.7 se detalla el conjunto de casos de tests más relevantes implementados para probar el rol de usuario normal del sistema y los resultados obtenidos.[5]

En la tabla 4.8 se detalla el conjunto de casos de tests más relevantes implementados para probar el rol de usuario administrador del sistema y los resultados obtenidos.

[5]Para tener un detalle completo de los test remitirse al documento de Casos de Uso y Casos de Test del sistema [3].

TABLA 4.7. Listado de pruebas de sistemas realizadas para el rol de usuario normal.

Pantalla del sistema testeada/Acción	**Resultados**
Mis tareas en curso/Visualizar tareas.	El sistema muestra el listado de tareas en curso del usuario o indica que no tiene tareas en curso.
Mis tareas en curso/-Cerrar tarea.	El sistema marca como completa la tarea del usuario y muestra una alerta indicando que se cerró la tarea correctamente.
Mis tareas en curso/Ajustar observación en tarea.	El sistema ajusta la observación en la tarea y muestra una alerta indicando que se actualizó la observación correctamente.
Mi historial tareas/Visualizar historial tareas de usuario.	El sistema muestra el listado de tareas cerradas del usuario o indica que no tiene tareas cerradas.
Mi perfil/Visualizar perfil.	El sistema muestra la información del usuario, incluyendo su *username*, email, sector y roles asignados.

TABLA 4.8. Listado de pruebas de sistemas realizadas para el rol de usuario administrador.

Pantalla del sistema testeada/Acción	**Resultados**
Gestión usuarios/Modificar estado de rol de usuario.	El sistema modifica el estado del rol del usuario en la base de datos y muestra una alerta indicando que el cambio se realizó correctamente.

En la tabla 4.9 se detalla el conjunto de casos de tests más relevantes implementados para probar el rol de usuario vigilante del sistema y los resultados obtenidos.

En la tabla 4.10 se detalla el conjunto de casos de tests más relevantes implementados para probar el rol de usuario gerente del sistema y los resultados obtenidos.

TABLA 4.9. Listado de pruebas de sistemas realizadas para el rol de usuario vigilante.

Pantalla del sistema testeada/Acción	**Resultados**
Pantalla principal/-Visualizar ingreso en portería.	Cuando un tercero pretende ingresar en planta, el sistema muestra en su pantalla principal una alerta, indicando el intento de acceso y si el mismo se permitió o se rechazó, junto a un color particular para diferenciar ambos casos. La alerta se muestra por 6 segundos.

TABLA 4.10. Listado de pruebas de sistemas realizadas para el rol de usuario gerente.

Pantalla del sistema testeada/Acción	**Resultados**
Estadísticas ingresos/-Visualizar ingresos del mes.	El sistema muestra un gráfico de barras con la cantidad de ingresos de cada día del mes especificado (ingresos ok y no ok) y un gráfico de torta con el total de ingresos ok y no ok del mes.
Estadísticas tareas/Visualizar estadísticas de tareas.	El sistema muestra un gráfico de torta con la cantidad de tareas en curso, un gráfico de barras con la cantidad de tareas cerradas por cada mes del año y un gráfico de torta con el total de tareas completas del año.
Tareas en curso/Visualizar tareas en curso.	El sistema muestra el listado de tareas en curso, junto a información de antigüedad, estado, fecha de inicio y detalle de las sub-tarea de cada sector.
Ingresos del día/Visualizar ingresos del día.	El sistema muestra el listado de los ingresos, junto a información de horario, si el ingreso fue permitido o no y el detalle de la habilitación o rechazo junto al nombre del tercero.

4.4. Pruebas de aceptación

En esta sección se detallan dos de las pruebas de aceptación realizadas en conjunto con el cliente. Estos tests fueron llevados a cabo por el desarrollador del sistema, en el ambiente de pruebas, para verificar el funcionamiento correcto del sistema implementado.

Para preparar dicho ambiente se procedió de igual modo que el descripto en la sección anterior para las pruebas de sistema.

4.4.1. Descripción y detalles de prueba de ingreso habilitado

En esta sección se detalla la prueba de ingreso de un tercero a la empresa que cuenta con la documentación requerida en regla.

Para realizar la prueba se preparó el sistema y se lo dejó operativo. Se utilizó la tarjeta RFID de un tercero que estaba habilitado y un usuario con rol de vigilancia generado previamente.

A continuación, se detallan los pasos realizados para el caso descripto:

1. Se ingresó al sistema con el usuario Portería y se accedió a la página principal.
2. Se tomó la tarjeta RFID del personal de tercero habilitado en el sistema y se la acercó al módulo sensor. Como respuesta a esta interacción, sucedieron las siguientes acciones:
   a) El módulo sensor prendió el led verde de lectura de la tarjeta y luego prendió el led verde de comunicación correcta con el sistema de control.
   b) En la pantalla del usuario de portería se visualizó la alerta de ingreso del tercero durante 6 segundos. A continuación, se corroboró que el contador de ingresos correctos se había incrementado en la pantalla.
   c) El módulo actuador prendió de forma intermitente el led verde de ingreso habilitado durante 4 segundos y se cerró el pestillo de la cerradura electrónica. Pasado ese tiempo el pestillo se abrió nuevamente.

En la figura 4.11 se muestra el paso 2 de la prueba con el acercamiento de la tarjeta RFID al módulo sensor y la respuesta de dicho módulo.

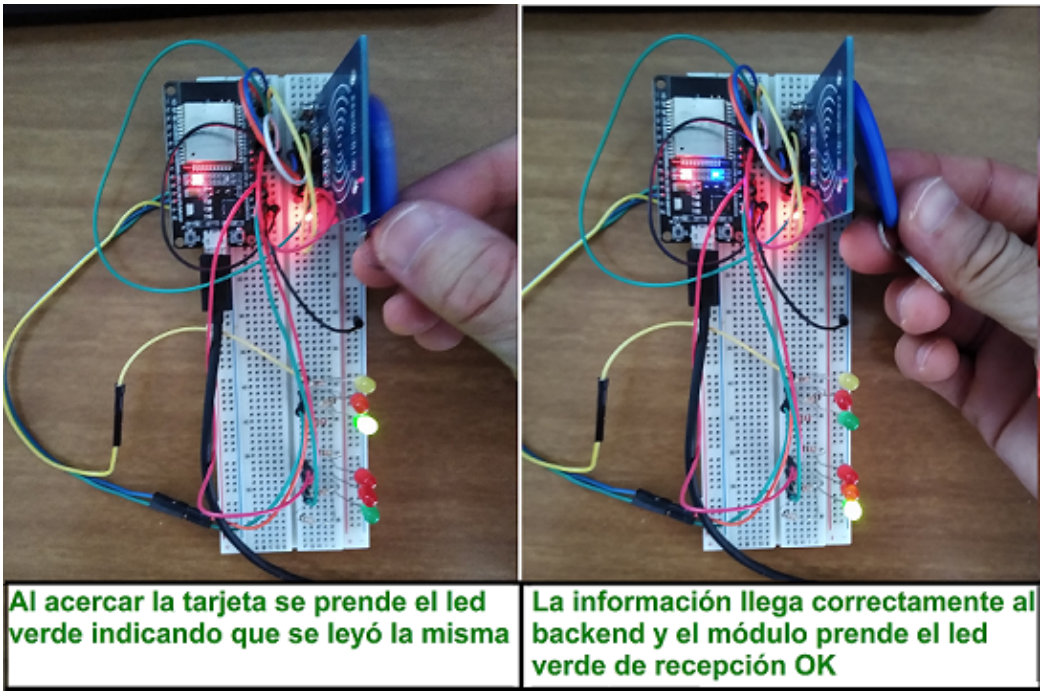

FIGURA 4.11. Accionamiento de módulo sensor con tarjeta RFID y respuesta del módulo.

En la figura 4.12 se muestra el paso 2 b) de prueba con la información visualizada por la vigilancia.

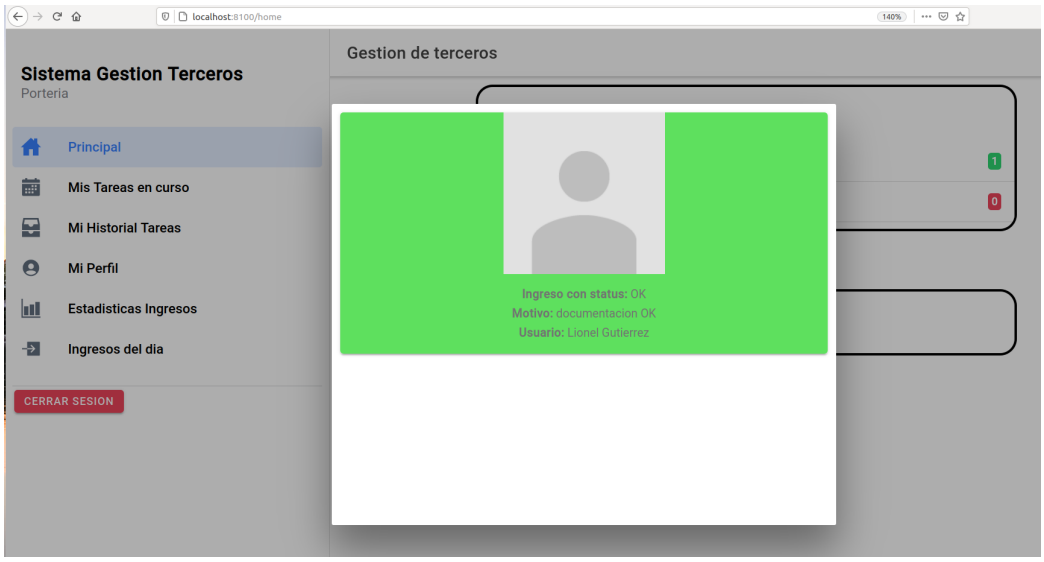

FIGURA 4.12. Visualización de la alerta recibida en pantalla.

En la figura 4.13 se muestra el paso 2 c) de la prueba, donde se ve el módulo actuador con el led prendido y el pestillo de la cerradura cerrada.

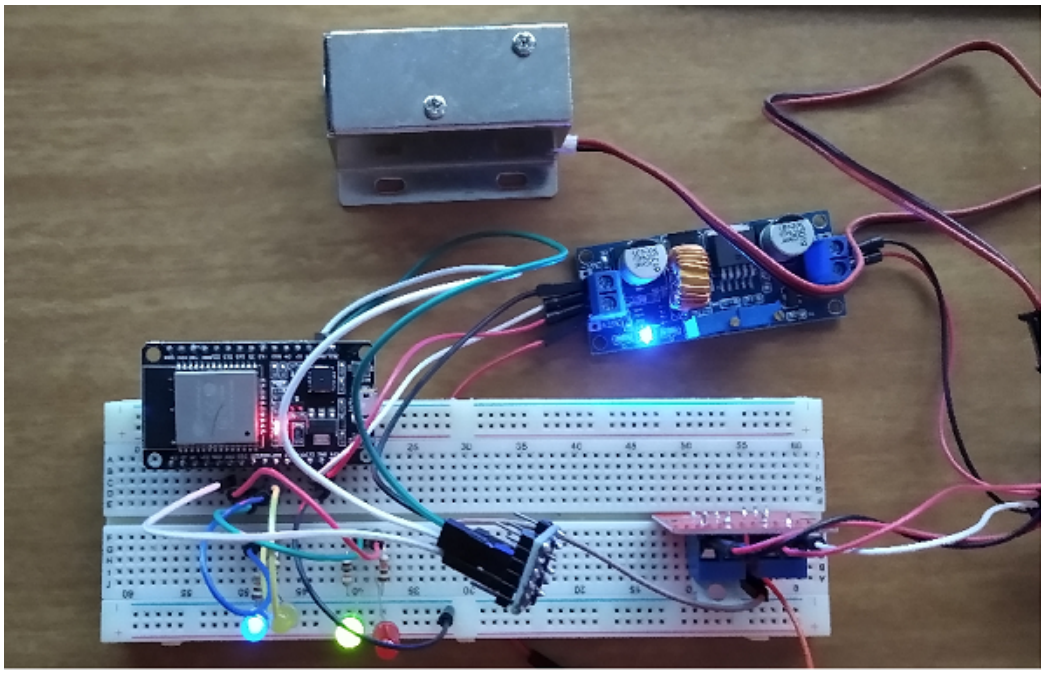

FIGURA 4.13. Accionamiento de módulo actuador y respuesta del módulo.

4.4.2. Descripción y detalles de prueba de ingreso inhabilitado

En esta sección se detalla la prueba de ingreso de un tercero a la empresa que no cuenta con la documentación requerida en regla.

Para realizar la prueba se preparó el sistema y se lo dejó operativo. Se utilizó la tarjeta RFID de un tercero que tenía problemas con su documentación y un usuario con rol de vigilancia generado previamente.

A continuación, se detallan los pasos realizados para el caso descripto:

1. Se ingresó al sistema con el usuario Portería y se accedió a la página principal.
2. Se tomó la tarjeta RFID del personal de tercero inhabilitado en el sistema y se acercó la misma al módulo sensor. Como respuesta a esta interacción, sucedieron las siguientes acciones:
   a) El módulo sensor prendió el led verde de lectura de la tarjeta y luego prendió el led verde de comunicación correcta con el sistema de control.
   b) En la pantalla del usuario de portería se visualizó la alerta de ingreso inhabilitado del tercero durante 6 segundos. A continuación, se corroboró que el contador de intento de ingresos incorrectos se había incrementado en la pantalla.
   c) El módulo actuador prendió durante 2 segundos el led rojo de ingreso inhabilitado. El pestillo de la cerradura electrónica se mantuvo abierto todo el tiempo.
   d) Se envió un email a los usuarios de los sectores encargados de controlar la documentación del tercero, con el detalle del intento de ingreso.
   e) Se generó una tarea de control de documentación para cada uno de los sectores de la empresa responsables del proceso.
3. Se ingresó al sistema con un usuario de sector HESA (Seguridad y Medio Ambiente) para corroborar la generación de la tarea de control. Se accedió a la aplicación Mis tareas en curso (preguntar como pongo estas cosas) y se visualizó la nueva tarea asignada al usuario.

En la figura 4.14 se muestra el paso 2 b) de la prueba con la información visualizada por la vigilancia.

En la figura 4.15 se muestra el paso 2 c) de la prueba, donde se ve el módulo actuador con el led rojo prendido y el pestillo de la cerradura abierta.

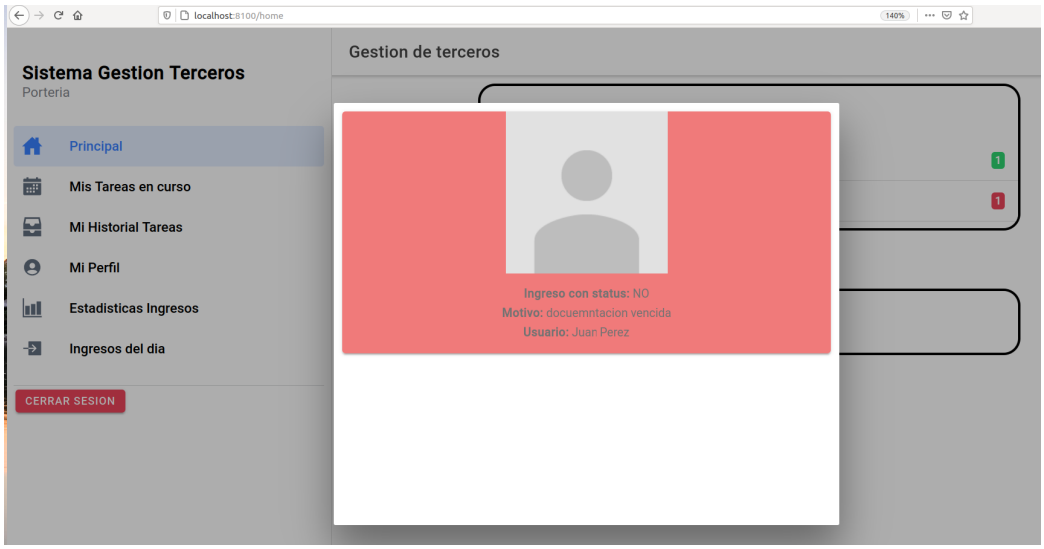

FIGURA 4.14. Visualización de la alerta recibida en pantalla.

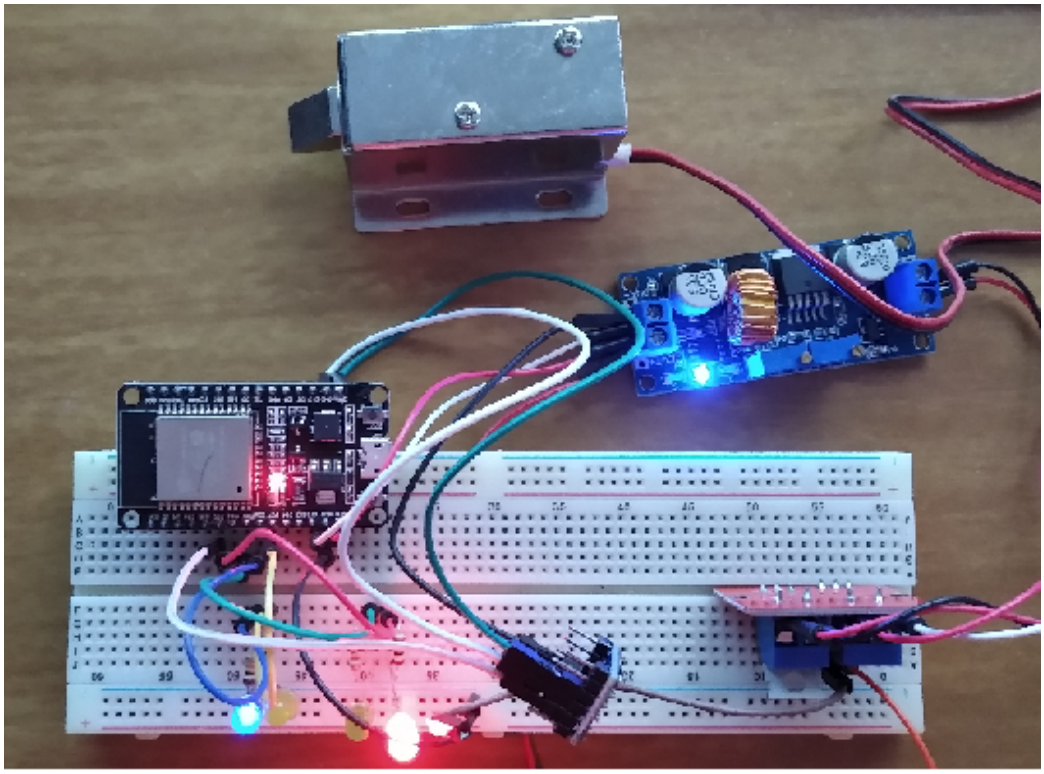

FIGURA 4.15. Accionamiento de módulo actuador y respuesta del módulo.

En la figura 4.16 se ve el paso 2 d) de la prueba, con el email recibido por el usuario con la alerta del intento de ingreso con documentación vencida.

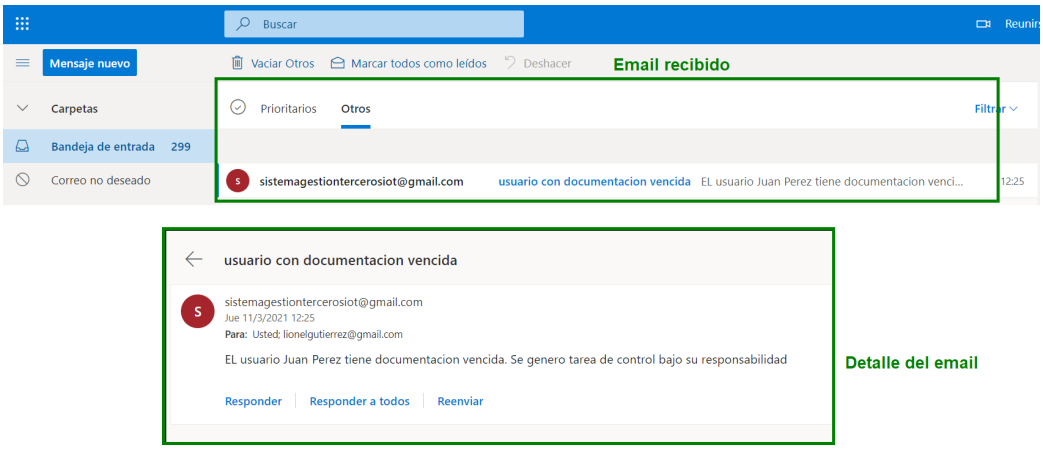

FIGURA 4.16. Email recibido por el usuario con la alerta de intento de ingreso con documentación vencida.

En la figura 4.17 se ve el paso 3 de la prueba, donde el usuario ve la tarea generada por parte del usuario.

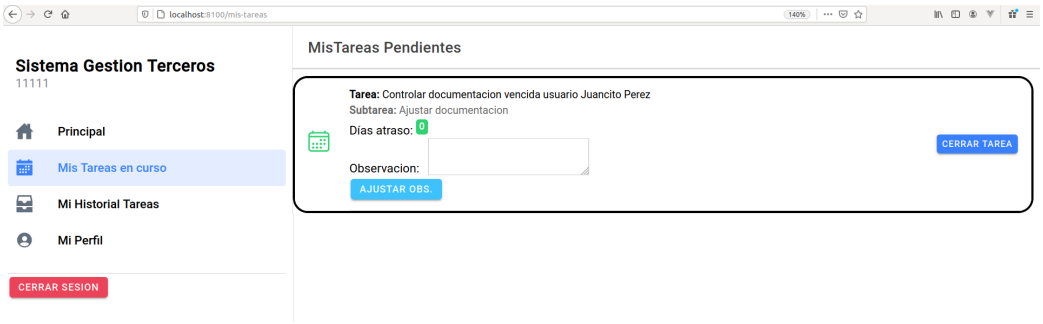

FIGURA 4.17. Pantalla de usuario se sector HESA con la tarea de control generada.

4.5. Comparativa con otras soluciones del mercado

A modo de conclusión de los ensayos realizados, se hace una comparativa contra soluciones similares que existen en el mercado para el control de ingreso. De acuerdo con el análisis comparativo realizado, se destaca el diferencial de este sistema al permitir comunicarse con otros sistemas de terceros y lograr una gestión integral de accesos, con capacidad de generar alertas y tareas de control sobre el proceso.

En la tabla 4.11 se muestra la comparación entre el sistema implementado y soluciones similares del mercado, en la que se puede apreciar el diferencial del sistema desarrollado.

TABLA 4.11. Comparación contra otras soluciones similares del mercado.

Característica	**Sistema**	**Pronext KY800**	**Samsung H505**
Log de ingresos	Si	Si	No
Interfaz a sistemas externos	Si	No	No
Gestión integral de accesos	Si	No	No
Máximo usuarios	Indefinido	500	30
Conectividad/Protocolos	Wi-Fi	Bluetooth	No
Doble factor autenticación	No	No	Si
Tarjetas RFID	Si	Si	Si
Alerta de acceso	Si	Si	Si
Acceso con huella	No	Si	No

Capítulo 5

Conclusiones

5.1. Resultados obtenidos

Se logró cumplir el alcance y objetivo del proyecto. En primer lugar, se implementó el sistema de control solicitado que incluyó la puesta en funcionamiento de un módulo sensor, un módulo actuador y una aplicación web para la gestión de tareas de control y de alertas. En segundo lugar, se sentaron las bases para el agregado de futuros casos de uso, para lo cual se hizo un diseño modular. Esto permitirá al sistema el sensado de datos de nuevos procesos de planta, según el requerimiento 1.1, especificado en la sección 2.5.

Adicionalmente, se incorporó al sistema un módulo para gestionar la autenticación y autorización de los usuarios y el acceso a la API Rest del backend del sistema de modo seguro. Esto posibilitó cumplir con el requerimiento 1.6, que fue agregado al trabajo durante su desarrollo. Este requerimiento permite independizar al sistema desarrollado del sistema de autenticación de la empresa y lograr una mayor portabilidad, lo que le da la posibilidad a futuro de expandir el mismo a otras empresas.

Finalmente, el trabajo cumplió con el requerimiento de lograr una gestión efectiva de los terceros, al implementar un proceso de control estricto de los requisitos de ingreso. Si bien existen otros sistemas de control de ingreso en el mercado, se logró agregar valor mediante la comunicación del sistema en cuestión con el sistema de control de documentación de terceros sumado a los procesos de alerta y gestión implementados. El trabajo realizado habilita una gestión proactiva, rápida y ordenada de los terceros, de forma de actuar inmediatamente ante problemas de ingresos. Las ventajas obtenidas incluyen:

- Reducir tiempo para la gestión.
- Evitar atrasos en ingresos por falta de ajustes en la documentación.
- Evitar problemas legales ante incidentes del personal externo.
- Evitar el uso de papel y herramientas des-centralizadas para el control y gestión de los terceros por parte de cada sector de la empresa.

Si bien todavía el sistema no se implementó en operativo, con las pruebas realizadas y analizando los ingresos de personal en los últimos dos años, se prevé evitar un 5 % de ingresos incorrectos o con problemas, y reducir los tiempos ante inconvenientes con la documentación en unas 10 horas hombre/mes.

Cabe destacar la importancia de los conocimientos obtenidos a lo largo de la carrera. En primer lugar, fueron muy importantes los aportes de la asignatura de gestión de proyectos. Una buena gestión de proyectos fue fundamental, tanto para lograr una planificación clara que actúe como guía a lo largo de todo el desarrollo, como para minimizar riesgos. En lo particular del trabajo, luego del comienzo del desarrollo se solicitó agregar un nuevo requerimiento al mismo. Dado que se contaba con un diagrama de Gantt [6] detallado y el avance real al momento de la solicitud, se pudo determinar que el nuevo requisito podía cumplirse en tiempo y forma, sin penalizar la fecha de finalización del proyecto. Sin una gestión de proyectos clara, es muy probable que este nuevo requerimiento haya sido rechazado.

Adicionalmente, los conocimientos en desarrollo de aplicaciones multiplataforma permitieron plantear una solución que sea *web responsive* y que a futuro se puede implementar en un entorno mobile con mínimo esfuerzo. También se abre la posibilidad de implementar el sistema en la nube.

5.2. Trabajo futuro

Para la continuidad y mejora de este trabajo se plantean dos líneas de acción.

Como primera línea de acción, referida al trabajo desarrollado, se incluyen:

- Realizar mejoras en seguridad:
  - Agregar comunicación HTTPS entre los diferentes módulos del sistema. De este modo, se asegura que la información viaje encriptada y se evitan ataques del tipo *man in the middle*. Esto permitirá migrar el sistema a la nube, donde las comunicaciones viajan a través de Internet y no son seguras.
  - Agregar un segundo factor de autenticación al sistema. Con el objetivo de evitar que ante pérdidas o robos de la tarjeta RFID de ingreso o duplicación de la misma un atacante pueda ingresar a planta, se plantea la posibilidad de agregar un teclado matricial de forma de requerir además de la tarjeta una clave numérica durante el proceso de ingreso.
- Automatizar tareas de configuración: se analiza implementar una aplicación para poder configurar las acciones del sistema ante las diferentes variantes de entradas (ingreso correcto, ingreso de usuario inactivo, ingreso con documentación vencida). Actualmente esta información se guarda y administra en una base de datos no relacional, por parte del personal de sistemas, que permite definir para cada tipo y valor de entrada un conjunto de acciones de salidas (tareas de control, alertas, emails). Se evalúa desarrollar una aplicación para que el usuario pueda configurar estas salidas y generar diferentes tipos de acción, independizándose del área de sistemas.
- Realizar una prueba de implementación en la nube: utilizar la nube de Azure para probar y asegurar el escalamiento de la solución. Esto permitirá incluir nuevas locaciones o plantas industriales al trabajo, ya sea dentro de la empresa actual o para ser implementado en nuevas empresas.

Como segunda línea de acción, en el marco del proyecto integral de gestión de alertas y procesos, el objetivo es incorporar nuevos procesos y casos de uso al sistema. De hecho, ya fue solicitado un primer caso de uso por parte del laboratorio de metrología de la empresa. El mismo implica el control de temperatura y humedad de dicho laboratorio, para asegurar que ambas variables se encuentren dentro de los límites requeridos y generar alertas en caso de desvíos para poder actuar en consecuencia, manual o automáticamente.

Bibliografía

[1] Lionel Gutiérrez. *Master Test Plan TP final CeIoT Lionel Gutiérrez*. https://docs.google.com/document/d/1-KEsjv7V-AbL9H0XLFSUNzzbdWEjcfdD8p6gnV1Ut7A/edit?usp=sharing. Oct. de 2020. (Visitado 12-03-2021).

[2] C.B. Jurado. *Diseño Ágil con TDD*. https://docs.google.com/document/d/1-KEsjv7V-AbL9H0XLFSUNzzbdWEjcfdD8p6gnV1Ut7A/edit?usp=sharing. Ene. de 2020. (Visitado 12-03-2021).

[3] Lionel Gutiérrez. *Casos de uso y de test del TP final CeIoT Lionel Gutiérrez*. https://docs.google.com/document/d/1RDlGW6ZQfH5MMlm_lq8_HcNIaUIL8euAkz1Hp3-zlKI/edit?usp=sharing. Feb. de 2021. (Visitado 12-03-2021).

[4] Postman Learning Center. *Managing environments in Postman*. https://learning.postman.com/docs/sending-requests/managing-environments/. Ene. de 2021. (Visitado 12-03-2021).

[5] Smartbear. *What is an API Endpoint?* https://smartbear.com/learn/performance-monitoring/api-endpoints. Nov. de 2018. (Visitado 12-03-2021).

[6] Wikipedia. *Diagrama de Gantt*. https://es.wikipedia.org/wiki/Diagrama_de_Gantt. Oct. de 2020. (Visitado 07-03-2021).