Prototipo automatizado para la dispensación controlada de medicamentos con verificación electrónica de prescripciones mediante códigos QR

Blanca Patricia Quintero Castillo

Asesor

Juan Alejandro Chica García

Universidad Nacional Abierta y a Distancia

Escuela de Ciencias Básicas Tecnología e Ingeniería - ECBTI

Ingeniería Electrónica

2025

Dedicatoria

Dedico este trabajo de grado con profundo amor y gratitud a dos mujeres que han sido pilares fundamentales en mi vida:

A Blanca Ríos, a quien tengo la dicha de llamar mamá, por haberme criado con dedicación, esfuerzo y un amor incondicional que ha marcado mi camino.

A Patricia Castillo, por estar a mi lado durante mi etapa universitaria, brindándome su apoyo incondicional, sus palabras de ánimo y su compañía constante.

A ambas les agradezco por estar, por creer en mí y por ayudarme a ser la persona y la profesional que hoy soy. Este logro también les pertenece.

Agradecimientos

Quiero expresar mi profundo agradecimiento a todas las personas que, de una u otra manera, contribuyeron a que este proyecto y mi formación académica fueran posibles.

A mi familia, por su apoyo constante y amor incondicional.

A mis docentes y compañeros, por compartir sus conocimientos y experiencias que enriquecieron mi aprendizaje.

Gracias a todos por acompañarme en este camino y ser parte de este logro.

Resumen

Este trabajo presenta el diseño, implementación y validación de un prototipo de dispensador automatizado para la entrega controlada de medicamentos en farmacias, que integra verificación electrónica de prescripciones mediante códigos QR. El sistema combina una API backend para registro y gestión de recetas (Flask + MongoDB Atlas), una API intermedia de validación (Node.js/Express) y un módulo físico de dispensación coordinado por una Raspberry Pi, ESP32 y Arduino.

El objetivo es disminuir errores de dispensación, prevenir el uso indebido de fármacos controlados y garantizar trazabilidad en tiempo real. El prototipo fue desarrollado siguiendo la metodología CDIO y probado en escenarios controlados; se validó la generación y verificación segura de tokens QR, la comunicación entre microcontroladores (UART/UDP) y la secuencia automatizada de dosificación, tapado y registro de entregas.

Los resultados muestran un prototipo funcional que mejora la seguridad y la trazabilidad del proceso de dispensación y constituye una base para su escalamiento y evaluación clínica posterior.

Palabras claves: Automatización, Dispensación, Prescripción, QR, IoT.

Abstract

This work presents the design, implementation, and validation of a prototype automated medication dispenser for pharmacies that integrates electronic prescription verification via QR codes. The system combines a backend API for prescription management (Flask + MongoDB Atlas), an intermediary validation API (Node.js/Express), and a physical dispensing module coordinated by a Raspberry Pi, ESP32, and Arduino.

The objective is to reduce dispensing errors, prevent misuse of controlled substances, and ensure real-time traceability. The prototype was developed following the CDIO methodology and tested in controlled scenarios; validation covered secure token generation and QR verification, inter-microcontroller communication (UART/UDP), and the automated sequence for dosing, capping, and transaction logging.

Results demonstrate a functional prototype that enhances dispensing safety and traceability and provides a foundation for further scaling and clinical evaluation.

Keywords: Automation, Dispensing, Prescription, QR, IoT.

Tabla de Contenido

Lista de Tablas

Lista de Figuras

Lista de Apéndices

Introducción

La dispensación manual de medicamentos en farmacias y centros de salud continúa siendo un proceso crítico que requiere precisión, trazabilidad y control. No obstante, en muchos establecimientos aún se realizan procedimientos manuales que incrementan el riesgo de errores en la entrega, demoras en la atención y ausencia de registros automáticos, especialmente en entornos con alta demanda.

Ante esta problemática, surge la necesidad de implementar soluciones tecnológicas que optimicen la entrega de medicamentos controlados, garantizando exactitud, seguridad y eficiencia. La automatización de estos procesos no solo mejora la atención al usuario, sino que también reduce el margen de error humano y fortalece la trazabilidad dentro del sistema farmacéutico.

En respuesta a esta necesidad, el presente proyecto desarrolla un prototipo automatizado para la dispensación de medicamentos controlados, basado en tecnologías de electrónica, control y robótica. El sistema integra una base giratoria controlada por motores paso a paso, un mecanismo de dosificación con sensores infrarrojos, un brazo robótico de tres grados de libertad para la manipulación de frascos, y un sistema de tapado automático gestionado por una Raspberry Pi. La arquitectura se complementa con una comunicación distribuida mediante enlaces seriales (Raspberry Pi – Arduino) y UDP (Raspberry Pi – ESP32). A nivel de software, el proyecto implementa dos componentes esenciales:

Una API de gestión de prescripciones médicas, desarrollada en Flask (Python) y conectada a una base de datos en la nube (MongoDB Atlas), que permite registrar, validar y firmar recetas digitales, generando un token único por paciente con información del medicamento y dosis.

Una API intermedia de validación, construida en Node.js, que recibe solicitudes desde la Raspberry Pi, verifica la validez del token y consulta la base de datos para autorizar la dispensación. Esta capa actúa como un puente seguro entre el sistema físico y la información médica, garantizando la autenticidad de cada transacción.

El desarrollo se llevó a cabo bajo la metodología CDIO (Concebir, Diseñar, Implementar y Operar), asegurando un proceso ordenado desde la identificación del problema hasta la validación del prototipo. Se realizaron pruebas unitarias e integradas para garantizar la comunicación efectiva entre los módulos de software y hardware.

En conjunto, el sistema propuesto representa una contribución innovadora a la automatización farmacéutica, al ofrecer una alternativa segura y trazable que reduce errores humanos, mejora la eficiencia del proceso de dispensación y fortalece la confianza en la distribución de medicamentos controlados.

Planteamiento del Problema

La dispensación de medicamentos controlados continúa siendo un desafío crítico en el sistema de salud, especialmente en el ámbito farmacéutico, donde se requiere un estricto control sobre la cantidad, tipo y destino de los fármacos entregados. Los medicamentos con alto potencial de abuso o dependencia, como analgésicos opioides y ansiolíticos, demandan mecanismos de validación y trazabilidad que garanticen su uso adecuado. Sin embargo, en la mayoría de las farmacias este proceso sigue siendo manual o semiautomatizado, lo que incrementa la posibilidad de errores humanos y de uso indebido.

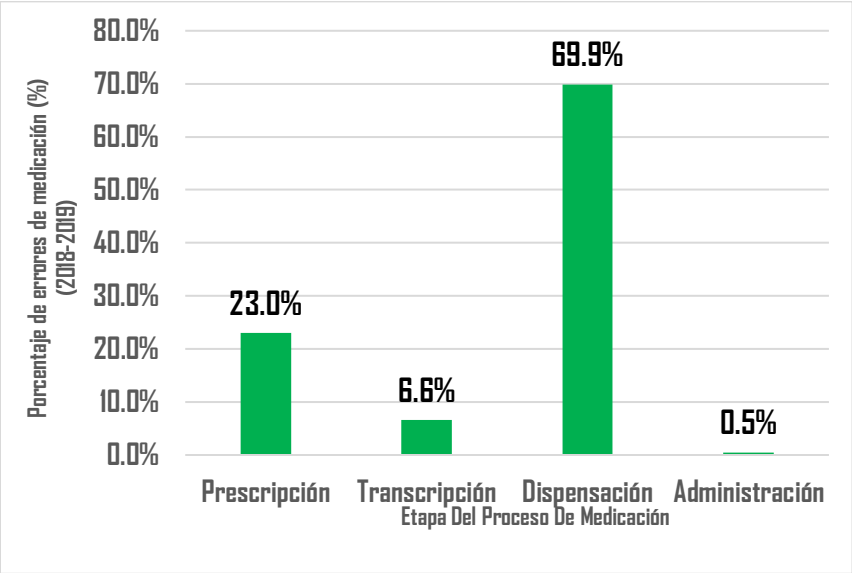
En Colombia, esta problemática se agrava debido a factores como la sobrecarga laboral del personal farmacéutico, la falta de capacitación en el manejo de tecnologías digitales y la carencia de sistemas automatizados que aseguren la correspondencia entre la prescripción médica y el medicamento entregado. Según un estudio de Audifarma (2018–2019) citado por El Espectador (2021), se registraron 29.853 errores de dispensación, equivalentes a 1,93 errores por cada 10.000 medicamentos. Aunque solo el 0,18 % fueron graves, la alta frecuencia de eventos demuestra deficiencias estructurales en los procesos de control y reporte, considerando además que apenas el 5 % de los errores son registrados oficialmente.

Los resultados del estudio original de Machado, M. et al. (2021), publicado en la revista Biomédica, amplían esta información, al mostrar que los errores se concentraron principalmente en la etapa de dispensación, con causas predominantemente humanas como la falta de concentración del personal, la similitud fonética entre medicamentos y los errores de almacenamiento. Asimismo, se identificaron diferencias en la frecuencia de errores según la ciudad y estabilidad en los reportes entre 2018 y 2019.

A continuación, se presentan las principales gráficas derivadas de este estudio, que ilustran la magnitud y distribución de los errores de medicación en el país.

Figura 1

Distribución de Errores por Etapa del Proceso (2018–2019)

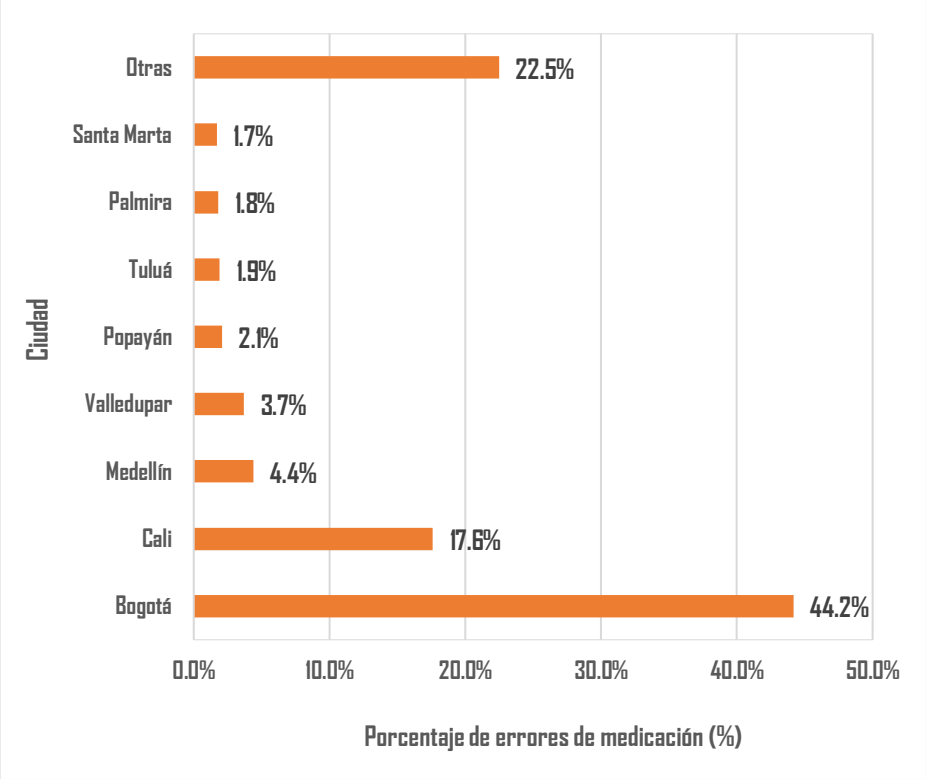


Nota. La mayor proporción de errores de medicación se presenta en la etapa de dispensación (69,9 %), seguida de la prescripción (23,0 %), mientras que la transcripción (6,6 %) y la administración (0,5 %) muestran incidencias menores. Adaptada de Machado, M. et al. (2021). https://doi.org/10.7705/biomedica.5544

Casi el 70 % de los errores se originan en la dispensación, lo que confirma que esta es la etapa más crítica del proceso farmacoterapéutico y la que más se beneficiaría de mecanismos automatizados de validación digital.

Figura 2

Errores de Medicación por Ciudad (2018–2019)

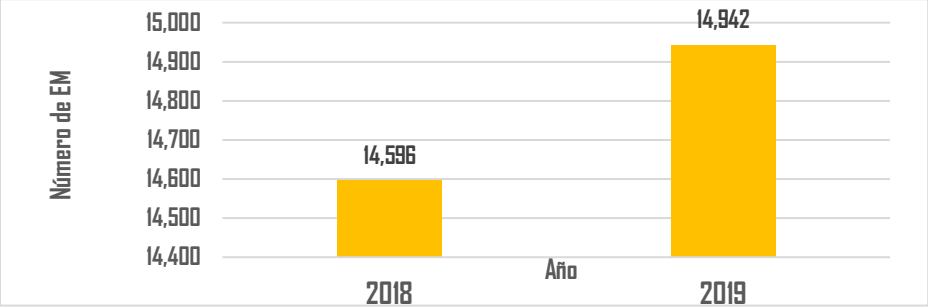


Nota. La mayor proporción de errores de medicación se concentra en Bogotá (44,2 %), seguida de Cali (17,6 %) y otras ciudades (22,5 %), mientras que las demás ciudades presentan porcentajes menores. Esto refleja una mayor incidencia en los centros urbanos con alto volumen de dispensación. Adaptada de Machado, M. et al. (2021). https://doi.org/10.7705/biomedica.5544

Las ciudades con mayor volumen de dispensación concentran los errores, lo que sugiere que el riesgo aumenta con la carga laboral y el volumen de recetas.

Figura 3

Evolución de los Errores de Medicación (2018–2019)

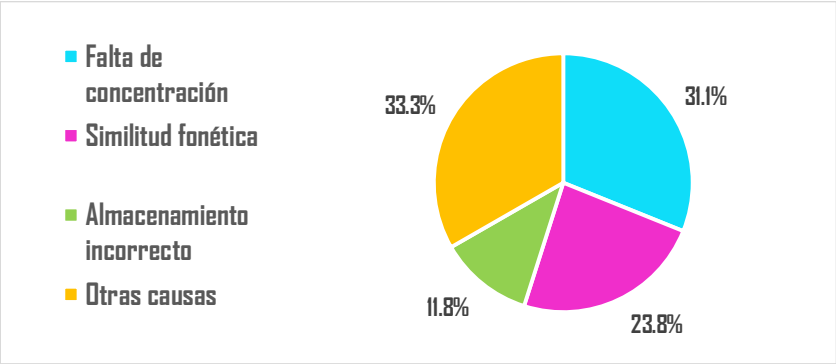


Nota. Se observa pasando de 14.596 en 2018 a 14.942 en 2019, lo que evidencia una tendencia al incremento en los reportes durante el periodo analizado. Adaptada de Machado, M. et al. (2021). https://doi.org/10.7705/biomedica.5544

Los errores se mantuvieron estables en los dos años evaluados, lo que evidencia que los procedimientos manuales tradicionales no han reducido significativamente los eventos de error, reforzando la necesidad de automatización.

Figura 4

Principales Causas de Errores de Medicación (2018–2019)



Nota. Las principales causas de errores de medicación fueron la falta de concentración (31,1 %), la similitud fonética (23,8 %) y el almacenamiento incorrecto (11,8 %), mientras que otras

causas representaron el 33,3 %. Adaptada de Machado, M. et al. (2021). https://doi.org/10.7705/biomedica.5544

Los factores humanos son los principales responsables de los errores; el desarrollo de un sistema automatizado con validación por código QR podría disminuir drásticamente su incidencia.

Figura 5

Porcentaje de Errores de Medicación Reportados vs. No Reportados.

Nota. Solo se reporta el 5 % de los errores de medicación; el 95 % no se registra. Adaptada de Machado, M. et al. (2021). https://doi.org/10.7705/biomedica.5544

El bajo nivel de notificación (solo 5 %) evidencia deficiencias en los mecanismos de reporte y seguimiento de errores de medicación. Esta falta de registro limita la capacidad de las autoridades para implementar medidas correctivas y refuerza la necesidad de sistemas automatizados que documenten cada dispensación en tiempo real.

La ausencia de herramientas tecnológicas integradas no solo incrementa los errores en la entrega, sino que también facilita el fraude y el desvío ilegal de medicamentos controlados. Muchos establecimientos aún dependen de verificaciones visuales o de documentos físicos que

pueden ser falsificados o reutilizados. La falta de trazabilidad digital impide además el monitoreo en tiempo real de las transacciones, limitando la capacidad de las autoridades sanitarias para ejercer un control efectivo.

En este contexto, surge la necesidad de desarrollar un sistema automatizado que combine validación electrónica de recetas y dispensación controlada de medicamentos, garantizando que el paciente reciba únicamente la dosis y tipo de fármaco autorizado por el médico tratante. Para ello, es indispensable integrar tecnologías de verificación digital mediante códigos QR y automatización robótica que reduzcan la intervención humana, aseguren la precisión y registren electrónicamente cada entrega.

Por tanto, el problema central que aborda este proyecto se puede expresar en la siguiente pregunta de investigación:

¿Cómo diseñar e implementar un sistema automatizado de dispensación de medicamentos que permita verificar electrónicamente las prescripciones médicas y garantizar la entrega segura, trazable y exacta de medicamentos controlados en farmacias?

La respuesta a este problema implica desarrollar una solución tecnológica que unifique el registro digital de recetas, la validación automática de prescripciones y la dispensación física controlada, con el fin de minimizar errores humanos, prevenir el uso indebido y mejorar la trazabilidad del proceso farmacéutico.

Este proyecto busca, entonces, desarrollar un prototipo funcional capaz de registrar, validar y dispensar medicamentos de forma automatizada, estableciendo una base para futuras implementaciones a gran escala en farmacias y centros hospitalarios.

Justificación

La dispensación de medicamentos controlados representa un proceso crítico dentro del sistema de salud, pues requiere altos niveles de precisión, trazabilidad y control. Los errores humanos, la automedicación y el uso indebido de fármacos de prescripción restringida constituyen un riesgo permanente para la seguridad de los pacientes y la salud pública. En Colombia, a pesar de los esfuerzos de entidades como el Instituto Nacional de Vigilancia de Medicamentos y Alimentos (INVIMA), gran parte de las farmacias aún realiza la validación de recetas de forma manual, lo que incrementa el margen de error y dificulta la supervisión en tiempo real de la distribución de medicamentos controlados.

A nivel internacional, organismos como la Organización Mundial de la Salud (OMS) y la Food and Drug Administration (FDA) promueven el uso de herramientas tecnológicas para garantizar la seguridad en la cadena de suministro de medicamentos. Estas entidades recomiendan el empleo de sistemas digitales que integren prescripciones electrónicas, automatización y trazabilidad, con el fin de reducir los riesgos de falsificación, sobredosificación o entrega incorrecta. El desarrollo de tecnologías de automatización farmacéutica se alinea, por tanto, con estas directrices globales y con los objetivos de salud pública nacional.

En este contexto, el presente proyecto propone el desarrollo de un prototipo automatizado para la dispensación de medicamentos controlados con verificación electrónica de prescripciones mediante códigos QR, que permita reducir la intervención manual y asegurar la exactitud en cada entrega. La solución integra una base de datos en la nube para el registro de prescripciones, una API de validación intermedia y un sistema físico de dispensación controlado por microprocesadores. Con ello, se busca garantizar que el medicamento entregado coincida

estrictamente con lo indicado por el médico tratante, evitando duplicaciones, falsificaciones o uso no autorizado.

Desde una perspectiva social, el proyecto contribuye a mejorar la seguridad del paciente y a prevenir el abuso de sustancias controladas, fortaleciendo las políticas públicas de control sanitario. Desde un enfoque tecnológico, promueve la adopción de sistemas inteligentes en entornos farmacéuticos, impulsando la transición hacia modelos de salud digital. Finalmente, desde un punto de vista académico y profesional, representa una oportunidad para aplicar conocimientos de ingeniería electrónica, automatización, control y desarrollo de software en la solución de una problemática real del sector salud. En síntesis, este proyecto se justifica por su impacto integral en tres dimensiones:

Sanitaria: al garantizar la dispensación segura y controlada de medicamentos.

Tecnológica: al incorporar un sistema automatizado e interoperable con bases de datos y validación electrónica.

Académica: al aportar una experiencia práctica que fortalece las competencias profesionales en el diseño de sistemas embebidos aplicados al sector salud.

De esta forma, el trabajo propuesto no solo responde a una necesidad técnica actual, sino que también constituye una contribución innovadora al fortalecimiento del control y la trazabilidad en los procesos de dispensación farmacéutica.

Este proyecto no solo automatiza un proceso técnico, sino que incorpora control, validación digital y trazabilidad, aportando directamente al cumplimiento de políticas de seguridad en medicamentos que hoy demanda el sistema de salud colombiano.

Objetivos

Objetivo General

Desarrollar un sistema automatizado de dispensación y validación de medicamentos controlados en farmacias, que garantice la entrega correcta según las prescripciones médicas mediante el uso de una base de datos y la validación con códigos QR, asegurando la trazabilidad, seguridad y precisión en todo el proceso.

Objetivos Específicos

Diseñar un sistema seguro y accesible para el registro de prescripciones médicas, utilizando una base de datos que permita almacenar y consultar recetas electrónicas de forma eficiente para su posterior validación.

Desarrollar un sistema de validación mediante códigos QR, que permita a los pacientes escanear el código recibido para verificar la autenticidad de la receta y su correspondencia con los medicamentos a dispensar, e integrar el registro electrónico de cada transacción en la base de datos, garantizando la trazabilidad y transparencia en la entrega de los medicamentos.

Construir un mecanismo automatizado para la dispensación de medicamentos, que asegure la entrega precisa del tipo y cantidad de fármacos indicados en la receta médica.

Validar el flujo completo del sistema, comprobando que todas las etapas, desde el registro de la receta, la validación mediante código QR, hasta la dispensación automatizada y funcionen correctamente, de manera segura y sin errores.

Marco Conceptual y Teórico

El presente proyecto se fundamenta en diversos conceptos técnicos y científicos relacionados con la automatización, el control electrónico, la gestión de datos y la seguridad informática aplicada al ámbito farmacéutico.

A continuación, se desarrollan los principales fundamentos conceptuales y teóricos que sustentan la propuesta.

Marco Conceptual

Errores de Medicación

Los errores de medicación son eventos prevenibles que ocurren en cualquier etapa del proceso de uso de medicamentos: prescripción, preparación, dispensación o administración, y que pueden causar daño al paciente. Estos errores son producto de múltiples factores, como fallas en la comunicación, deficiencias en la capacitación del personal o ausencia de protocolos estandarizados.

En América Latina, los errores de medicación generan impactos sanitarios y económicos significativos, por lo que la implementación de sistemas automatizados y estandarizados es fundamental para garantizar la seguridad del paciente (Fifarma, 2024)

Código QR (Quick Response)

El código QR es una herramienta de identificación digital de rápida lectura que puede almacenar información compleja de manera compacta. En el ámbito sanitario, su aplicación permite mejorar la gestión de citas, el acceso a historiales médicos y la validación de recetas electrónicas. Además, facilita el acceso rápido a registros médicos, instrucciones de medicación, material educativo y resultados de pruebas, tanto para pacientes como para profesionales de la salud. Su uso en hospitales y clínicas optimiza la administración de

medicamentos, el seguimiento de pacientes y la comunicación entre los proveedores de salud y los pacientes. Su integración en los procesos farmacéuticos fortalece la trazabilidad y la autenticidad de las prescripciones, reduciendo el riesgo de fraude o duplicación (Joshi, P., y Sawant, S., 2024).

Automatización Farmacéutica

Se refiere al uso de sistemas tecnológicos que permiten optimizar la preparación, almacenamiento, distribución y dispensación de medicamentos. A través de sensores, actuadores y software especializado, la automatización farmacéutica busca reducir errores, aumentar la eficiencia y liberar tiempo del personal sanitario para tareas clínicas. Este campo ha evolucionado hacia el uso de inteligencia artificial, robótica e Internet de las cosas (IoT) para mejorar la seguridad del paciente (PharmD, M. B. 2025)

Sistema Embebido

Un sistema embebido es un conjunto de hardware y software diseñado para realizar tareas específicas, a diferencia de los sistemas generales. Integra componentes como microcontroladores, sensores y unidades de comunicación. En el contexto del presente proyecto, los sistemas embebidos son esenciales para la operación coordinada de los microcontroladores Raspberry Pi, ESP32 y Arduino, que gestionan la lectura de códigos QR, la comunicación de datos y la dispensación física de medicamentos (Yul Chu, & Park, J. H. ,2023).

Bases de Datos

Una base de datos es un conjunto estructurado de información que permite el almacenamiento y la consulta eficiente de datos. En este proyecto se utiliza MongoDB Atlas, una base de datos NoSQL alojada en la nube, ideal para manejar grandes volúmenes de datos no

estructurados, permitiendo la conexión simultánea entre la API médica, la API intermedia y el dispensador físico (Božić, R., 2022).

Marco Teórico

En los últimos años, la automatización de la dispensación farmacéutica ha sido objeto de múltiples investigaciones orientadas a garantizar seguridad, precisión y trazabilidad en la entrega de medicamentos.

Según Shanthini E. et al. (2024), la implementación de sistemas automáticos basados en tecnología QR permite reducir significativamente los errores humanos y los tiempos de atención en farmacias y hospitales. Estos sistemas integran software de gestión, lectores de códigos QR, sensores y mecanismos electromecánicos de dispensación.

De manera complementaria, Muthulakshmi et al. (2023) proponen un modelo de gestión sanitaria con encriptación basada en identidad (IBE) y control de acceso por roles, asegurando la confidencialidad de los datos médicos. Su arquitectura, fundamentada en la nube, permite validar prescripciones electrónicas y garantizar el acceso seguro solo a personal autorizado, principios aplicables al presente proyecto.

Chavhan et al. (2022) realizan un análisis comparativo de tecnologías utilizadas en dispensadores automáticos, destacando la combinación de sensores infrarrojos, bases de datos NoSQL y microcontroladores como el enfoque más eficiente para lograr trazabilidad y seguridad. Este modelo es directamente coherente con la arquitectura técnica del prototipo propuesto.

Asimismo, Shanthini M. et al. (2023) desarrollan un dispensador IoT programable que mejora la adherencia terapéutica, al permitir la dosificación automatizada y monitoreo remoto.

Este tipo de sistemas demuestra la viabilidad del uso de tecnologías embebidas para dispensación precisa y personalizada.

Estos avances reafirman la pertinencia de integrar en este proyecto hardware embebido, IoT, y validación digital mediante QR, con miras a ofrecer un sistema confiable y seguro.

En conjunto, las investigaciones revisadas confirman que la automatización farmacéutica, apoyada en tecnologías digitales y sistemas embebidos, representa una solución efectiva para reducir errores, mejorar la eficiencia operativa y garantizar la trazabilidad en la gestión de medicamentos. El presente proyecto se sustenta en estos principios para diseñar un sistema que integre validación electrónica, control automatizado y registro digital en un solo flujo funcional.

Metodología CDIO

El proyecto se fundamenta en la metodología CDIO (Conceive, Design, Implement, Operate), la cual orienta el desarrollo integral del sistema automatizado de dispensación de medicamentos desde su concepción teórica hasta su validación práctica. Este enfoque permite estructurar el trabajo de manera progresiva y controlada, garantizando la trazabilidad de cada fase de diseño, implementación y operación del prototipo.

Concebir

En la fase de Concepción se identificó la problemática principal: la necesidad de mejorar la precisión, trazabilidad y control en la dispensación de medicamentos controlados. A partir de este diagnóstico, se definieron los requerimientos funcionales del sistema y se establecieron los objetivos específicos del proyecto.

Durante esta etapa se ideó la arquitectura general del sistema automatizado, conformada por tres módulos principales: el módulo de gestión de prescripciones médicas, el módulo de validación mediante códigos QR, y el módulo de dispensación automatizada de medicamentos, los cuales se ilustran en las Figuras 6, 7 y 8. Estas figuras reflejan la división funcional del sistema y su interconexión, evidenciando la integración entre el software de gestión médica y la infraestructura física encargada de la dispensación.

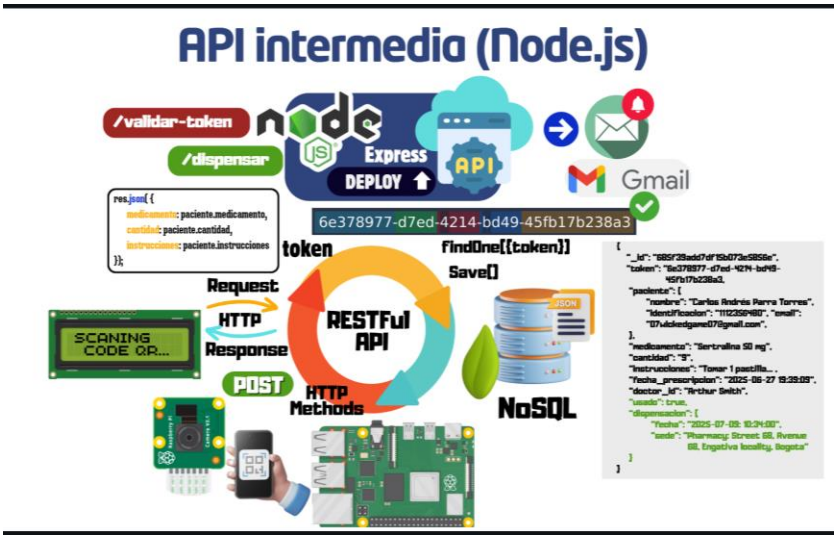
Figura 6

Módulo de Prescripciones Médicas (Frontend, Backend y BD)

Nota. La figura ilustra la arquitectura general del módulo de gestión de prescripciones médicas, que integra las capas de frontend, backend y base de datos. Elaboración propia.

Figura 7

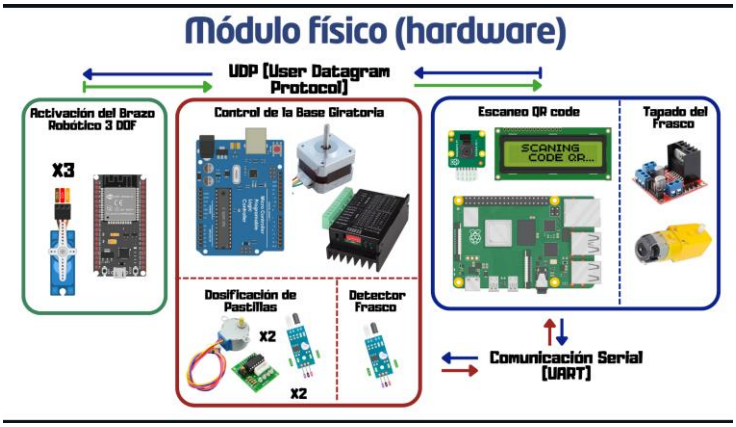
Módulo de Validación de Recetas mediante Código QR



Nota. La figura muestra la arquitectura del módulo de validación de recetas médicas, desarrollado con Node.js y Express. Elaboración propia.

Figura 8

Módulo de Dispensación Automatizada de Medicamentos



Nota. El módulo incluye la Raspberry Pi para escaneo de QR, comunicación serial (UART) y el tapado del frasco. El ESP32 controla el brazo robótico mediante UDP, y el Arduino Uno gestiona la base giratoria y la dosificación de pastillas. Elaboración propia.

Para garantizar la correcta planificación del proyecto, se elaboró un cronograma de actividades que permitió organizar las fases de desarrollo de acuerdo con la metodología CDIO. Este cronograma abarcó desde la concepción del problema hasta la validación del prototipo, distribuyendo las tareas de diseño, implementación y pruebas en un periodo de ocho semanas. En la Tabla 1 se presenta el detalle de las actividades programadas y su distribución temporal.

Tabla 1

Cronograma de Actividades

Objetivos	Actividades	S1	S2	S3	S4	S5	S6	S7	S8
General	Planificación y gestión del sistema automatizado completo	X							
Específico 1	Análisis de requisitos funcionales y diseño de la base de datos para registro de prescripciones médicas	X	X						
	Implementación del backend y frontend para la creación de recetas electrónicas y generación de códigos QR.		X	X					
	Escaneo de códigos QR para validar recetas médicas			X	X				
Específico 2	Integración del validador con la base de datos y registro de transacciones				X	X			
	Verificación de autenticidad, trazabilidad y transparencia del sistema de validación					X			

Objetivos	Actividades	S1	S2	S3	S4	S5	S6	S7	S8
Específico 3	Diseño del sistema físico de dispensación (componentes, planos, simulaciones)				X	X			
	Implementación y programación de motores, sensores y actuadores del mecanismo					X	X		
Específico 4	Integración del sistema mecánico con la validación y base de datos						X	X	
	Validación completa del flujo (registro → validación QR → dispensación)							X	X

Nota. El cronograma muestra la planificación de las actividades del proyecto en ocho semanas (S1–S8), alineadas con los objetivos generales y específicos.

En concordancia con el cronograma establecido, se identificaron los recursos materiales, tecnológicos y humanos requeridos para el desarrollo del prototipo automatizado. En la Tabla 2 se describen los componentes electrónicos, dispositivos de control, herramientas de software y demás insumos necesarios, junto con su costo estimado y su relación con las fases del proyecto.

Tabla 2

Recursos Necesarios

Recurso	Descripción	Presupuesto
Equipo Humano	El proyecto es realizado de forma individual por un estudiante de Ingeniería, quien se encarga del diseño, desarrollo, programación, integración y pruebas del prototipo. Esto incluye la configuración de la Raspberry Pi, el desarrollo de la API, el montaje mecánico y la documentación técnica del proyecto.	$ 0.00
Equipos y Software	Raspberry Pi 3, Cámara Pi, motores servo y paso a paso, sensores infrarrojos, actuadores, drivers, cables jumper, hosting en Railway o Render, fuente de alimentación.	$ 800,000.00
Viajes y Salidas de Campo	Desplazamientos para la adquisición de componentes, materiales y pruebas de campo del prototipo en entornos reales.	20,000.00
Materiales y suministros	Balso, herramientas básicas, elementos de construcción y prototipado.	100,000.00
Bibliografía	Acceso a recursos educativos gratuitos en línea, incluyendo tutoriales técnicos, documentación oficial de hardware y	$ 30,000.00

Recurso	Descripción	Presupuesto
	software, foros de desarrolladores y comunidades de código abierto.	
TOTAL		950,000.00

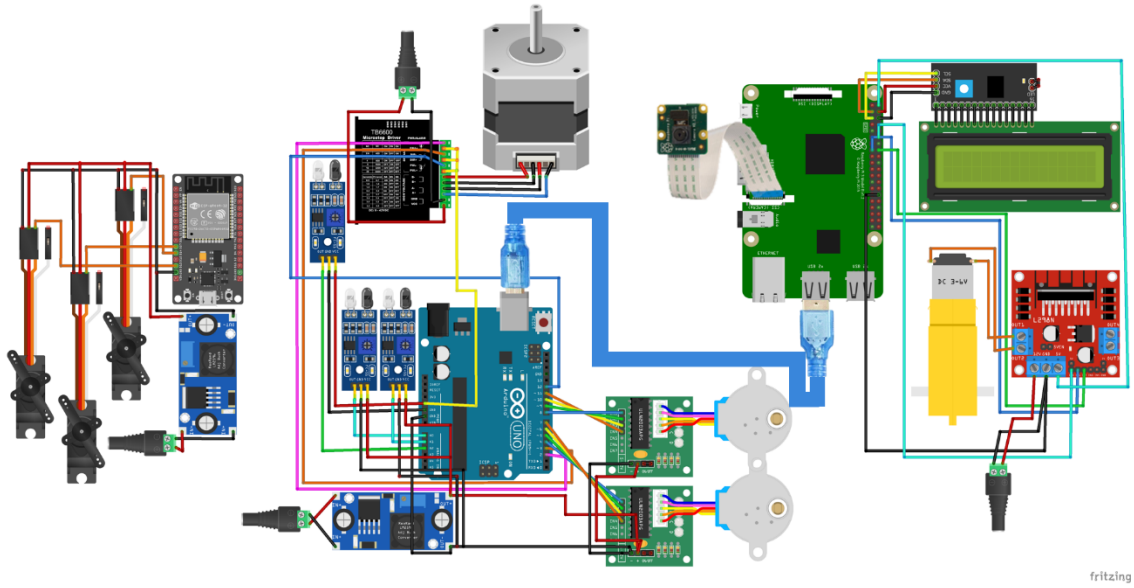
Nota. La tabla presenta el presupuesto estimado del proyecto con sus recursos humanos, materiales y tecnológicos.

Diseñar

La fase de Diseño implicó el desarrollo del modelo electrónico, estructural y lógico del sistema. Se elaboró el diagrama de conexión electrónica mostrado en la Figura 4, diseñado en Fritzing, que detalla la interconexión entre los microcontroladores (Raspberry Pi, Arduino Uno y ESP32), los sensores infrarrojos, los actuadores y los módulos de comunicación utilizados para controlar los procesos de dispensación. Este esquema permitió definir la topología de control, la distribución eléctrica y las rutas de señal necesarias para garantizar una comunicación estable entre los distintos módulos.

Figura 9

Conexión Electrónica

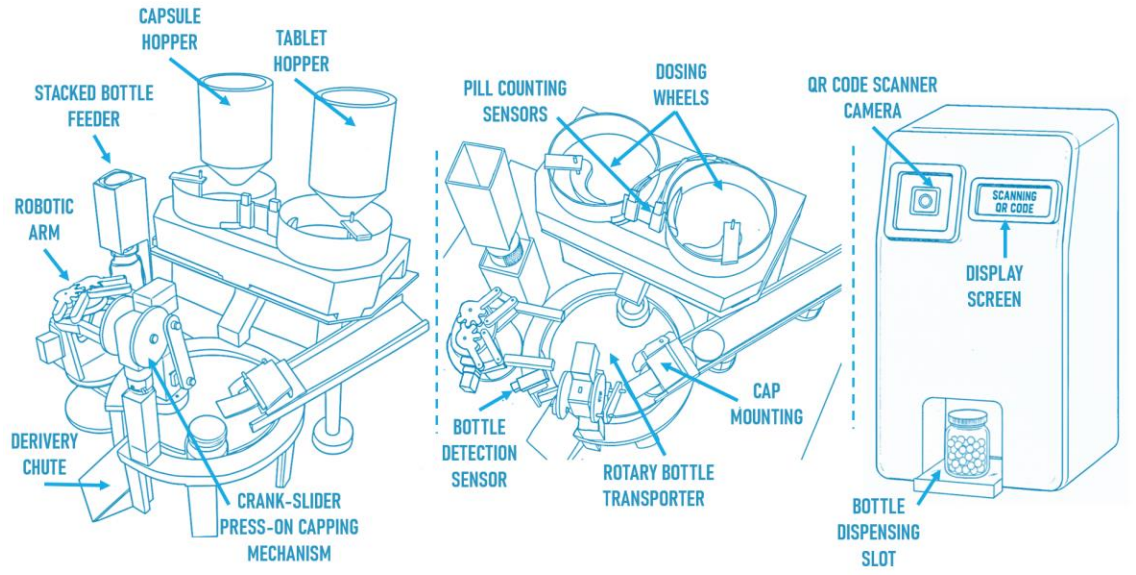


Nota. El diagrama muestra la conexión electrónica entre los distintos componentes del sistema, incluyendo microcontroladores, sensores, actuadores y módulos de comunicación. Elaboración propia mediante Fritzing.

Adicionalmente, se elaboró el boceto estructural del mecanismo de dispensación (Figura 10), donde se representa la disposición física del brazo robótico, la base giratoria, las tolvas de dosificación y el sistema de tapado. Estas representaciones gráficas facilitaron la planificación de la construcción física del prototipo, asegurando la correcta distribución de los componentes electrónicos y mecánicos.

Figura 10

Boceto Estructural del Mecanizado



Nota. Estas representaciones gráficas facilitaron la planificación de la construcción física del prototipo, asegurando la correcta distribución de los componentes electrónicos y mecánicos. Elaboración propia.

En esta etapa también se diseñaron los algoritmos que modelan la lógica operativa de cada módulo del sistema, constituyendo el componente lógico del diseño CDIO.

El Algoritmo 1, ilustrado en la Figura 11, describe el flujo de registro y validación de prescripciones médicas, desde la creación de la receta en la API hasta la generación y envío del código QR al paciente.

El Algoritmo 2, mostrado en la Figura 12, representa el proceso de validación del código QR en el punto farmacéutico, donde la Raspberry Pi escanea el código, verifica la autenticidad del token en la base de datos y autoriza la dispensación.

Finalmente, el Algoritmo 3, presentado en la Figura 13, modela la secuencia de dispensación física, incluyendo la activación del brazo robótico, la dosificación mediante sensores y motores paso a paso, y el cierre automático del frasco.

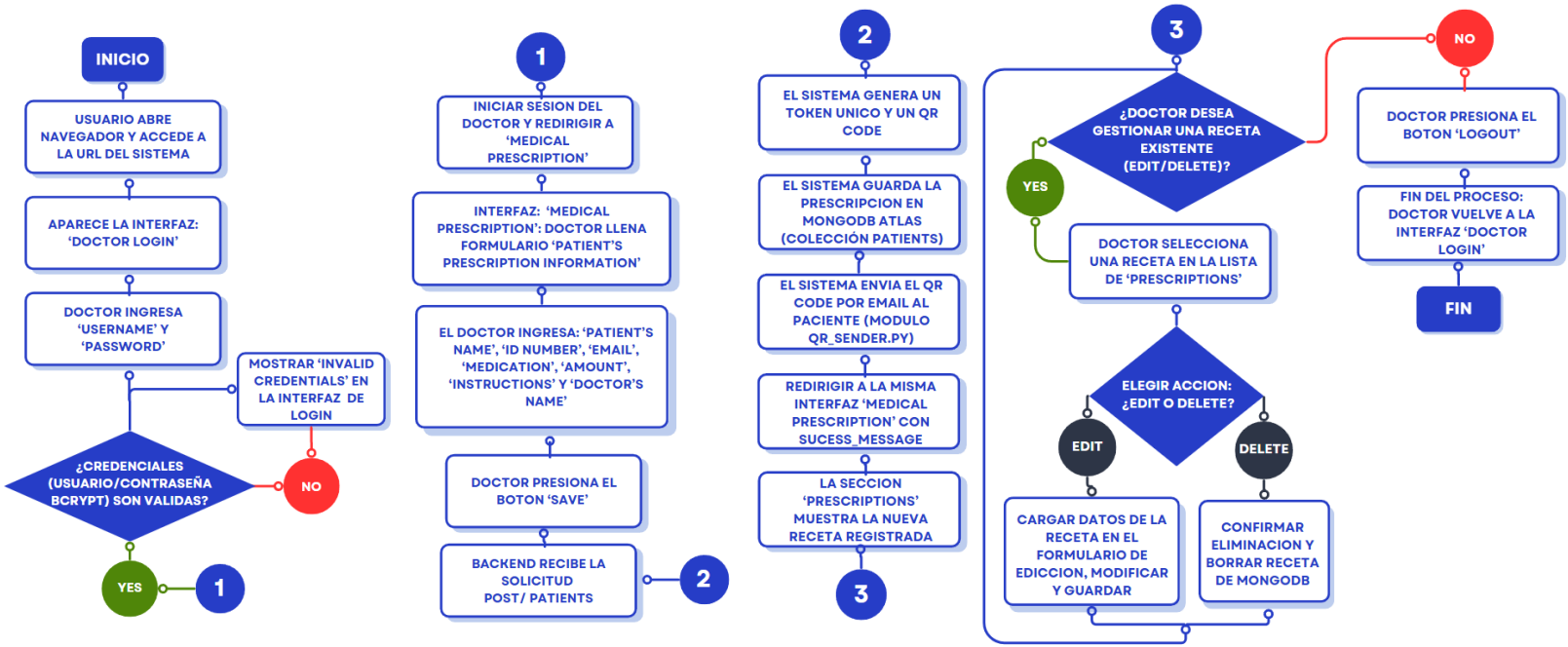
Estos algoritmos constituyen la base lógica del sistema y complementan las figuras técnicas al expresar de manera secuencial la interacción entre el software y el hardware.

Sirvieron como guía directa para la programación de los microcontroladores durante la fase de implementación.

Su desarrollo detallado y flujos de ejecución se presentan en el capítulo Diseño de la Solución, donde se explica su estructura y aplicación dentro del sistema.

Figura 11

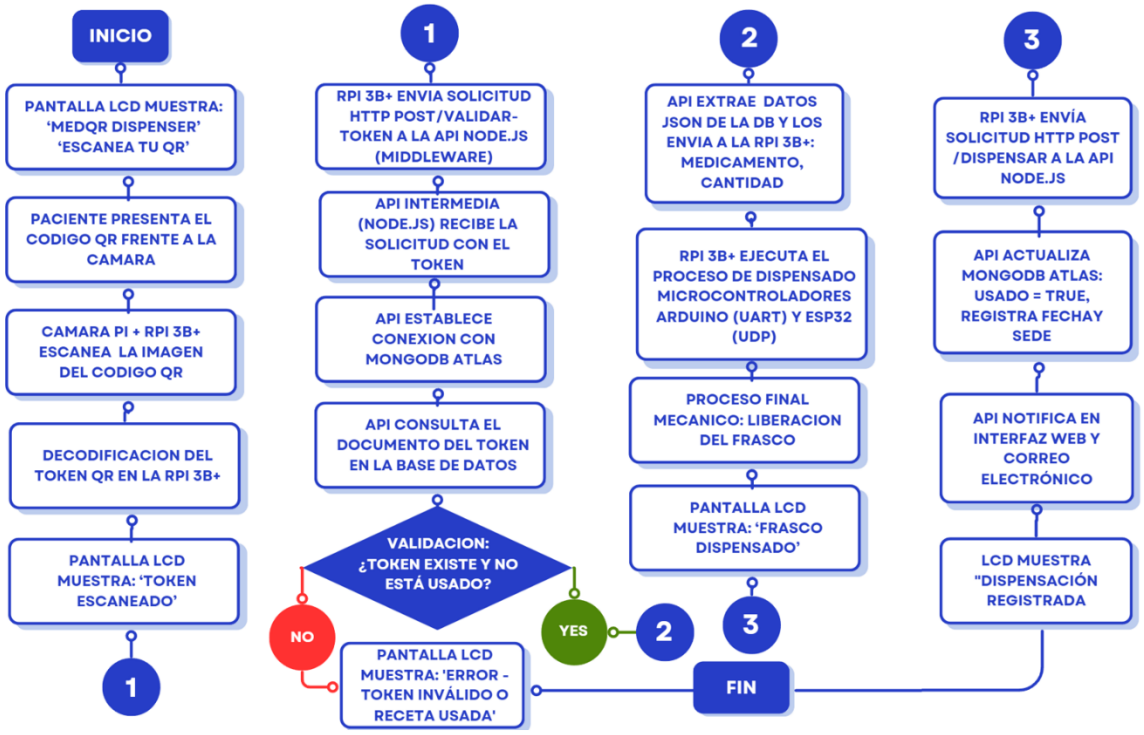
Primera Etapa: Registro y Validación de Prescripciones



Nota. Flujo lógico del módulo de prescripciones médicas, que registra, valida y genera el código QR asociado a cada receta. Elaboración propia.

Figura 12

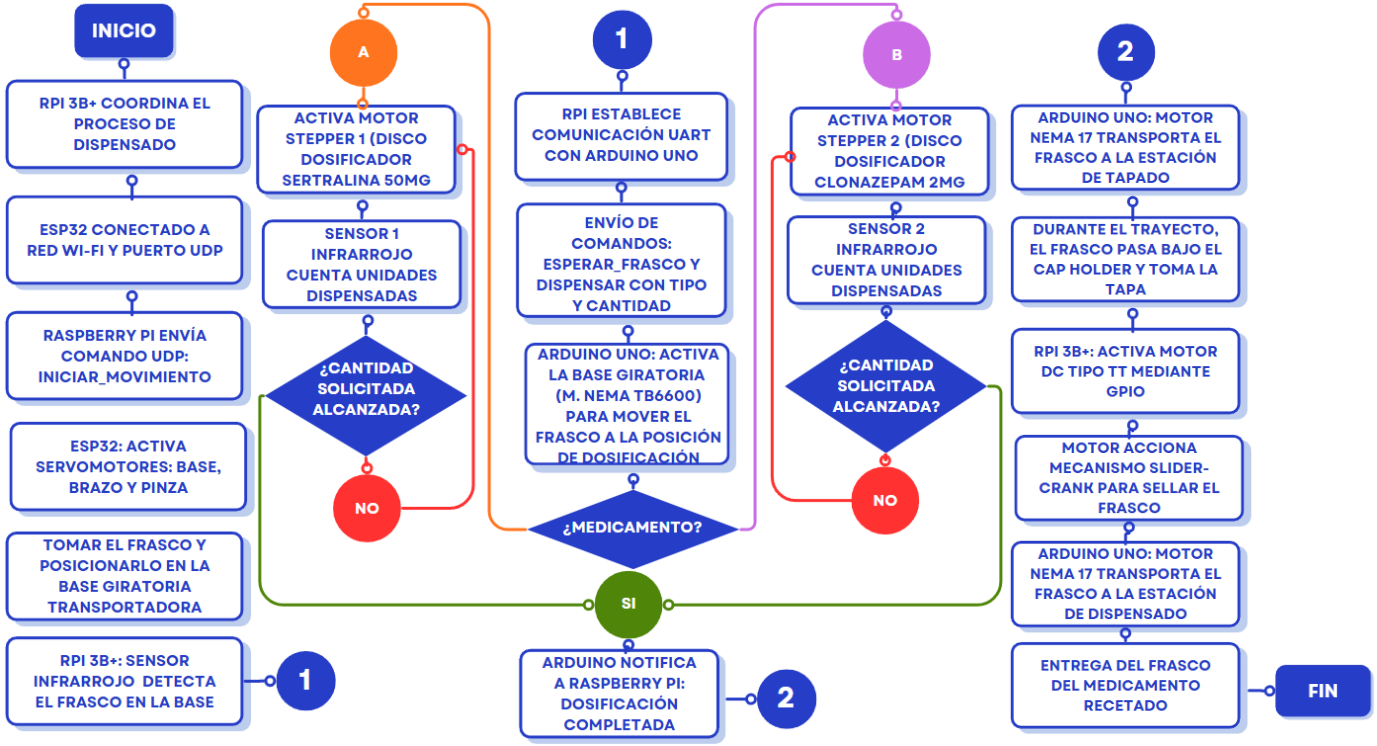
Segunda Etapa: Validación de Tokens QR para Dispensación Farmacéutica



Nota. Proceso de validación del código QR mediante la Raspberry Pi, que consulta la base de datos y autoriza la dispensación. Elaboración propia.

Figura 13

Tercera Etapa: Mecanizado del Dispensado



Nota. Secuencia de dispensación donde la Raspberry Pi coordina al ESP32 y al Arduino Uno, controlando el brazo robótico, la base transportadora y los discos dosificadores. Elaboración propia

Especificaciones Técnicas de Software

En la primera fase, se llevará a cabo el desarrollo de un sistema para registrar y gestionar prescripciones médicas de pacientes. Este sistema utiliza una API RESTful construida con el framework Flask de Python y una base de datos MongoDB Atlas para almacenar de manera remota los datos de las prescripciones médicas. Los médicos pueden acceder a la aplicación desde un entorno localhost durante la fase de desarrollo.

Cuando un doctor realiza una prescripción médica, se crea un registro en la base de datos que contiene información clave del paciente, tipo de medicación y cantidad de pastillas. Este registro se almacena en MongoDB Atlas, lo que permite que los datos sean accesibles de manera remota desde cualquier ubicación con conexión a internet, sin depender de una red local específica. El sistema también genera un código QR único asociado con cada prescripción. Este código contiene un token firmado, que es utilizado para verificar que la prescripción ha sido registrada correctamente.

Este código QR se envía al correo electrónico del paciente, y en la base de datos se guarda un estado para saber si el código ha sido usado o no. El token utilizado en el código QR es un identificador único generado mediante un UUID, firmado con HMAC utilizando un hash SHA256, garantizando que no contiene datos sensibles y solo actúa como un identificador seguro.

Tabla 3

Componentes y Características del Backend de Prescripción Médica

Característica	Descripción
Backend	Flask (Python) - API RESTful para manejar las operaciones CRUD de las prescripciones médicas.
Base de Datos	MongoDB Atlas - Almacenamiento remoto de datos accesible desde cualquier lugar.

Autenticación	El sistema permite el inicio de sesión de los médicos con un entorno de login basado en HTML.
Conexión a MongoDB Atlas con URI seguro	La conexión al clúster remoto se realiza mediante una URI protegida con credenciales y parámetros de conexión seguros
Generación de Token	UUID + HMAC-SHA256 - Token único para cada prescripción, generado para seguridad y validación.
Generación de QR	Código QR generado para cada prescripción, que contiene el token único de la prescripción.
Envío de QR	El código QR es enviado por correo electrónico al paciente.
Verificación de QR	En la base de datos, se guarda el estado del QR, indicando si ha sido usado o no.
Comunicación	API RESTful con solicitudes GET y POST para interactuar con la base de datos.
Protocolo	HTTP(S) para la comunicación entre cliente y servidor.
Seguridad	Uso de HMAC para la firma de tokens, autenticación de médicos y encriptación de datos en la base de datos.
Entorno de Desarrollo	Localhost durante la fase de pruebas, con posibilidad de migrar a un entorno de producción en la nube.

Nota. La tabla detalla las características funcionales y de seguridad implementadas en el sistema backend para la gestión y validación de prescripciones médicas.

La segunda fase corresponde al proceso de validación y dispensación de medicamentos a través de un dispositivo dispensador automatizado ubicado en un punto farmacéutico. El sistema utiliza una Raspberry Pi (RPI) equipada con una cámara Pi que escanea el código QR recibido por el paciente al momento de su prescripción.

Una vez escaneado el código QR, se extrae y decodifica el token firmado que está embebido en él. La Raspberry Pi realiza una solicitud a una API intermedia, la cual actúa como enlace entre el dispensador y la base de datos remota (MongoDB Atlas).

Esta API se encarga de:

Verificar la existencia del token en la base de datos.

Confirmar que el token no haya sido usado previamente.

Validar la autenticidad del token.

Consultar los detalles de la prescripción, como el tipo de medicamento y la cantidad de pastillas prescritas.

Si todas las validaciones son exitosas, la Raspberry Pi activa el sistema de dispensación automática, permitiendo la entrega del medicamento según la cantidad autorizada.

Una vez realizada la dispensación, se actualiza el estado del token en la base de datos como "usado", impidiendo que sea reutilizado. Además, se puede registrar la hora, la fecha y el punto de dispensación como parte del historial de la prescripción para fines de trazabilidad.

Tabla 4

Sistema de Validación con Raspberry Pi y API Intermedia

Categoría	Elemento / Tecnología	Descripción / Función
Hardware Principal	Raspberry Pi 3B+	Ejecuta el software que escanea el QR y valida el token con la API intermedia.
Cámara	Raspberry Pi Camera Module v2	Captura el código QR presentado por el paciente.
Sistema Operativo	Raspberry Pi OS (32-bit)	Sistema operativo ligero y estable para la Raspberry Pi.
Lenguaje (RPi)	Python 3.2	Control del escaneo del QR, comunicación HTTP y procesamiento local.
Librerías (RPi)	pyzbar, OpenCV, requests	Decodificación de QR y envío de solicitudes HTTP a la API intermedia.
Tipo de Comunicación	HTTP/HTTPS (API RESTful)	La RPi se comunica con la API mediante peticiones GET/POST.
Protocolo de Red	TCP/IP	Protocolo base para la conexión de red entre la RPi e internet.
Conectividad	Wi-Fi / Ethernet	Permite la conexión de la RPi a internet para comunicarse con la API.
Lenguaje (API)	JavaScript (Node.js)	Lenguaje usado para construir la API que valida los tokens QR.
Framework de la API	Express.js	Framework web en Node.js para construir la API RESTful.

Categoría	Elemento / Tecnología	Descripción / Función
Base de Datos	MongoDB Atlas	Almacena prescripciones, tokens y su estado (used: true/false).
Verificación de Token	UUID + HMAC-SHA256	Seguridad del token escaneado; evita falsificaciones y protege los datos.
Formato de Datos	JSON	Formato utilizado para intercambiar datos entre la RPi y la API.
Gestión del Token	Campo used: true/false	Evita el uso múltiple del mismo QR; garantiza control sobre la dispensación.
Alimentación	Fuente 5V / 2.5A	Fuente de poder estable para la RPi y la cámara.

Nota. La tabla describe los componentes y tecnologías empleados en el sistema de validación de recetas mediante la Raspberry Pi y una API intermedia para la verificación del código QR.

Especificaciones Técnicas de Hardware

La tercera fase consiste en la automatización del proceso de dispensación de medicamentos, utilizando un brazo robótico controlado por un ESP32. Este brazo recoge el frasco del tubo alimentador y lo coloca en una base giratoria transportadora controlada por un motor paso a paso NEMA 17. Posteriormente, un sistema de dosificación con sensores infrarrojos y motores paso a paso cuenta las pastillas, y un motor reductor sella el frasco antes de entregarlo al usuario. El proceso se inicia con el escaneo de la receta usando una cámara Pi, y la información se muestra en una pantalla LCD. La Raspberry Pi gestiona y coordina todo el sistema, controlando los

microcontroladores ESP32 y Arduino a través de UDP (para el ESP32) y UART serial (para el Arduino), y validando la prescripción mediante una API intermedia.

Tabla 5

Fases del Proceso de Mecanizado

Fase	Descripción	Componentes Involucrados
1. Activación del Brazo Robótico	El brazo robótico de 3 grados de libertad recoge el frasco desde un tubo alimentador y lo posiciona en la base giratoria transportadora.	ESP32 (Control del brazo robótico), Servo Motores 180° (para el movimiento del brazo), Raspberry Pi (para la secuenciación).
2. Control de la Base Giratoria	La base giratoria comienza a rotar a una posición específica para iniciar la dosificación de las pastillas.	Arduino Uno (controla el motor paso a paso NEMA 17 con driver TB6600 después de detectar el frasco con sensor IR), Raspberry Pi (para la secuenciación).
3. Dosificación de Pastillas	El sistema de dosificación comienza a contar las pastillas. Utiliza dos tolvas para cápsulas y tabletas, con discos rotatorios y sensores infrarrojos para contar las pastillas.	Una vez que reciba el comando por RPI, el Arduino Uno (controla los Motores paso a paso 28BYJ-48, Sensores infrarrojos, para la dosificación y conteo de pastillas), Raspberry Pi. (Ordena la dosificación, y envía datos como: tipo de medicamento y cantidad).

Fase	Descripción	Componentes Involucrados
4. Tapado del Frasco	Una vez que la cantidad de pastillas es correcta, el frasco pasa a la posición de tapado, donde un motor reductor con mecanismo *slide crank* aplica presión para cerrar el frasco.	Motoreductor (para el tapado). Raspberry Pi
5. Entrega del Medicamento	Finalmente, el frasco sellado es entregado al usuario en la estación de dispensación.	Raspberry Pi (gestión y control de la secuencia).

Nota. La tabla describe las fases secuenciales del sistema de dispensación automatizada, desde la activación del brazo robótico hasta la entrega del medicamento al usuario.

Tabla 6

Tipos de Comunicación y Protocolos

Componente	Tipo de Comunicación	Descripción
ESP32 y Raspberry Pi	UDP (User Datagram Protocol)	Comunicación rápida entre el ESP32 (control del brazo robótico) y la Raspberry Pi para iniciar el proceso.
Raspberry Pi y Arduino Uno	Comunicación Serial (UART)	El Arduino Uno recibe comandos de la Raspberry Pi para controlar el motor paso a paso de la base giratoria, motores 28BYJ-48, y sensores infrarrojos (dosificación pastillas)

Componente	Tipo de Comunicación	Descripción
Raspberry Pi	GPIO (General Purpose Input/Output)	Controla el motorreductor mediante los pines GPIO, usa una cámara Pi a través del puerto CSI y una pantalla LCD con módulo I2C. Comunicación con la API mediante solicitudes RESTful a través de Wi-Fi.

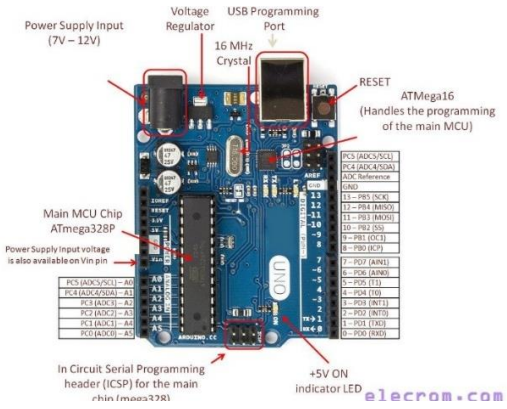
Nota. Detalla los protocolos de comunicación utilizados entre los distintos módulos de hardware y la del sistema de dispensación automatizada. Elaboración propia.

Especificaciones de los Componentes

En esta sección se describirán las características, funciones, capacidades, consumo energético y especificaciones técnicas de los microcontroladores, sensores y demás componentes eléctricos y electrónicos del sistema, proporcionando una visión general de su papel en la automatización y funcionamiento del proceso.

Figura 14

Arduino Uno



Nota. Controla los discos dosificadores de pastillas y maneja la base transportadora giratoria. Tomada de Instructables (2019, junio 1). Arduino Ultrasonic Distance Meter

With 7 Segment Display. https://www.instructables.com/Arduino-Ultrasonic-Distance-Meter-With-7-Segment-D/

Tabla 7

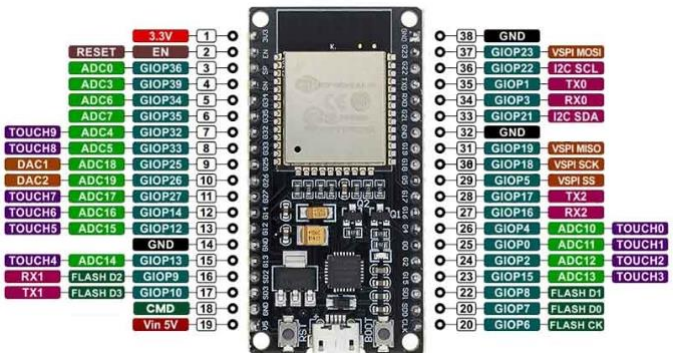
Características Electrónicas del Arduino

Categoría	Especificación
Microcontrolador	ATmega 328P.
Voltaje límite de entrada	(6 – 20) V
Pines	14 digitales I/O – (6 PWM).
Voltaje de trabajo	5V
Entradas Análogas	6
SRAM	2KB
EEPROM	1KB
Corriente máxima en pines I/O	40 mA.

Nota. La tabla presenta las principales especificaciones del microcontrolador Arduino Uno. Adaptada de Arduino (2025). Arduino Uno Rev3 Technical Specifications. https://www.arduino.cc/en/Main/ArduinoBoardUno

Figura 15

Microcontrolador ESP32



Nota. Controla el mecanizado del brazo robótico, las acciones son coordinadas por la RPi a través de UDP, lo que permite la ejecución sincronizada de las tareas de

dispensación. Tomada de Arduino Forum (2024, diciembre 19). Esp32 with qr sensor gm60. https://forum.arduino.cc/t/esp32-with-qr-sensor-gm60/1333870

Tabla 8

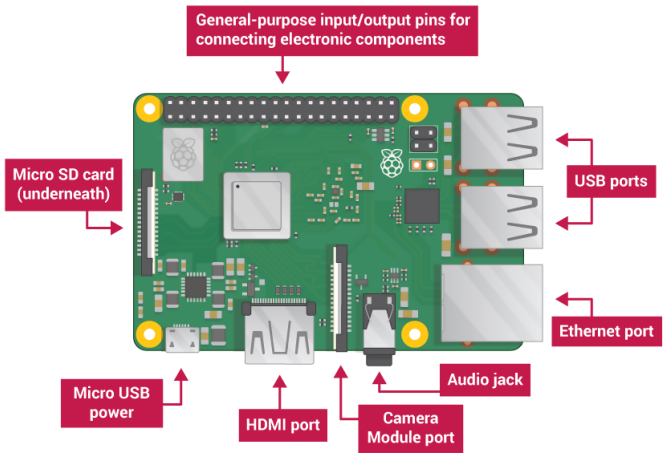
Características Electrónicas del ESP32

Categoría	Especificación
Memoria	520 KB SRAM, 4 MB Flash
Conectividad	Wi-Fi 802.11 b/g/n @ 2.4GHz, Bluetooth 4.2
GPIO	24 digitales, 18 ADC (12 bit), 2 DAC (8 bit)
Voltaje	Alimentación 5V (USB), E/S a 3.3V
UART	2 interfaces UART
USB-Serial	CP2102
Consumo de Corriente	Hasta 260 mA (activo), hasta 10 µA (suspensión profunda)

Nota. La tabla presenta las especificaciones técnicas del módulo ESP32 DEVKIT V2. Adaptada de Espressif Systems. (2021). ESP32-WROOM-32 Datasheet. Espressif Systems. https://www.espressif.com/en/products/socs/esp32/resources

Figura 16

Microprocesador Raspberry pi 3B+



Nota. Como unidad central de control del sistema, coordinando los microcontroladores ESP32 y Arduino.Tomada de Posada, F. (2020, septiembre 8). Raspberry Pi: mini-ordenador para proyectos educativos. https://canaltic.com/blog/?p=3734

Tabla 9

Características Electrónicas del Raspberry pi 3B+

Categoría	Especificación
Procesador	Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4GHz
Memoria RAM	1 GB
Conectividad	Wi-Fi 802.11b/g/n/ac (2.4GHz y 5GHz), Bluetooth 4.2, BLE, Ethernet USB 2.0 (300Mbps), 4 × USB 2.0
Video y Sonido	HDMI completo, MIPI DSI, MIPI CSI, salida estéreo 4 polos + video compuesto
Multimedia	H.264/MPEG-4 (1080p30 decode y encode), OpenGL ES 1.1/2.0
Almacenamiento	Micro SD (SO y datos)

Alimentación	5V/2.5A por micro USB, 5V por GPIO, PoE con PoE HAT opcional
Vida útil en producción	Hasta al menos enero de 2028

Nota. Especificaciones electrónicas del modelo Raspberry Pi 3B. Adaptada de Raspberry Pi Foundation. (2018). Raspberry Pi 3 Model B+ Product Brief. Raspberry Pi Foundation. https://www.raspberrypi.com/documentation/computers/raspberry-pi.html

Figura 17

TB6600 Stepper Motor Driver

Nota. Controlador de motor a pasos TB6600. Tomada de Amazon. (2025). DFROBOT TB6600 Stepper Motor Driver. https://www.amazon.com.be/-/en/DFROBOT-TB6600-Stepper-Motor-Driver/dp/B073TLMVZJ

Tabla 10

Características Electrónicas del TB6600 Stepper Motor Driver

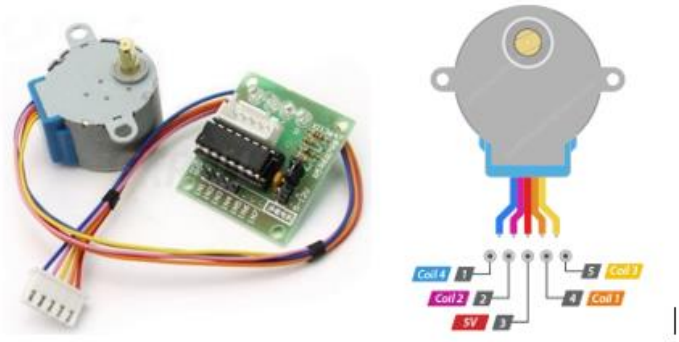
Categoría	Especificación
Corriente de Entrada	0 ~ 5 A
Corriente de Salida	0.5 ~ 4.0 A
Señal de Control	3.3 V ~ 24 V

Categoría	Especificación
Potencia Máxima	160 W
Micro pasos (Micro Step)	1, 2/A, 2/B, 4, 8, 16, 32
Temperatura de Operación	-10 ~ 45 °C
Humedad	Sin condensación
Dimensiones	96 × 56 × 33 mm (3.78 × 2.2 × 1.3 in)
Peso	0.2 kg
Chip de Control	TB67S109AFTG

Nota. La tabla presenta las especificaciones técnicas del controlador TB6600. Adaptada de Bakker, B. (2025, 21 abril). TB6600 Stepper Motor Driver with Arduino Tutorial. https://www.makerguides.com/tb6600-stepper-motor-driver-arduino-tutorial/

Figura 18

Motor Paso a Paso 28BYJ-48



Nota. Diagrama que muestra la conexión de las bobinas del motor 28BYJ-48 con el controlador ULN2003. Tomada de Arduinokit Project. (2023, julio 15). Interfacing 28BYJ-48 stepper motor Arduino using ULN2003 driver. *https://arduinokitproject.com/28byj48-stepper-motor-arduino-tutorial/#google_vignette*

Tabla 11

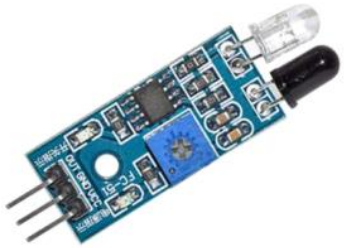
Características Electrónicas del Motor Paso a Paso 28BYJ-48

Elemento	Especificación
Voltaje	5V DC
Tipo de motor	Paso a paso unipolar
Fases	4
Relación de reducción	1/64
Ángulo por paso	≈0.088° (5.625° / 64)
Torque en carga	>34.3 mN·m
Driver	ULN2003A
Salidas del driver	4 canales, 500 mA c/u
Protección	Diodos para cargas inductivas
Indicadores	4 LEDs
Control manual	Jumper de encendido/apagado

Nota. Muestra las especificaciones técnicas del motor paso a paso 28BYJ-48. Adaptada de Electronilab. (2025). Motor paso a paso 28BYJ-48 0.3 Kgr/Cm. https://electronilab.co/tienda/motor-paso-a-paso-28byj-48-0-3kgr-cm/

Figura 19

Sensor de Proximidad Infrarrojo fc-51



Nota. El sensor FC-51 utiliza un emisor y un receptor infrarrojos para detectar objetos cercanos mediante la reflexión de la luz. Tomada de Bentrán, A. (2024). Sensor de proximidad infrarrojo FC-51. https://agelectronica.lat/pdfs/textos/O/OKY3127.PDF

Tabla 12

Características Electrónicas del Motor Paso a Paso 28BYJ-48

Parámetro	Descripción
Voltaje de trabajo	3V ~ 5V
Ángulo de cobertura	35°
Rango de detección	2 cm ~ 30 cm
Consumo de corriente	23 mA (a 3.3V), 43 mA (a 5V)
Circuito de detección	Basado en comparador LM393 y tecnología infrarroja
LED indicador	Estado de alimentación y detección de obstáculos
Sensibilidad	Ajustable mediante potenciómetro

Nota. La tabla muestra las principales especificaciones del sensor de proximidad infrarrojo FC-51 utilizado en sistemas de detección de obstáculos. Adaptada de Bentrán, A. (2024). Sensor de proximidad infrarrojo FC-51. https://agelectronica.lat/pdfs/textos/O/OKY3127.PDF

Figura 20

Micro Servo SG90

Nota. El micro servo SG90 es un actuador rotativo que permite controlar la posición angular mediante señales PWM, ampliamente utilizado en proyectos de robótica y automatización. Tomada de Vistronica. (2025). Micro servomotor SG90 9g. https://www.vistronica.com/robotica/motores/servomotores/micro-servomotor-sg90-9g-detail.html

Tabla 13

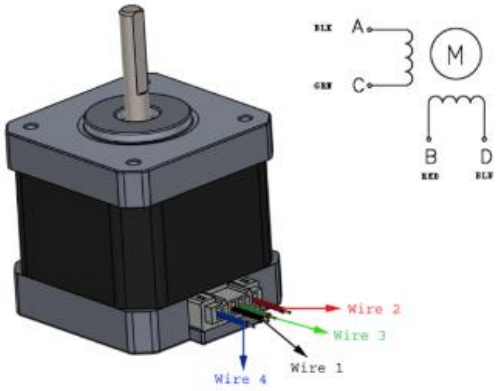
Características Electrónicas del Micro Servo SG90

Parámetro	Descripción
Peso	9 g
Dimensiones	22.2 x 11.8 x 31 mm (aprox.)
Torque de parada	1.8 kgf·cm
Velocidad de operación	0.1 s / 60°
Voltaje de operación	4.8 V (~5V)
Zona muerta (Dead band)	10 µs
Rango de temperatura	0 ºC – 55 ºC

Nota. La tabla presenta las principales especificaciones del micro servo SG90, un actuador utilizado en proyectos de robótica y control de movimiento. Adaptada de Vistronica. (2025). Micro servomotor SG90 9g. https://www.vistronica.com/robotica/motores/servomotores/micro-servomotor-sg90-9g-detail.html

Figura 21

NEMA 23 Stepper Motor



Nota. Muestra la estructura general y el diagrama de conexión de un motor paso a paso tipo NEMA 23. Tomada de Components101. (2025). NEMA 23 stepper motor. https://co.pinterest.com/pin/732538695625153703/

Tabla 14

Características Electrónicas del NEMA 23 Stepper Motor

Parámetro	Descripción
Voltaje nominal	3.2 V
Corriente nominal	2.8 A
Torque de retención	270 oz·in (≈1.9 N·m)
Ángulo por paso	1.8°

Parámetro	Descripción
Pasos por revolución	200
Número de fases	4
Número de cables	4 (bipolar)
Inductancia por fase	3.6 mH
Longitud del motor	3.1 pulgadas (≈78.7 mm)

Nota. Características eléctricas y mecánicas del motor paso a paso NEMA 23. Adaptada de Electronilab. (2025). Motor paso a paso NEMA 23 – 178.5 oz·in (1.26 Nm) – 200 pasos/vuelta. https://electronilab.co/tienda/motor-paso-a-paso-nema-23-125-oz-in-200-pasos-vuelta/

Figura 22

Servo Motor MG995

Nota. Base brazo robot. Tomada de Electronilab. (2025). Servomotor MG995 piñonería metálica. https://electronilab.co/tienda/mg995-tower-pro-servo-motor-metalico/

Tabla 15

Características Electrónicas del Servo motor MG995

Parámetro	Descripción
Peso	55 g

Parámetro	Descripción
Dimensiones	40.7 x 19.7 x 42.9 mm (aprox.)
Torque de parada	8.5 kgf·cm (a 4.8 V) / 10 kgf·cm (a 6 V)
Velocidad de operación	0.2 s / 60° (4.8 V), 0.16 s / 60° (6 V)
Voltaje de operación	4.8 V a 7.2 V
Ancho de banda muerta	5 µs
Ángulo de rotación	120° (±60° desde el centro)
Diseño	Doble rodamiento de bolas, estable y a prueba de golpes
Engranajes	Metálicos (mayor durabilidad)
Rango de temperatura	0 ºC – 55 ºC

Nota. Características eléctricas y mecánicas del servomotor MG995. Adaptada de Electronilab. (2025). Servomotor MG995 piñonería metálica. https://electronilab.co/tienda/mg995-tower-pro-servo-motor-metalico/

Figura 23

LM2596S Step-Down DC-DC Buck Converter

Nota. Empledo para la regulación de voltaje. Tomada de Electronilab. (2025). Módulo LM2596 DC-DC Buck reductor con voltímetro. https://electronilab.co/tienda/modulo-lm2596-dc-dc-buck-1-25v-35v-con-voltimetro/

Tabla 16

Características Electrónicas del DC-DC Buck Converter

Parámetro	Descripción
Voltaje de entrada	4 V – 40 V
Voltaje de salida	1.5 V – 35 V
Corriente de salida	2 A constante, hasta 3 A con disipación adicional
Precisión del voltímetro	±1%
Dimensiones	66 x 36 x 13 mm
Peso	22 g

Nota. Especificaciones del módulo LM2596. Adaptada de Electronilab. (2025). Módulo LM2596 DC-DC Buck reductor con voltímetro. https://electronilab.co/tienda/modulo-lm2596-dc-dc-buck-1-25v-35v-con-voltimetro/

Figura 24

Cámara para Raspberry Pi V3

Nota. Para escanear el token. Tomada de Electronilab. (2025). Cámara para Raspberry Pi V3. https://electronilab.co/tienda/camara-para-raspberry-pi-v2-8-mpx/

Tabla 17

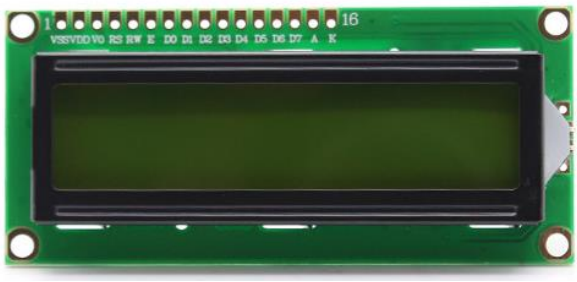
Características Electrónicas de la Cámara para Raspberry Pi V3

Parámetro	Descripción
Tipo de sensor	CMOS retroiluminado y apilado de 12 MP
Modelo	Sony IMX708
Relación señal/ruido (SNR)	Alta
Corrección de píxeles defectuosos	Corrección dinámica 2D (DPC) integrada
Autoenfoque	PDAF (Phase Detection Autofocus)
Función QBC	Re-mosaic para mejora de imagen
Modo HDR	Salida HDR de hasta 3 megapíxeles
Interfaz de salida	Serie CSI-2
Comunicación	Serial de 2 hilos (compatible con I2C rápido y rápido plus)
Control de enfoque	Serial de 2 hilos para el mecanismo de enfoque

Nota. Características principales de la cámara utilizadas para proyectos de visión y captura de imágenes en Raspberry Pi. Adaptada de de Electronilab. (2025). Cámara para Raspberry Pi V3 – 12 Megapíxeles Original. https://electronilab.co/tienda/camara-para-raspberry-pi-v2-8-mpx

Figura 25

Pantalla Display LCD 2x16



Nota. Se observa el diseño físico y la distribución de pines del display LCD. Tomada de Electronilab. (2025). Display LCD 16x2 con Backlight Amarillo. https://electronilab.co/tienda/display-lcd-16x2-con-backlight-amarillo/

Tabla 18

Características Electrónicas de la Pantalla Display LCD 2x16

Parámetro	Descripción
Voltaje de operación	5V DC
Consumo de corriente	20 mA
Formato	2 líneas x 16 caracteres
Controlador	HD44780 (compatible con KS0066U)
Interfaz	Paralela (modo 8 bits o 4 bits para ahorro de pines)
Color de fondo	Verde
Color de caracteres	Negro

Nota. Características principales del display LCD utilizadas en la integración con sistemas de microcontroladores. Adaptada de Electronilab. (2025). Display LCD 16x2 con Backlight Amarillo.https://electronilab.co/tienda/display-lcd-16x2-con-backlight-amarillo/

Figura 26

Modulo Adaptador Interfaz I2C

Nota. Se muestra el diseño físico y la disposición de pines del módulo adaptador I2C para LCD. Tomada de Ferretrónica. (2025). Módulo Adaptador Interfaz I2C para LCD 2x16 – LCD 4x20. https://www.ferretronica.com/products/modulo-adaptador-interfaz-i2c-para-lcd-2x16-lcd-4x20

Tabla 19

Características Electrónicas del Módulo Adaptador Interfaz I2C

Parámetro	Descripción
Voltaje de operación	3.3V DC
Corriente de operación	<5 mA (sin retroiluminación), <20 mA (con retroiluminación)
Interfaz de conexión	Serial SPI
Controlador	PCD8544
Temperatura de operación	-10 °C a 70 °C

Nota. La tabla presenta las características principales del módulo I2C para su integración con displays LCD. Adaptada de Ferretrónica. (2025). Módulo Adaptador Interfaz I2C para LCD 2x16 – LCD 4x20.

https://www.ferretronica.com/products/modulo-adaptador-interfaz-i2c-para-lcd-2x16-lcd-4x20

Figura 27

Motorreductor TT

Nota. Se muestra el diseño físico del motorreductor con caja reductora 6V. Tomada de Electronilab. (2025). Motorreductor con caja reductora 6V. https://electronilab.co/tienda/motorreductor-con-caja-reductora-6v-1-48/

Tabla 20

Características Electrónicas del Motorreductor TT

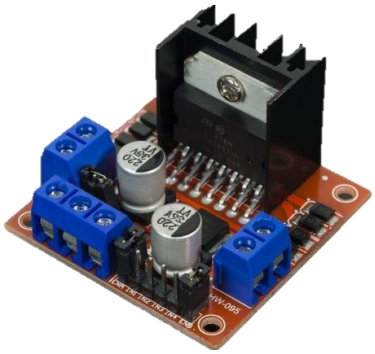
Parámetro	Descripción
Voltajes de operación	3V / 5V / 6V DC
Reducción	48:1
Velocidad sin carga	125 / 200 / 230 RPM (según voltaje)
Velocidad con carga	95 / 152 / 175 RPM
Torque de salida	0.8 / 1.0 / 1.1 kg·cm
Corriente nominal	80–100 mA (sin caja) / 110–150 mA (con carga y rueda)
Velocidad del carro (sin carga)	25.9 / 41.4 / 47.7 m/min

Parámetro	Descripción
Diámetro máximo de llanta	6.5 cm
Dimensiones	70 mm x 22 mm x 18 mm
Peso	50 g

Nota. La tabla resume las especificaciones técnicas del motorreductor. Adaptada de Electronilab. (2025). Motorreductor con caja reductora 6V. https://electronilab.co/tienda/motorreductor-con-caja-reductora-6v-1-48/

Figura 28

Módulo L298N



Nota. Se observa el diseño físico y la disposición de pines del driver L298N. Tomada de Electronilab. (2025). Driver Dual L298N para motores – Puente H. https://electronilab.co/tienda/driver-dual-para-motores-full-bridge-l298n/

Tabla 21

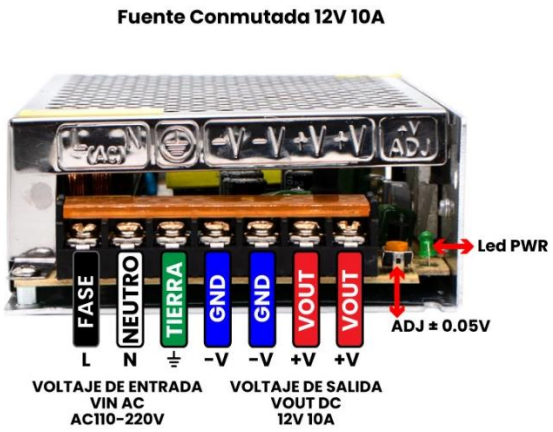
Características Electrónicas del Motorreductor TT

Parámetro	Descripción
Controlador	L298 (doble puente H)
Voltaje de alimentación	7V ~ 46V
Voltaje de control	5V
Corriente máxima por canal	2 A
Corriente de control	36 mA
Potencia de salida	25 W
Niveles de entrada (señales)	Alto: 2.3V a Vss / Bajo: -0.3V a 1.5V
Indicadores LED	Encendido, control y dirección
Temperatura de operación	-20 °C a +135 °C

Nota. La tabla muestra las principales características del driver L298N. Adaptada de Electronilab. (2025). Driver Dual L298N para motores – Puente H. https://electronilab.co/tienda/driver-dual-para-motores-full-bridge-l298n/

Figura 29

Fuente Conmutada 12 V 10 A



Nota. Se muestra el diseño físico de la fuente conmutada, destacando su estructura metálica y terminales de conexión. Tomada de UNIT Electronics. (2025, 24 octubre). Fuente conmutada 12V 10A - AC110-220V 50/60Hz. https://uelectronics.com/producto/fuente-conmutada-12v-10a/

Tabla 22

Características Electrónicas del Fuente Conmutada

Parámetro	Descripción
Modelo	S-120-12
Entrada de CA	110–220V ±15%, 50/60 Hz
Voltaje de salida	12V DC
Corriente de salida	10 A
Potencia de salida	120 W

Nota. La tabla presenta las especificaciones técnicas de la fuente conmutada alimentación estable de 12V DC. Adaptada de de UNIT Electronics. (2025, 24 octubre). Fuente conmutada 12V 10A - AC110-220V 50/60Hz. https://uelectronics.com/producto/fuente-conmutada-12v-10a/

Implementar

En la fase de implementación se integraron los módulos de software y hardware del sistema automatizado de dispensación. Se desarrollaron las APIs de gestión y validación de prescripciones en Flask y Node.js, conectadas a una base de datos MongoDB Atlas. La Raspberry Pi se configuró como unidad central de control, comunicándose con el ESP32 y el Arduino Uno mediante protocolos UDP y UART. En hardware, se ensambló el brazo robótico, los motores y sensores para la dosificación y

tapado automatizado. La integración permitió validar el flujo completo de registro, verificación y entrega del medicamento de forma segura y precisa.

Proceso de Construcción del Sistema

El sistema embebido se construyó de forma modular, iniciando con el desarrollo del software, que incluyó la interfaz web para el registro de prescripciones médicas y una API intermedia encargada de validar y actualizar el estado de los códigos QR. Por separado, se desarrollaron los módulos físicos de hardware: la construcción del robot, la base transportadora, el tubo alimentador de frascos, las tolvas y discos dosificadores, el soporte de frascos (cap holder) y el mecanismo de presión tipo slide crank en balso. Se implementó también el control del brazo robótico mediante un ESP32, la base giratoria y los dosificadores con un Arduino Uno, y la gestión central mediante una Raspberry Pi. Durante esta etapa, cada componente se construyó y probó de forma independiente.

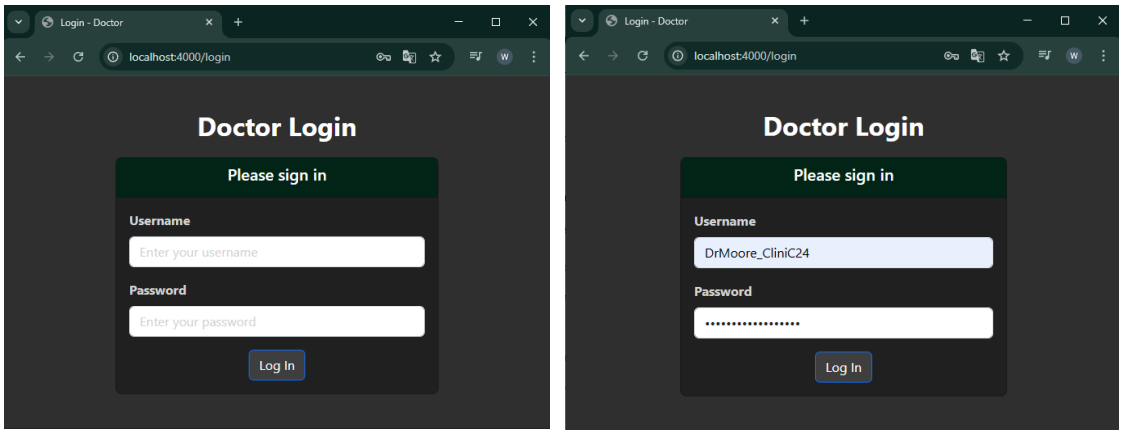
Finalmente, en la fase de resultados, se integraron todos los módulos, estableciendo la comunicación vía UDP entre la Raspberry Pi y el ESP32, y mediante UART serial entre la Raspberry Pi (como maestro) y el Arduino Uno (como esclavo). Se configuraron los protocolos necesarios para lograr un funcionamiento sincronizado, implementando así todo el sistema de dispensación automatizada.

En esta etapa, se presenta un sistema completo en el que un médico registra prescripciones a través de una API Flask con frontend, generando un código QR que se envía al paciente. En la farmacia, una Raspberry Pi con cámara Pi escanea y valida el QR mediante una API intermedia en Node.js. Si el QR es válido, se activa la dispensación automática. Al finalizar, se registra la sede, fecha y hora de la dispensación, se actualiza el estado del QR como "usado" en la base de datos y se envía

una notificación al correo del paciente. Esta información también se refleja en el frontend Flask, permitiendo al médico consultar el estado de cada prescripción.

Figura 30

Pantalla de Inicio de sesión (Login) de la Interfaz Web Desarrollada



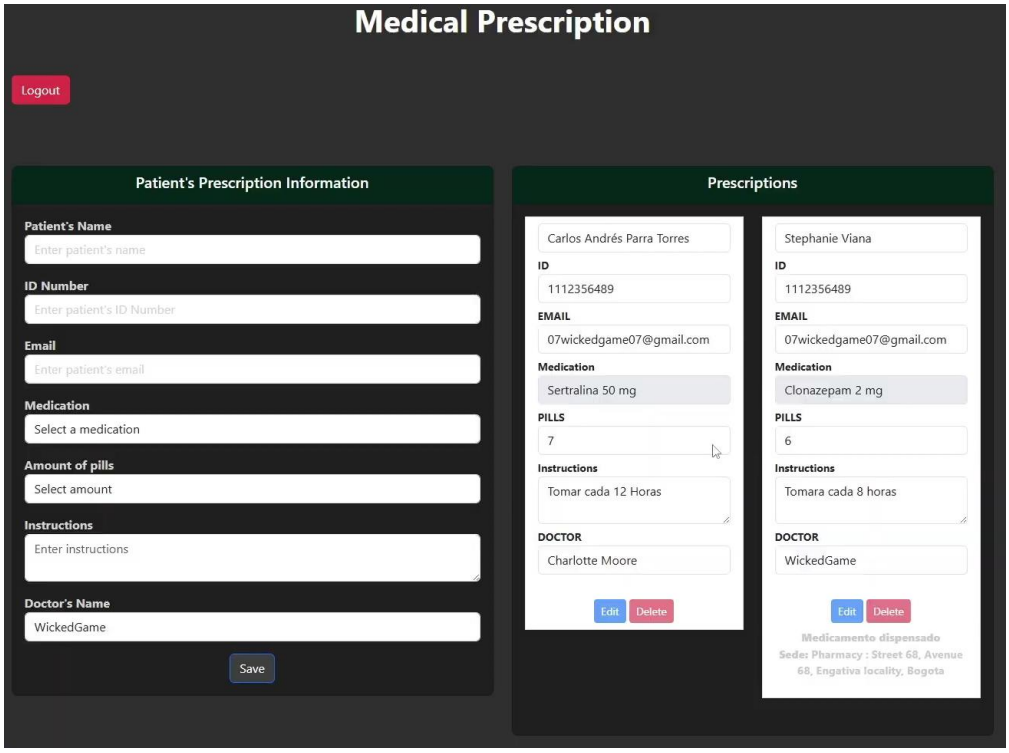
Nota. En las imágenes se observa el módulo de inicio de sesión del sistema web para médicos, implementado en la etapa de construcción del software. Este módulo permite el acceso autenticado al sistema de registro de prescripciones médicas. Elaboración propia.

En la figura 31 se observa el funcionamiento del sistema web en la etapa de implementación del software. Una vez validada la prescripción mediante el código QR y completada la dispensación automática, el sistema actualiza el estado del registro mostrando el mensaje “Medicamento dispensado”, junto con la información de la sede y la dirección donde se efectuó la entrega.

Este proceso evidencia la correcta comunicación entre la base de datos, la Raspberry Pi y el módulo de prescripción médica desarrollado en Flask, garantizando la trazabilidad y el control de cada dispensación realizada.

Figura 31

Formulario Web para Registro de Prescripciones Médicas



Nota. Interfaz desarrollada en Flask para el registro, gestión y actualización del estado de las prescripciones médicas. Elaboración propia.

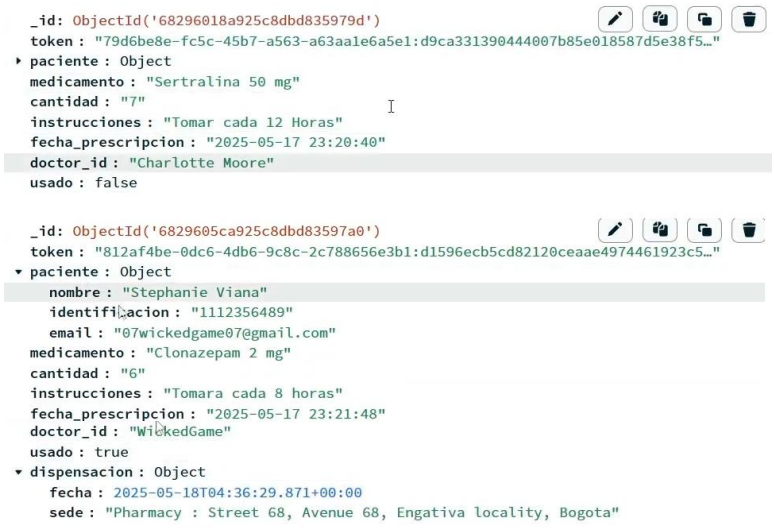
Durante el proceso de implementación del sistema, se configuró la base de datos en MongoDB Atlas, encargada de almacenar toda la información relacionada con las prescripciones médicas registradas y su respectivo seguimiento. Cada documento contiene los datos del paciente, el medicamento prescrito, la cantidad, las instrucciones de uso, la fecha de creación, el identificador del médico y el estado del token mediante el campo “usado”, el cual indica si la prescripción ya fue validada y dispensada.

Tal como se observa en la Figura 32, cuando la Raspberry Pi confirma la dispensación, el sistema actualiza este campo a true y agrega los datos de fecha, hora y sede, garantizando la trazabilidad y control de cada entrega registrada. Cuando la Raspberry Pi confirma la dispensación, el sistema actualiza este campo a true y agrega

los datos de fecha, hora y sede, garantizando la trazabilidad y control de cada entrega registrada.

Figura 32

Registros de Prescripciones en MongoDB Atlas



Nota. Visualización de los documentos almacenados en la base de datos MongoDB Atlas, donde se observa la información del paciente, el medicamento, la fecha de prescripción, el estado del token y los datos de dispensación. Elaboración propia.

Durante la fase de validación, la Raspberry Pi decodifica el código QR presentado por el paciente, el cual contiene un token único asociado a su prescripción médica. En la parte izquierda de la Figura 33 se ilustra el correo electrónico que recibe el paciente tras el registro de la prescripción, el cual incluye el código QR necesario para la dispensación. Cuando el paciente presenta este código en la farmacia, la Raspberry Pi, al escanearlo, envía el token a la API intermedia, que consulta la base de datos para verificar su existencia y el estado del campo “usado”. Si el token es válido y no ha sido utilizado (usado: false), la API retorna los datos correspondientes, como el tipo de medicamento y la cantidad, permitiendo que la Raspberry Pi active automáticamente el proceso de dispensación controlada.

Figura 33

Lectura y Verificación del Token QR por la Raspberry Pi

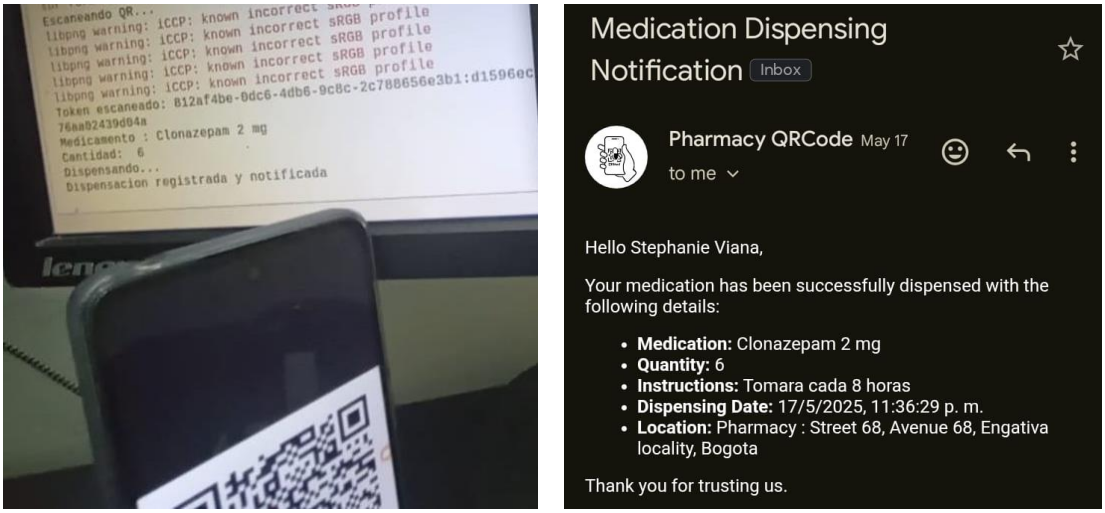
Nota. El correo muestra el código QR enviado al paciente, y la Raspberry Pi realiza su lectura y validación del token antes de iniciar la dispensación. Elaboración propia.

Durante la fase final del proceso, la Raspberry Pi ejecuta la dispensación automática del medicamento, activando los mecanismos físicos que liberan el frasco correspondiente. Una vez completada la dispensación, el sistema envía una notificación por correo electrónico al paciente, confirmando que el medicamento fue entregado exitosamente.

Tal como se observa en la Figura 34, el mensaje de confirmación incluye el nombre del medicamento, la cantidad dispensada, las instrucciones de uso, la fecha y hora del dispensado y la ubicación de la sede donde se realizó el proceso, asegurando así la trazabilidad y el registro completo del evento.

Figura 34

Registro de Dispensación y Notificación al Correo del Paciente.



Nota. Correo de confirmación generado tras la dispensación, con los datos del medicamento, cantidad, instrucciones, fecha y ubicación. Elaboración propia.

En la fase inicial del desarrollo del hardware se construyó el brazo robótico articulado, encargado de realizar la recolección y traslado de los frascos dentro del sistema de dispensación. Este módulo constituye la primera etapa de la arquitectura mecánica y electrónica del prototipo, integrando componentes estructurales y de control que permiten la manipulación precisa de los recipientes.

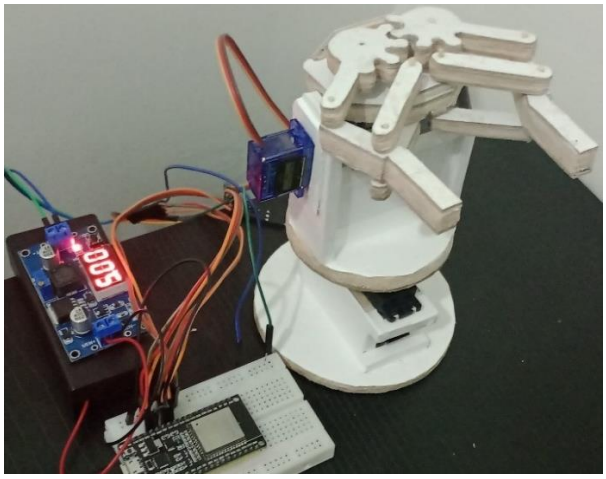
El brazo robótico, controlado por una ESP32 y coordinado desde la Raspberry Pi, ejecuta movimientos de rotación, extensión y elevación mediante servomotores de 180°, recibiendo las órdenes de activación desde el sistema central cuando se valida una prescripción.

Para garantizar un suministro eléctrico estable, se incorporó un regulador de voltaje Buck step-down LM2596, encargado de reducir la tensión de la fuente principal de 12 V a 5 V, permitiendo alimentar de manera segura tanto los servomotores como la ESP32 y evitando sobrecargas en el sistema. La Figura 35 muestra el brazo robótico

ensamblado, diseñado con tres grados de libertad (3 DOF) y un gripper mecánico acoplado a la estructura principal, responsable de sujetar y posicionar los frascos de medicamento durante el proceso de dispensación.

Figura 35

Brazo Eobot



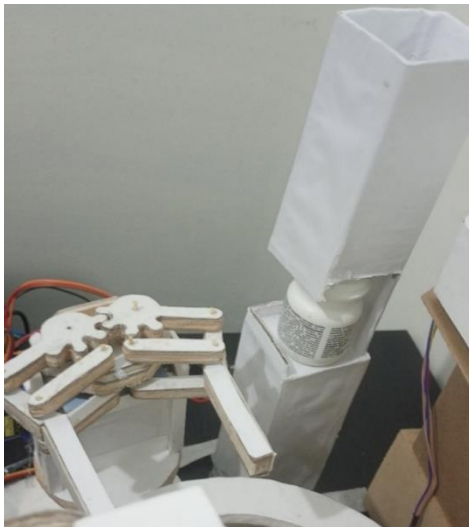
Nota. Brazo articulado de tres grados de libertad con garra mecánica, controlado por una ESP32 y alimentado mediante un regulador de voltaje Buck step-down LM2596. Elaboración propia.

El sistema cuenta además con una estructura pasiva denominada tubo alimentador de frascos, cuya función es almacenar los frascos de medicamento apilados verticalmente y guiarlos por gravedad hacia la zona de recolección. Cada vez que el brazo robótico extrae un frasco del extremo inferior del tubo y lo posiciona sobre la base transportadora, el siguiente frasco desciende automáticamente dentro del tubo, quedando listo para la siguiente operación de dispensación.

Tal como se muestra en la Figura 36, esta estructura, fabricada en material liviano y resistente, está firmemente fijada a la base principal del prototipo y forma parte integral de la arquitectura mecánica del sistema.

Figura 36

Tubo Alimentador de Frascos

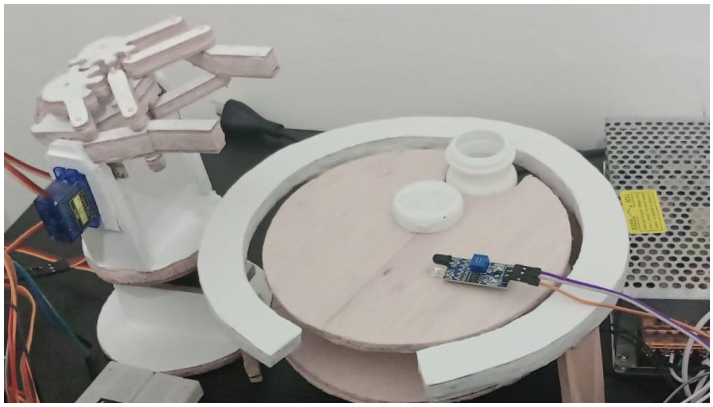


Nota. Estructura pasiva que guía los frascos hacia el brazo robótico. Elaboración propia.

En esta fase se desarrolló el control de la base giratoria, que constituye el mecanismo principal de soporte y posicionamiento de los frascos durante la dispensación automatizada. La base está controlada por un motor paso a paso NEMA 17, el cual permite realizar giros con alta precisión. Además, se incorporó un sensor infrarrojo (IR) encargado de detectar la presencia del frasco en la base giratoria, lo que permite iniciar de forma automática los cambios de estación dentro del proceso de dispensación. En la Figura 37 se muestra el montaje físico de la base giratoria, conectada al driver TB6600 y al microcontrolador Arduino Uno, encargado de ejecutar las señales de control de giro y recibir la señal del sensor IR para sincronizar los movimientos del sistema.

Figura 37

Control de la base giratoria

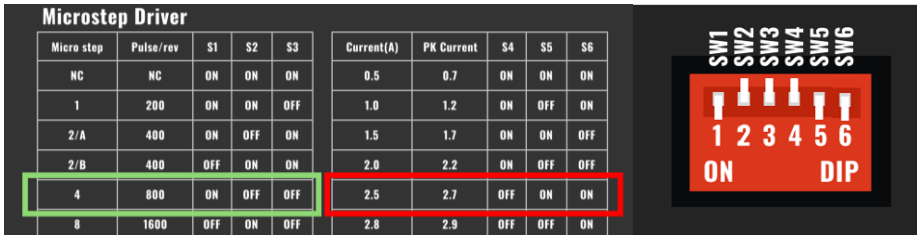


Nota. Se observa la conexión del sistema mecánico y electrónico de la base giratoria, implementado con componentes de control paso a paso. Elaboración propia.

El driver TB6600 es el componente encargado de regular la corriente y el número de pasos del motor. La configuración de sus interruptores DIP permite establecer los micro pasos y la corriente de salida para ajustar el rendimiento del motor según las necesidades del sistema. En la Figura 38 se presenta la tabla de configuración DIP utilizada para el control del motor paso a paso NEMA 17, donde se especifica la relación entre micro pasos y corriente ajustada.

Figura 38

Configuración DIP del driver TB6600.

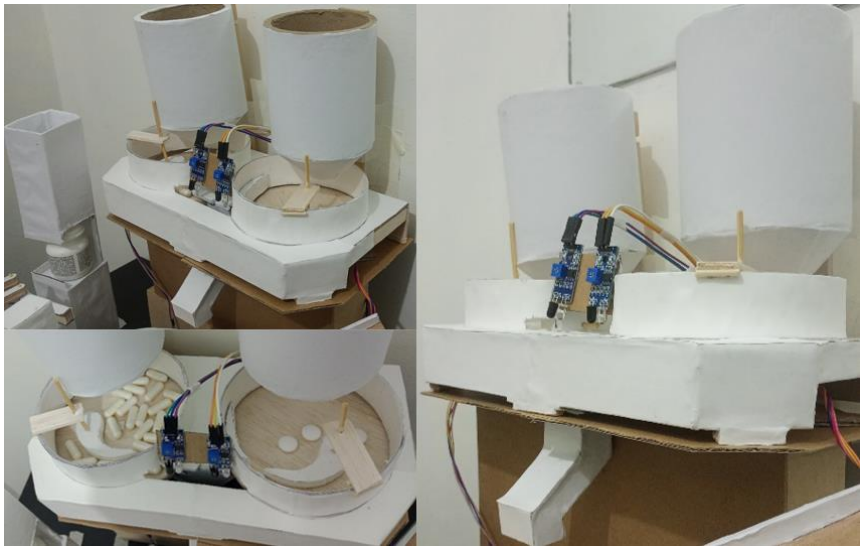


Nota. Se configuró el driver TB6600 con un micro paso de 4 (800 pulsos/rev) y una corriente de 2.5 A (pico 2.7 A) para optimizar el torque y la precisión del motor NEMA 17 en la base giratoria. Elaboración propia.

El sistema de dosificación fue construido en balso, material liviano y fácil de mecanizar, ideal para el prototipado estructural. Su diseño incluye dos tolvas independientes destinadas al almacenamiento de cápsulas y tabletas, conectadas a discos dosificadores que liberan una unidad por ciclo. Estos discos son accionados por motores paso a paso 28BYJ-48, controlados mediante drivers ULN2003, garantizando precisión en la entrega. La base cuenta con un diseño en espiral que guía las pastillas hacia un ducto de salida único, evitando atascos y asegurando un flujo constante hacia el frasco receptor. Además, el sistema incorpora sensores infrarrojos (IR) que detectan el paso de cada pastilla, permitiendo el conteo y deteniendo el proceso al alcanzar la cantidad prescrita. Toda la operación es coordinada por un microcontrolador Arduino, bajo la supervisión de la Raspberry Pi, que sincroniza esta fase dentro del flujo general del dispensador automatizado.

Figura 39

Sistema de Dosificación en Balso con Tolvas, Sensores y Discos Giratorios



Nota. Estructura construida en balso que integra tolvas, discos dosificadores, sensores infrarrojos y base en espiral para guiar las pastillas al ducto de salida. Elaboración propia.

El sistema de tapado fue construido en balso y combina dos mecanismos que trabajan de forma sincronizada para asegurar el cierre del frasco. En la Figura 40 se muestra el Cap Holder, una estructura pasiva que almacena varias tapas apiladas y las libera de manera automática. Cuando la base giratoria traslada el frasco hasta la estación de tapado, la boca del frasco toma una tapa directamente del Cap Holder y la posiciona sobre sí misma durante el recorrido. Posteriormente, en la Figura 36 se observa el brazo de presión tipo crank-slide, accionado por un motorreductor TT controlado por la Raspberry Pi mediante un driver L298N. Este mecanismo convierte el movimiento rotacional en un desplazamiento lineal descendente, aplicando presión para ajustar y sellar la tapa de forma precisa al llegar a la estación final.

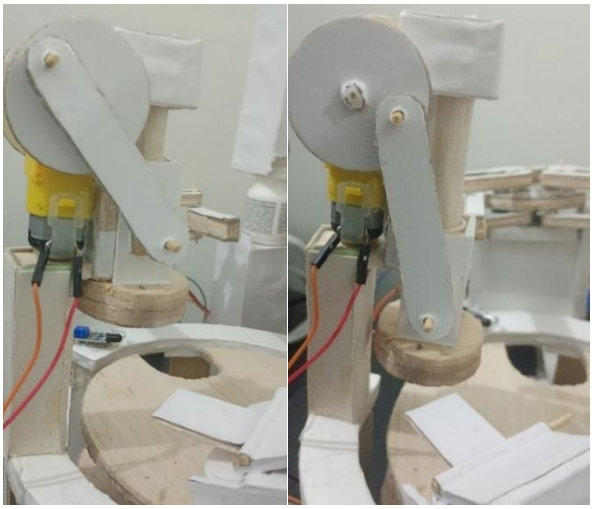
Figura 40

Estructura de Cap Holder

Nota. Elemento pasivo construido en balso que sostiene la tapa durante el proceso de sellado, permitiendo el posicionamiento del frasco bajo la tapa. Elaboración propia.

Figura 41

Brazo de Presión Tipo Crank-Slide.



Nota. Mecanismo en balso accionado por un motorreductor TT que transforma el giro en movimiento lineal para sellar el frasco. Elaboración propia.

Operar

La fase de operar corresponde al momento en que el sistema automatizado entra en funcionamiento real, permitiendo evaluar su desempeño y eficiencia en la dispensación controlada de medicamentos. En esta etapa se pone a prueba la interacción entre los módulos de software y hardware, comprobando la estabilidad de las comunicaciones, la precisión de la dosificación y la validez del proceso completo desde la lectura del código QR hasta la entrega del fármaco. Su objetivo es garantizar que el prototipo cumpla con los criterios de seguridad, trazabilidad y confiabilidad establecidos durante el diseño e implementación.

Diseño de un Sistema Seguro para Registro de Prescripciones

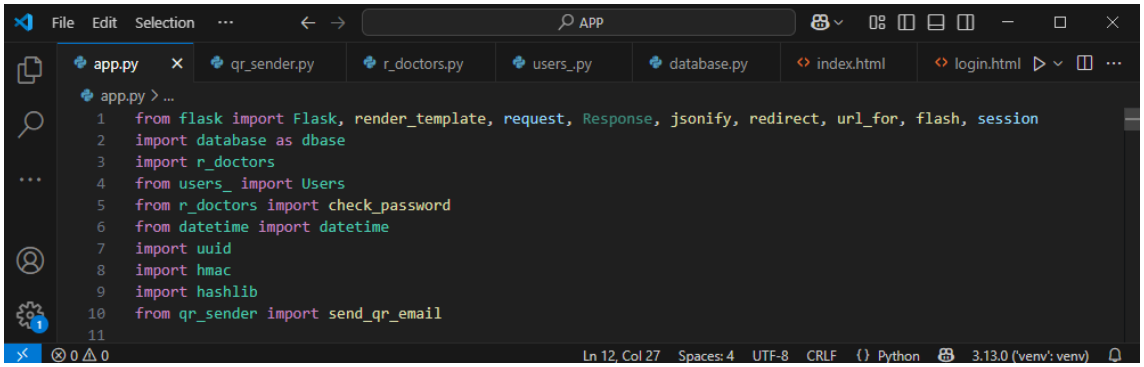
El sistema fue diseñado con una arquitectura modular, basada en una aplicación web cliente-servidor que permite a los médicos registrados ingresar prescripciones

médicas, almacenarlas de forma segura y enviar un código QR al paciente como medio de autenticación para la futura dispensación del medicamento.

El diseño contempla un enfoque orientado a la seguridad, usabilidad y escalabilidad. Cada módulo está claramente delimitado según su responsabilidad: autenticación de usuarios, manejo de datos, generación de tokens, envío de correos electrónicos y diseño de interfaces.

Figura 42

Fragmento de App.Py del Sistema de Gestión de Recetas Médicas



Nota. En el fragmento se observan las importaciones de módulos y librerías necesarias para el funcionamiento del sistema. Elaboración propia.

La figura 42 muestra las líneas iniciales del archivo app.py, el cual funciona como el núcleo del backend del sistema web desarrollado con el microframework Flask. Este archivo coordina la lógica del sistema para el registro, validación y envío de prescripciones médicas.

Importaciones desde Flask:

Se integran funciones clave como Flask, render_template, request, Response, jsonify, redirect, url_for, flash y session, que permiten manejar la creación de rutas, solicitudes HTTP, gestión de sesiones y renderizado de plantillas HTML.

Módulos internos del sistema:

database: Manejo de conexión y operaciones con la base de datos.

r_doctors: Contiene funciones para validar contraseñas y gestionar los registros de los doctores.

users: Incluye la clase Users, la cual modela una prescripción médica.

qr_sender: Se encarga de generar códigos QR a partir del token de una receta y enviarlos por correo electrónico al paciente, como parte del proceso de validación de prescripciones médicas.

Librerías estándar:

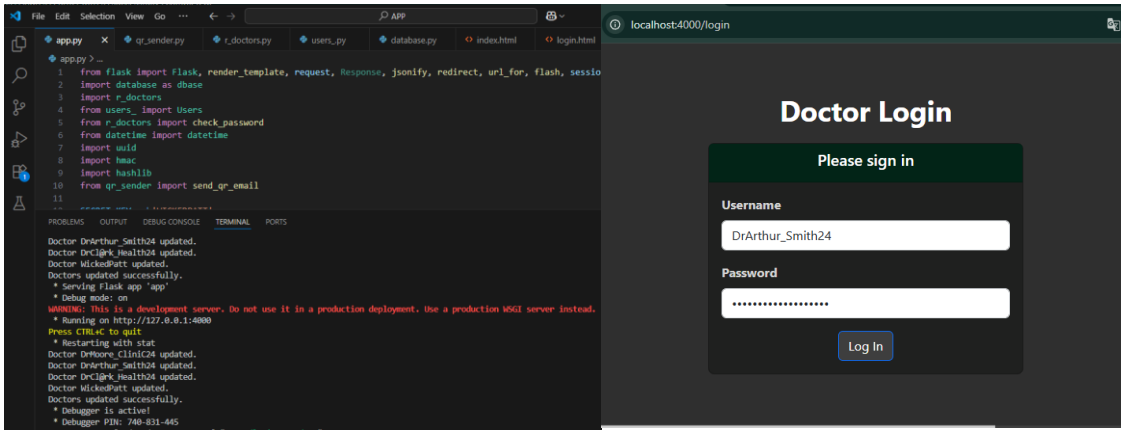
Se incluyen módulos como datetime, uuid, hmac y hashlib, utilizados para la generación de tokens únicos, manejo de fechas y fortalecimiento de la seguridad del sistema mediante funciones criptográficas.

Además, en las pestañas superiores del entorno de desarrollo (Visual Studio Code) se observan archivos HTML como index.html y login.html, que corresponden a las interfaces gráficas del sistema. Estas plantillas son renderizadas desde app.py mediante la función render_template, permitiendo una interacción amigable entre el usuario (un médico) y el sistema web.

Este archivo representa el punto de integración entre la lógica del servidor y la interfaz visual del sistema, gestionando tanto las peticiones del usuario como la presentación dinámica de las páginas HTML.

Figura 43

Interfaz Web Login



Nota. En la imagen se muestra la consola de ejecución del servidor Flask junto con la interfaz de autenticación del usuario tipo doctor. Elaboración propia.

En la interfaz web:

- ✓ Login de médicos (login.html)
- ✓ Autenticación protegida por contraseña.
- ✓ Validación de campos obligatorios.
- ✓ Muestra mensajes de error en caso de credenciales incorrectas.

En la figura 43 muestran dos componentes fundamentales del sistema web para la gestión de prescripciones médicas:

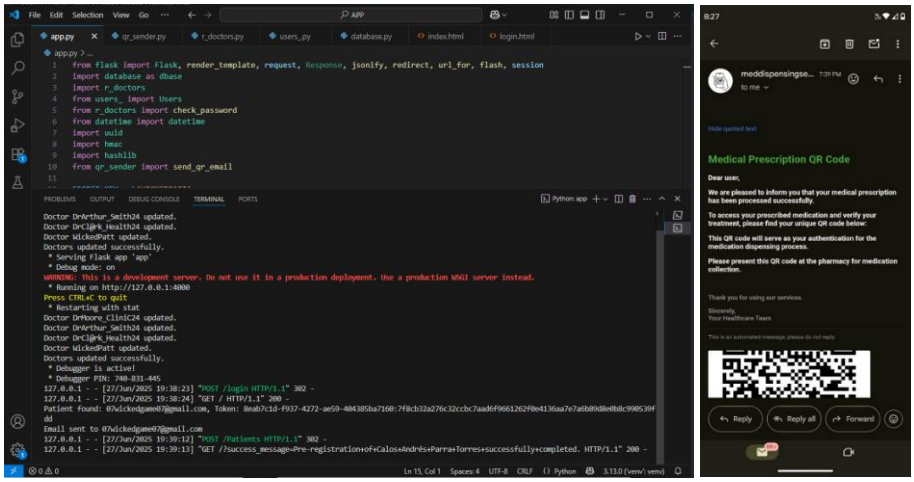
Parte superior (terminal en Visual Studio Code): se observa la ejecución del archivo app.py, donde se inicializa el servidor y se actualizan los registros de doctores en la base de datos. Además, se registran solicitudes HTTP entrantes, incluyendo una petición a la ruta /login.

Parte inferior (interfaz web): se presenta la página correspondiente a la ruta /login, la cual carga correctamente en el navegador y despliega un formulario de autenticación para doctores. En él, el usuario DrArthur_Smith24 ha ingresado su nombre de usuario y contraseña, listo para iniciar sesión.

La terminal refleja las acciones del servidor backend, mientras que la interfaz muestra el frontend con el que interactúan los usuarios médicos. Juntas evidencian el flujo inicial del sistema, desde la ejecución del servidor hasta la carga de la pantalla de inicio de sesión.

Figura 44

Acciones del Servidor Backend y Envío de la Prescripción al Correo Electrónico



Nota. Desde el backend, la prescripción se almacena exitosamente en la base de datos y, de forma simultánea, el sistema envía al correo electrónico del paciente un mensaje que incluye el código QR generado. Elaboración propia.

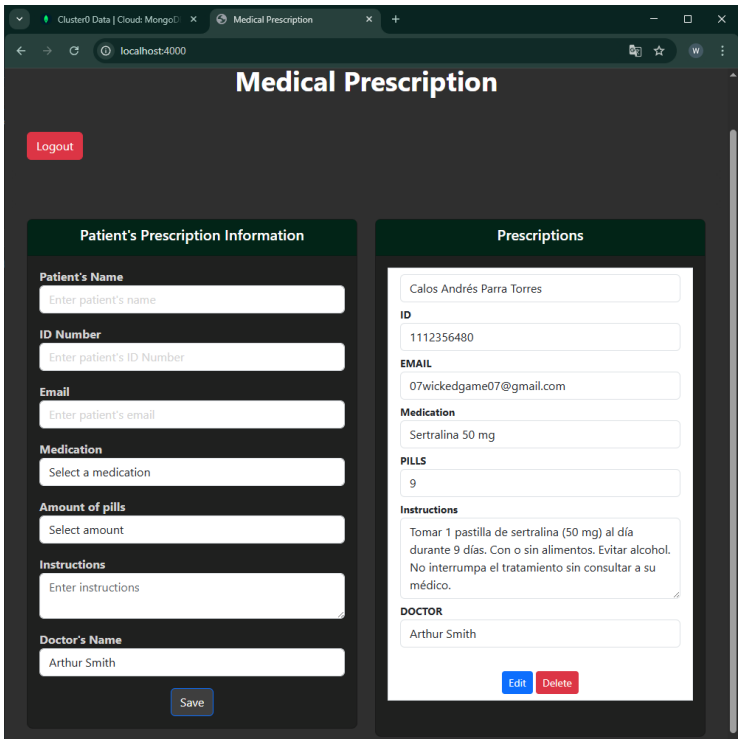
En la figura 44 muestra cómo se registran varias peticiones HTTP que reflejan las acciones realizadas en la interfaz:

- La línea POST /login indica que un doctor ha enviado el formulario de inicio de sesión.
- La respuesta GET / muestra que, tras autenticarse, el sistema redirige al usuario a la página principal del sistema de prescripciones.

- El mensaje Patient found: ... confirma que se ha recuperado un paciente desde la base de datos y que se ha generado un token, el cual fue enviado por correo electrónico. Esto está directamente relacionado con la acción de guardar una receta.
- La línea POST /Patients representa el envío del formulario de registro de prescripción médica.
- Finalmente, la petición GET /?success_message=... refleja la recarga de la interfaz con un mensaje de éxito, mostrando que el paciente fue registrado correctamente.

Figura 45

Interfaz Web del Formulario de Registro de Prescripciones



Nota. Desde esta interfaz, el médico puede registrar los datos del paciente, el medicamento prescrito y las instrucciones de uso, los cuales se almacenan en la base de datos del sistema. Elaboración propia.

En la interfaz web:

- ✓ Formulario de registro de prescripciones (index.html)

- ✓ Interfaz limpia, responsiva y oscura.
- ✓ Permite seleccionar medicamentos, ingresar cantidad, instrucciones y correo del paciente.
- ✓ Solo visible para médicos autenticados.

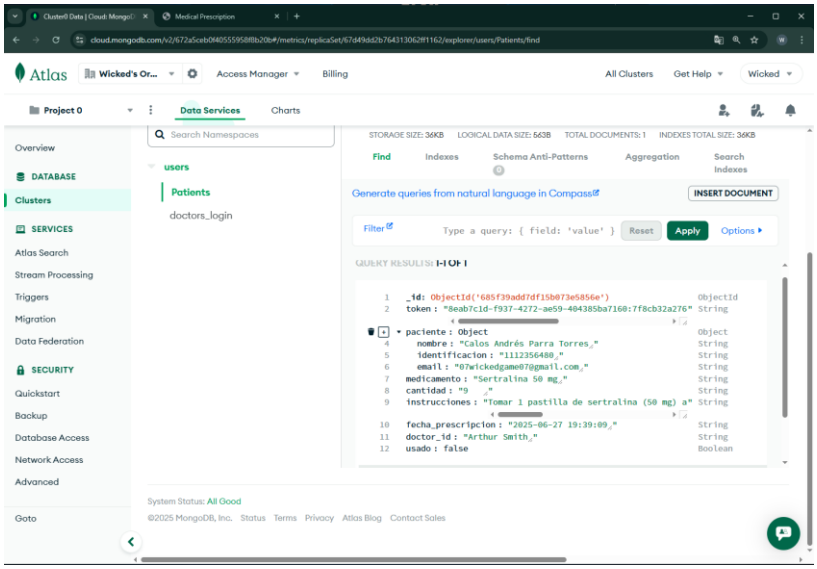
En la Figura 45 se observa la interfaz web del formulario de registro de prescripciones. En la parte izquierda se muestra el formulario vacío, listo para ser diligenciado con la información del paciente, el medicamento y las instrucciones médicas. En la parte derecha se presenta la misma interfaz, pero con la sección de prescripciones cargada y mostrando un registro completo, lo cual coincide con la actividad de registro (POST /Patients) evidenciada en la terminal del sistema.

En el **Apéndice A** se presenta un video demostrativo del funcionamiento del sistema web, en el cual se evidencia el proceso de registro, almacenamiento y envío de la prescripción médica al correo electrónico del paciente.

MongoDB Atlas: Prescripciones y Tokens. Se utilizó una base de datos NoSQL en la nube, para almacenar la información de los médicos y las prescripciones. Su naturaleza flexible y orientada a documentos permitió una estructura ágil y adaptable a los cambios. La conexión se establece de forma segura utilizando certificados digitales con la librería certifi.

Figura 46

Datos de Prescripción Almacenados en Mongo Atlas



Nota. En la imagen se observa el documento almacenado en la base de datos, donde se registra la información del paciente, el medicamento prescrito, las instrucciones de uso, el identificador del médico, el token generado para la validación y el campo “usado: false”, que indica que la prescripción aún no ha sido utilizada. Elaboración propia.

La figura 46 muestra la estructura de un documento JSON correspondiente a una prescripción médica, almacenado en la colección Patients de la base de datos MongoDB. Este registro se genera automáticamente al momento de completar el formulario en la interfaz web y contiene toda la información necesaria para validar y dispensar el medicamento.

En el documento se observa:

- Un identificador único (_id) y un token generado que sirve como base para el código QR enviado al paciente.
- Un subdocumento paciente que almacena el nombre, número de identificación y correo electrónico del usuario.

- Información detallada sobre el medicamento prescrito (medicamento, cantidad e instrucciones).
- El campo doctor_id, que vincula la receta con el profesional que la emitió.
- La fecha de prescripción, registrada automáticamente por el sistema.
- El campo usado, que indica si el código QR ha sido utilizado en la farmacia.

Este documento refleja cómo el sistema centraliza la información de cada receta en una estructura organizada, permitiendo trazabilidad, validación con QR y seguimiento de la dispensación en farmacia.

Aplicación del Modelo CRUD en la Web. En el desarrollo del sistema se implementó completamente el modelo CRUD (Create, Read, Update, Delete), que representa las operaciones fundamentales para la gestión de datos. Estas operaciones fueron aplicadas sobre la entidad "prescripción médica", permitiendo que los doctores puedan crear, visualizar, modificar y eliminar las recetas de los pacientes a través de una interfaz web. La lógica del CRUD fue implementada mediante rutas HTTP en el framework Flask, conectadas a una base de datos MongoDB. A continuación, se describen las rutas y funciones principales asociadas a cada operación.

Tabla 23

Detalles Técnicos de la Aplicación CRUD: Registro de Prescripciones

Operación	Método HTTP	Ruta	Función en Flask	Descripción
Create	POST	/Patients	add_Patient()	Registra una nueva prescripción médica con los datos del paciente.

Read	GET	/	home()	Lista todas las prescripciones almacenadas en la base de datos.
Update	GET/POST	/edit/<string:P_NAME>	edit(P_NAME)	Permite editar una receta previamente registrada.
Delete	GET	/delete/<string:P_NAME>	delete(P_NAME)	Elimina una receta si cumple las condiciones de autorización.

Nota. La tabla presenta las principales operaciones implementadas en el backend del sistema, donde se especifican los métodos HTTP, las rutas definidas y las funciones correspondientes en Flask que permiten crear, consultar, actualizar y eliminar prescripciones médicas.

Además de las operaciones CRUD, el sistema implementa una funcionalidad de login que permite autenticar a los doctores registrados en la base de datos. Esta operación se realizó a través de un formulario de acceso protegido, y fue fundamental para:

- Garantizar la seguridad de las prescripciones médicas.
- Asociar cada receta a un médico específico.
- Restringir la edición y eliminación de recetas solo al médico que las creó.

Tabla 24

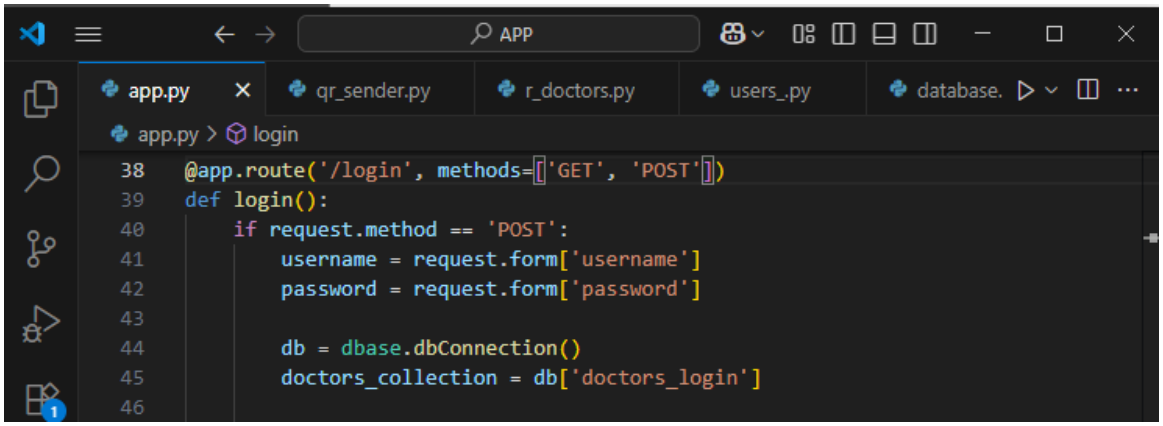
Detalles Técnicos de la Autenticación

Función	Método HTTP	Ruta	Descripción
login()	GET / POST	/login	Verifica las credenciales del doctor mediante bcrypt y habilita la sesión.
logout()	GET	/logout	Cierra la sesión activa del doctor.

Nota. La tabla muestra las funciones del módulo de autenticación que permiten el inicio y cierre de sesión del doctor en el sistema.

Figura 47

Fragmento del Manejo de Inicio de Sesión en la Aplicación



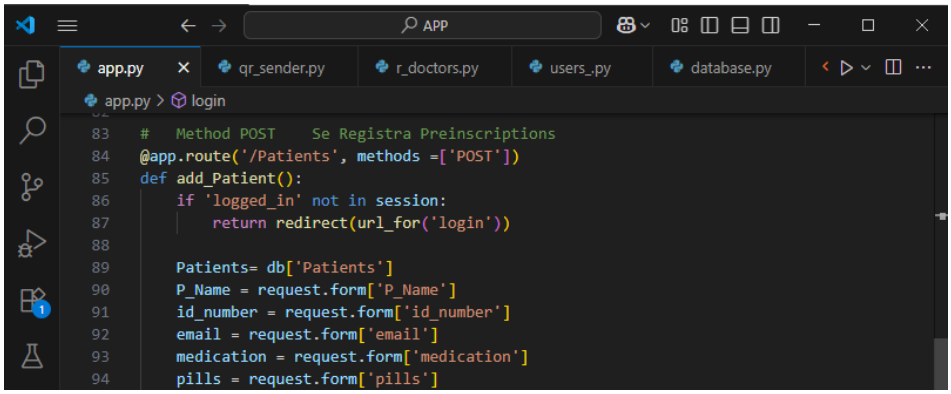
Nota. El fragmento de código muestra la definición de la ruta /login en Flask, que permite manejar las solicitudes GET y POST para la autenticación de usuarios. Elaboración propia.

En la Figura 47 se observa el fragmento de código encargado del manejo del inicio de sesión dentro de la aplicación web. En este módulo se define la ruta /login, la

cual acepta los métodos GET y POST para recibir los datos ingresados por el usuario, específicamente el nombre de usuario y la contraseña. Una vez obtenidos, el sistema establece la conexión con la base de datos y verifica las credenciales en la colección correspondiente, garantizando el acceso seguro al sistema. Este fragmento implementa la autenticación de médicos mediante usuario y contraseña cifrada. Si los datos coinciden, se inicia sesión segura con session.

Figura 48

Fragmento del Registro de Pacientes desde el Perfil del Doctor

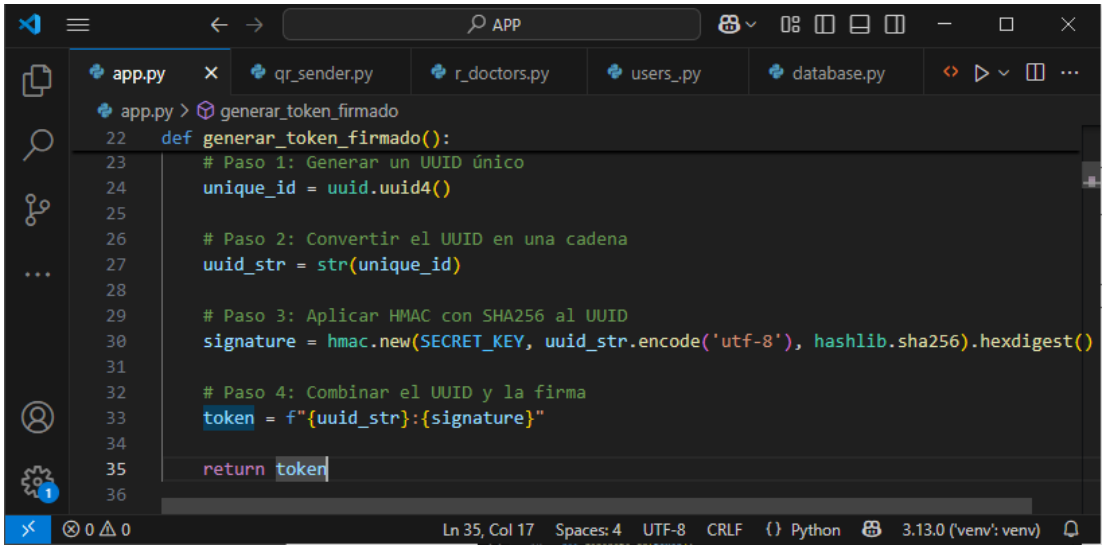


Nota. El código muestra la ruta /Patients, configurada para manejar solicitudes POST que permiten registrar nuevas prescripciones médicas en el sistema. Elaboración propia.

En la Figura 48 se presenta el fragmento de código responsable del registro de pacientes y la creación de nuevas prescripciones médicas desde el perfil del doctor. Este proceso se ejecuta mediante el método POST, que recibe los datos del formulario (nombre, identificación, correo electrónico, medicamento y cantidad de píldoras) y los almacena en la colección correspondiente de la base de datos. Además, el flujo de ejecución integra validaciones de sesión, conexión con el modelo de datos y posterior generación del código QR asociado a la prescripción.

Figura 49

Fragmento de Generación del Token



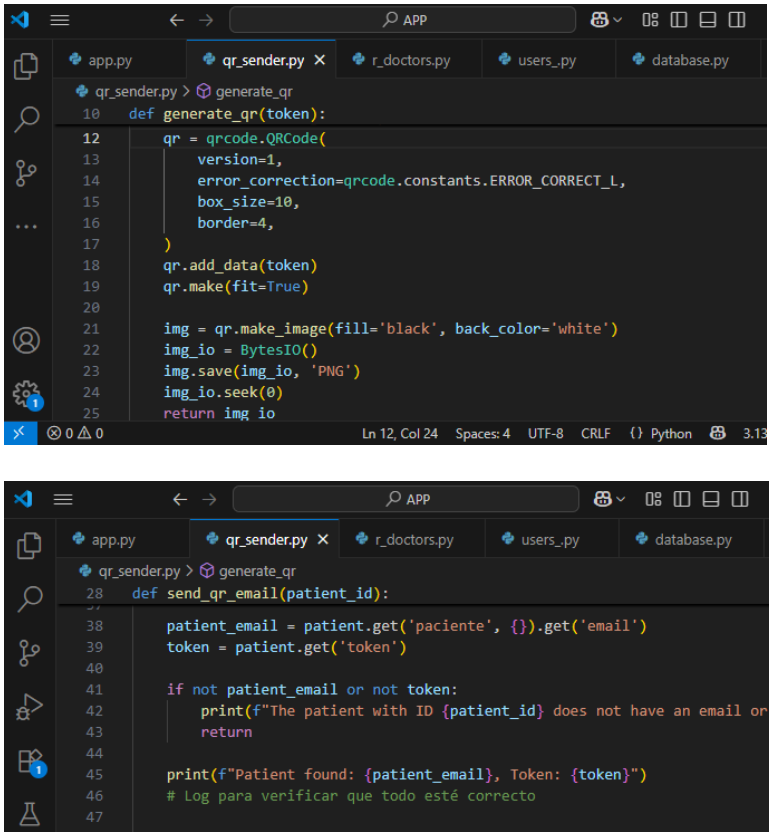
Nota. El código muestra la función generar_token_firmado(), encargada de crear un token único y seguro mediante el uso de un identificador UUID y una firma criptográfica HMAC con el algoritmo SHA256. Elaboración propia.

En la Figura 49 se observa el proceso de generación del token utilizado para la validación de las prescripciones médicas. Esta función combina un identificador único (UUID) con una firma generada mediante HMAC-SHA256, garantizando la integridad y autenticidad del mensaje. El token resultante se envía junto con el código QR al correo electrónico del paciente, asegurando que cada enlace o acceso sea irrepetible y verificable.

.

Figura 50

Fragmento de Generación de QR con Token y Envío por Correo

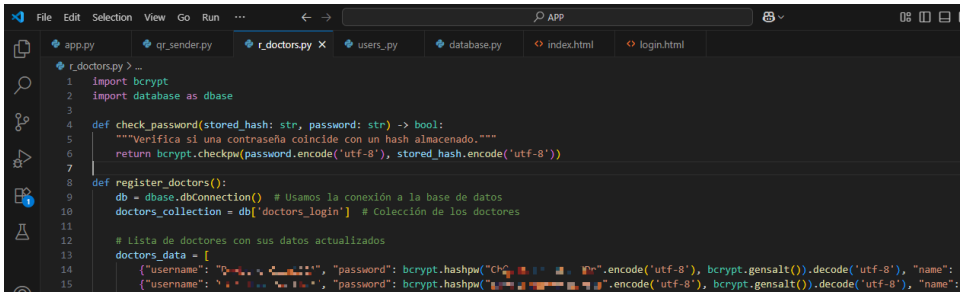


Nota. Muestra la creación del código QR y su envío al correo del paciente junto con el token de verificación. Elaboración propia.

Envío del Token QR por Email. En la Figura 50, el fragmento define la función generate_qr(token), que crea un código QR a partir del token único generado previamente. Este código se configura con parámetros de corrección de errores y diseño visual, y finalmente se guarda en memoria como una imagen. En la parte inferior, la función send_qr_email(patient_id) obtiene el correo del paciente y el token correspondiente desde la base de datos; posteriormente, verifica su validez antes de realizar el envío. Este procedimiento garantiza que cada paciente reciba su prescripción de manera segura y personalizada a través de su correo electrónico.

Figura 51

Fragmento de Registro y Validación de Doctores en el Sistema

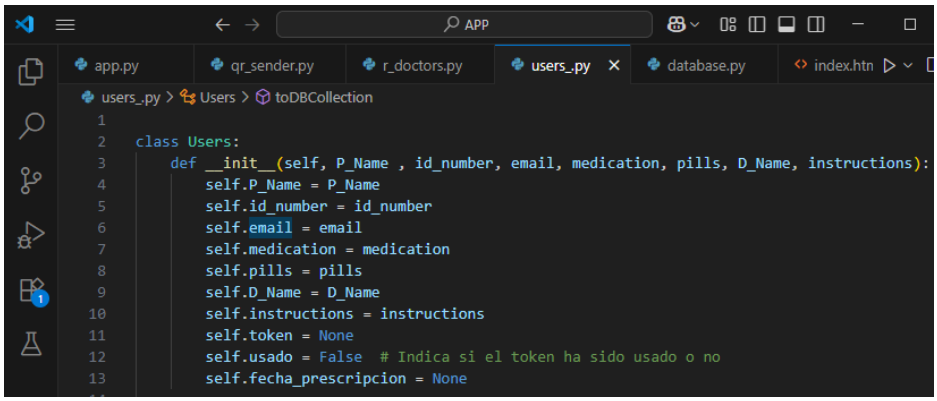


Nota. Parte esencial del manejo de autenticación en aplicaciones que requieren control de acceso y validación de credenciales. Elaboración propia.

En la figura 51 se muestra la implementación de un módulo en Python para el registro y validación de doctores dentro de un sistema. Se utiliza la librería bcrypt para cifrar y verificar contraseñas de forma segura, garantizando la protección de los datos de acceso. Además, se define una lista con información de doctores registrados, simulando una base de datos que posteriormente puede ser utilizada para autenticar usuarios mediante la función check_password.

Figura 52

Fragmento del Modelo de Datos del Paciente y Prescripción Médica

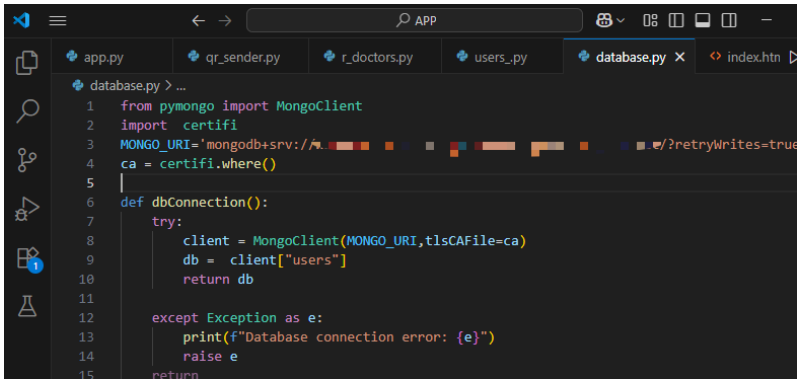


Nota. Fragmento de código Python correspondiente al modelo de datos del paciente y prescripción médica. Elaboración propia.

En la Figura 52 se presenta un fragmento de código en Python que define la clase Users, la cual modela los datos de un paciente dentro de un sistema de prescripción médica. Esta clase incluye atributos como el nombre, número de identificación, correo electrónico, medicación, dosis e instrucciones, permitiendo estructurar la información necesaria para el manejo y seguimiento de tratamientos médicos

Figura 53

Fragmento de Conexión a MongoDB Atlas



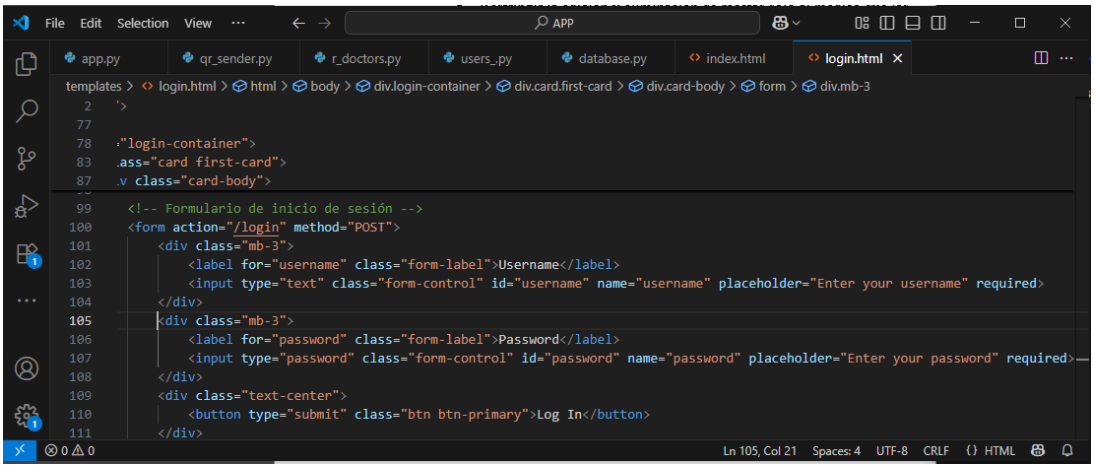
Nota. Fragmento de código en Python que muestra la conexión a la base de datos MongoDB Atlas mediante la librería pymongo. Elaboración propia.

En este fragmento de código se implementa la función encargada de establecer la conexión entre la aplicación y la base de datos MongoDB Atlas utilizando la librería pymongo. Se define la variable MONGO_URI, que contiene la dirección de conexión al clúster remoto, y se emplea la función MongoClient para crear la conexión de manera segura mediante un certificado SSL. Además, el bloque try-except permite manejar posibles errores de conexión, mostrando un mensaje en caso de fallo. Este

procedimiento es fundamental para garantizar el acceso estable y seguro a los datos almacenados en la base de datos del sistema.

Figura 54

Interfaz de Inicio de Sesión Protegida para Doctores

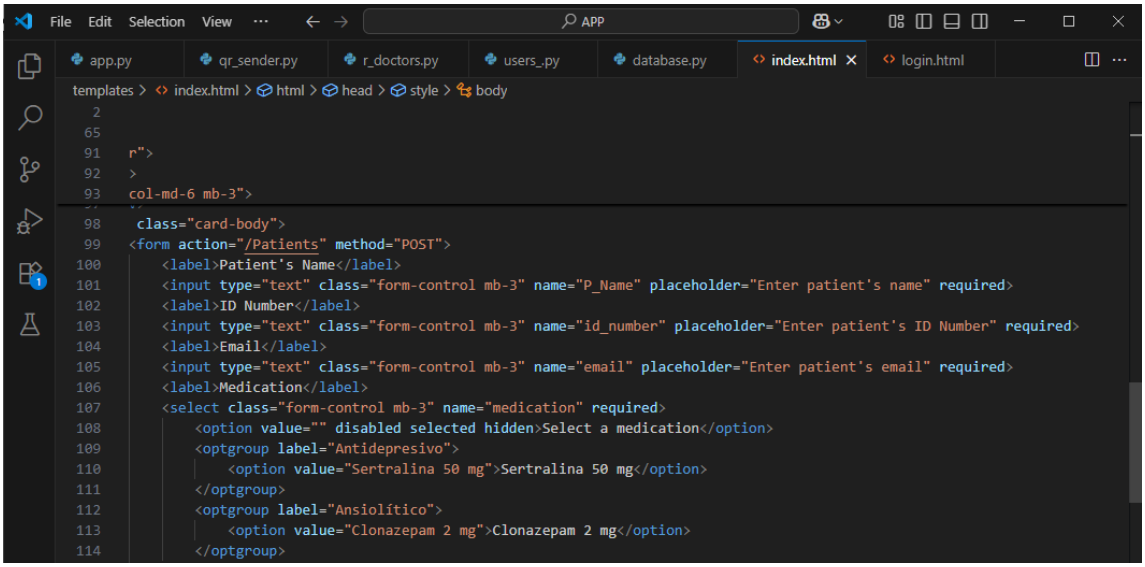


Nota. Login.html (interfaz de inicio de sesión). Vista protegida para el ingreso exclusivo de doctores. Refuerza control de acceso. Elaboración propia.

En la figura 54 se presenta la estructura HTML de la interfaz de inicio de sesión diseñada para los doctores dentro del sistema. El formulario incluye campos para ingresar el nombre de usuario y la contraseña, ambos requeridos para el acceso. Además, utiliza etiquetas y clases de estilo que permiten una presentación organizada y compatible con frameworks como Bootstrap, lo que mejora la usabilidad y el diseño visual de la página. Este formulario constituye la capa de entrada de datos que se comunica con el backend para validar las credenciales y garantizar un acceso seguro a la plataforma.

Figura 55

Interfaz Principal del Sistema: Vista CRUD para Gestión de Recetas



Nota. Fragmento de código HTML que muestra el formulario para el registro de pacientes y selección de medicación. Elaboración propia.

En la figura 55 presenta la estructura del formulario HTML utilizado para registrar los datos de los pacientes, incluyendo nombre, número de identificación y correo electrónico. Además, incorpora un menú desplegable que permite seleccionar el medicamento correspondiente al tratamiento del paciente. Cada campo cuenta con validaciones básicas mediante el atributo required, lo que garantiza que la información necesaria sea ingresada antes de enviar el formulario. Este componente constituye una parte esencial del sistema, al facilitar la gestión de la información de manera estructurada y accesible.

Realización de un Sistema de Validación Mediante Códigos QR

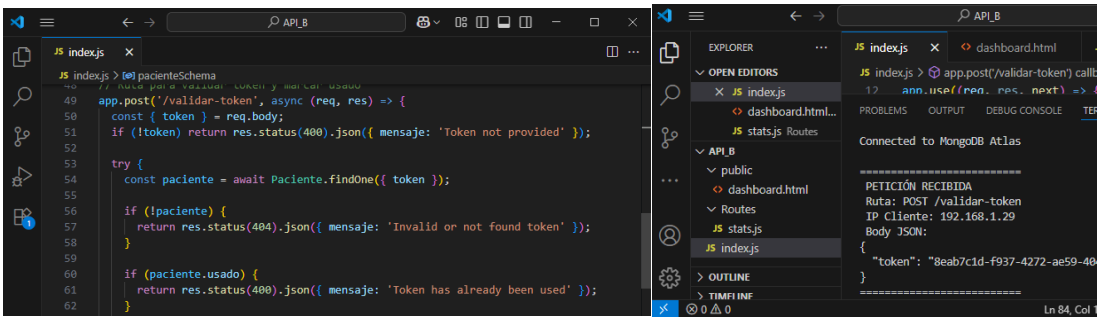
Este apartado corresponde al segundo objetivo específico del proyecto, el cual busca implementar un mecanismo seguro para validar recetas médicas mediante códigos QR, permitiendo así que un paciente acceda a su medicamento únicamente si presenta

un token válido y no utilizado. Este proceso se logra mediante la integración de tres componentes principales:

API REST en Node.js (Middleware). Se desarrolló un servidor intermedio en Node.js que cumple dos funciones clave.La API se conecta a MongoDB Atlas, base de datos que almacena los tokens, prescripciones e información del paciente.

Figura 56

Verificación en Terminal del Backend: Conexión, Solicitudes y Tokens



Nota. Fragmento de código en JavaScript que valida tokens de acceso y verifica la existencia de pacientes en la base de datos. Elaboración propia.

El fragmento de código mostrado corresponde a una función en JavaScript, implementada en un entorno Node.js, que gestiona la validación de tokens y la verificación de pacientes registrados.

Mediante una solicitud POST, el sistema recibe un token de autenticación y lo comprueba antes de permitir el acceso, la consulta o la modificación de datos. Si el token no es válido o ya ha sido utilizado, se devuelve un mensaje de error y un código de estado HTTP correspondiente. Este procedimiento es esencial para garantizar la seguridad de las transacciones y evitar accesos no autorizados dentro del sistema.

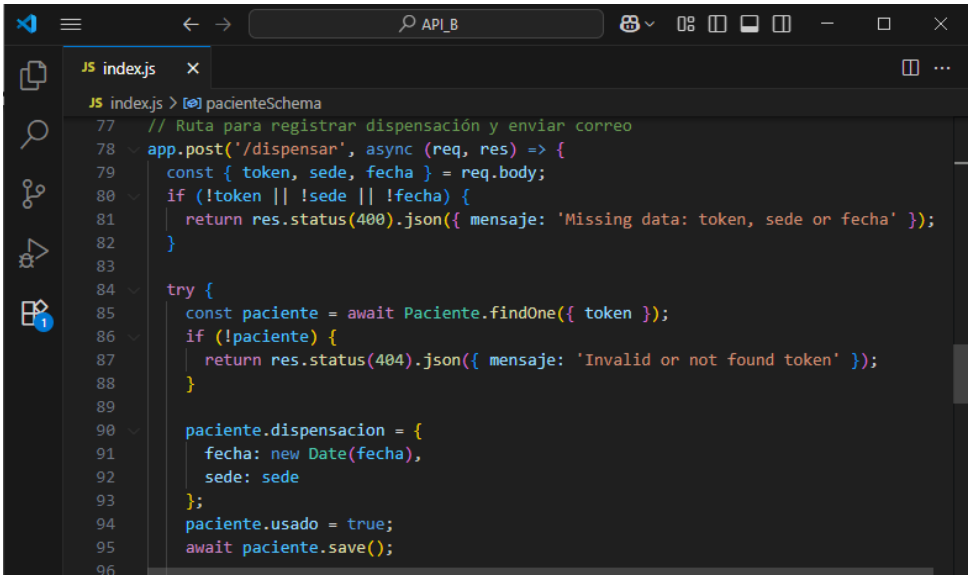
Lo que se muestra en la terminal del backend actúa como una forma de verificación visual que confirma que la API intermedia está funcionando correctamente. Al imprimir mensajes como "Conectado a MongoDB Atlas", la IP del cliente, la ruta

accedida (/validar-token) y el contenido del token recibido, se puede validar que la Raspberry Pi está enviando correctamente la solicitud, que la API la está recibiendo y que se encuentra conectada a la base de datos.

Estos mensajes sirven como evidencia de que la comunicación entre los componentes Raspberry Pi, API y la base de datos se está realizando de forma adecuada.

Figura 57

Código para actualizar token a "usado" y registrar dispensación



Nota. Fragmento de código en JavaScript que actualiza el estado del token y registra la dispensación del medicamento. *Fuente:* elaboración propia.

El fragmento de código de la figura 57 corresponde a una función desarrollada en JavaScript dentro de un entorno Node.js, encargada de actualizar el estado del token como "usado" y registrar la información de la dispensación del medicamento. A través de una solicitud POST, el sistema recibe los datos del token, la sede y la fecha, validando su existencia antes de realizar cualquier cambio. Si alguno de los datos está ausente o el token no es válido, se genera un mensaje de error con el código de estado

correspondiente. Este proceso garantiza la trazabilidad y control de las dispensaciones realizadas, fortaleciendo la seguridad y confiabilidad del sistema.

Interacción con la Raspberry Pi (Cliente Físico). La Raspberry Pi ejecuta un script en Python que realiza las siguientes tareas:

- Dentro del bloque de configuración del script, se definen las rutas (URLs) que permiten a la Raspberry Pi comunicarse con la API intermedia desarrollada.

Figura 58

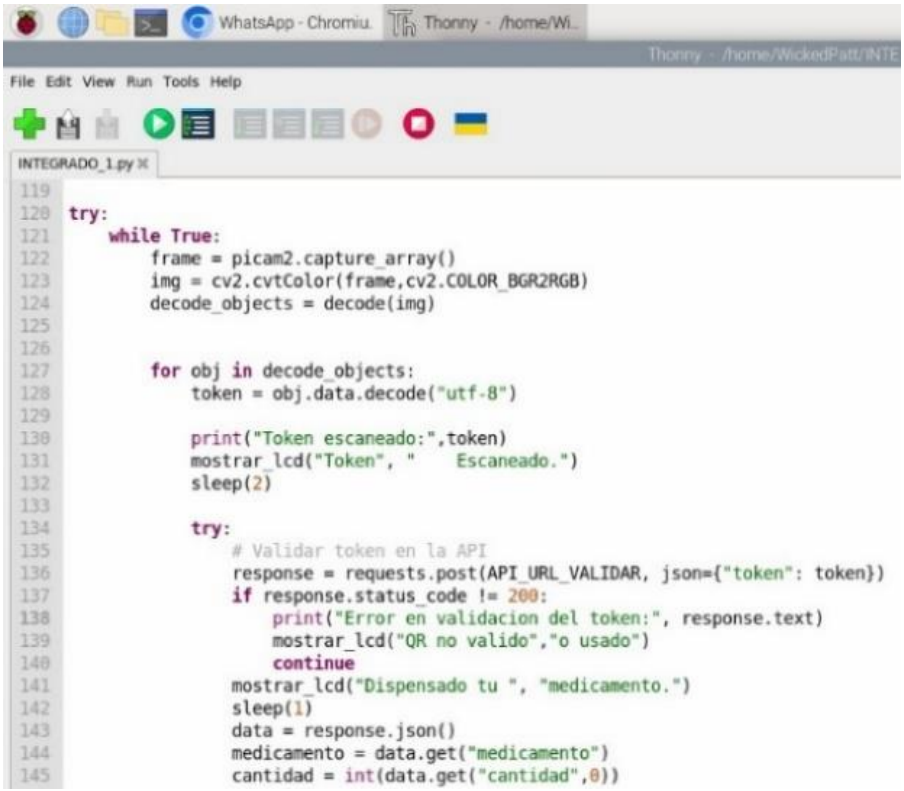
Rutas para Validación de Token y Registro de la Dispensación

Nota. Fragmento de código en Raspberry Pi que define las rutas de comunicación con la API intermedia en Node.js, encargada de validar tokens y registrar dispensaciones mediante consultas a la base de datos MongoDB. Elaboración propia.

En la figura 58 corresponde al código configurado en la Raspberry Pi, que establece las rutas y variables necesarias para comunicarse con la API intermedia desarrollada en Node.js. Esta API gestiona las solicitudes de validación de tokens y el registro de dispensaciones, realizando consultas directas a la base de datos MongoDB para verificar y actualizar la información correspondiente. Gracias a esta comunicación, el sistema logra integrar el hardware con la capa lógica de la aplicación, asegurando un flujo de datos confiable, en tiempo real y con trazabilidad completa del proceso de dispensación.

Figura 59

Raspberry Pi: Captura y Validación del Token QR

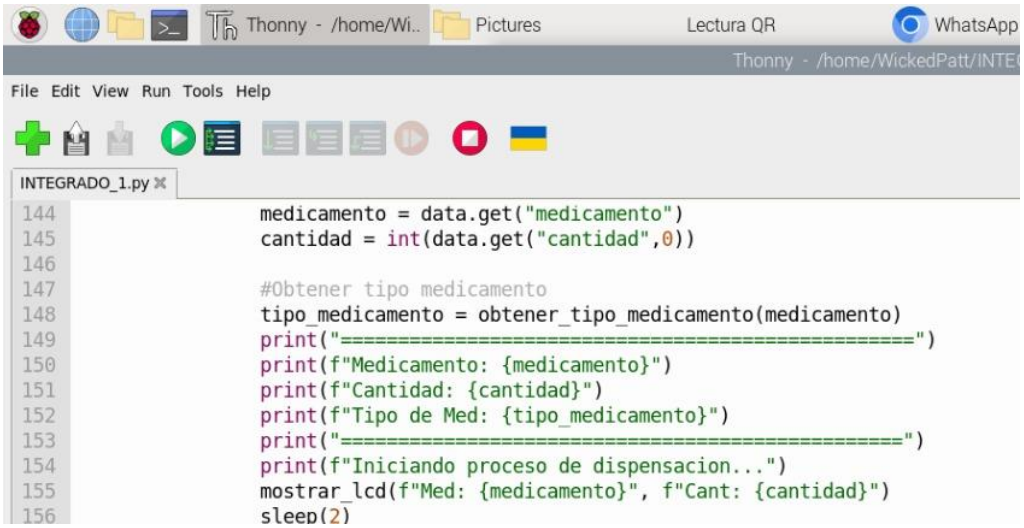


Nota. Fragmento de código en Raspberry Pi que define las rutas de comunicación con la API intermedia en Node.js, encargada de validar tokens y registrar dispensaciones mediante consultas a la base de datos MongoDB. Elaboración propia.

- Si la prescripción es válida, la API extrae los campos medicamento y cantidad del documento correspondiente y los devuelve en formato JSON. De esta manera, la Raspberry Pi recibe los datos necesarios para iniciar el proceso de dosificación según el nombre del medicamento y la cantidad recetada.

Figura 60

Obtención de Datos (Medicamento, Cantidad)

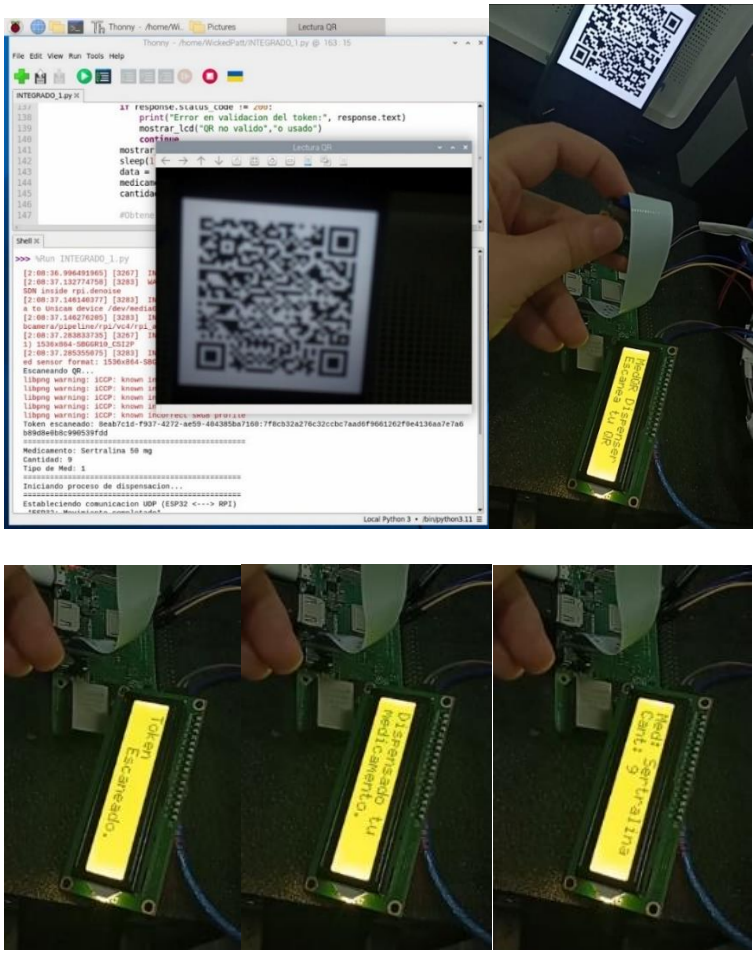


Nota. Fragmento de código en Python ejecutado en la Raspberry Pi para obtener y mostrar información del medicamento dispensado. Elaboración propia.

La figura 60 presentada pertenece al programa en Python que se ejecuta en la Raspberry Pi, encargado de procesar los datos recibidos desde la API intermedia en Node.js. En él, se extrae la información del medicamento y la cantidad a dispensar, determinando su tipo mediante una función específica. Además, el sistema imprime en consola los detalles del proceso y los muestra en la pantalla LCD del dispositivo, confirmando la operación al usuario.

Figura 61

Proceso de Escaneo de QR e Inicio de Dispensación con Indicador Visual LCD



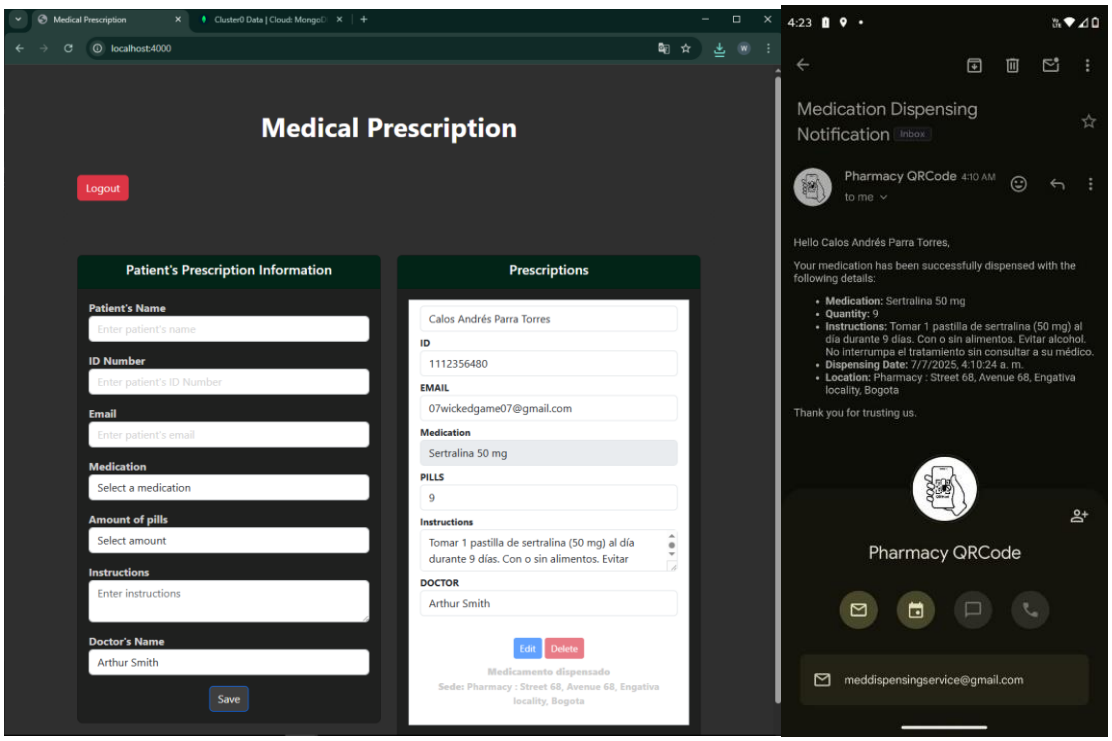
Nota. La imagen muestra el proceso de escaneo del QR mediante una cámara, cuyo token se envía a la API intermedia. Esta API valida la prescripción en la base de datos y extrae los datos de medicamento/cantidad, que luego son mostrados por la Raspberry Pi en la pantalla LCD para iniciar la dispensación. Elaboración propia.

En la figura 61 se observa cómo el usuario presenta el código QR frente al módulo de cámara, mientras que una pantalla LCD actúa como indicador visual mostrando mensajes que confirman la validación del token y la activación del sistema. Este mecanismo garantiza una interfaz clara y accesible para el usuario, validando la prescripción y controlando la entrega del medicamento de forma automatizada.

En el **Apéndice B s**e presenta un video demostrativo que ilustra la operación completa del sistema de validación de prescripciones mediante códigos QR y el inicio de la dispensación automatizada de medicamentos.

Figura 62

Dispensación Registrada en la Interfaz Web y Notificada en el Correo Electrónico



Nota. La imagen muestra la interfaz web en la prescripción del paciente, donde se evidencia una dispensación exitosa. En ella se detallan la fecha, la ubicación y la sede de la farmacia asignada al paciente, además de la notificación automática enviada por correo electrónico al mismo. Elaboración propia.

Estos procesos permiten que la Raspberry Pi controle el acceso a la dispensación física, solo cuando se valide una receta auténtica, evitando así la doble dispensación del medicamento.

Validación de Tokens y Registro en MongoDB Atlas. Toda la información sobre las prescripciones, pacientes y tokens se encuentra almacenada en la base de datos remota MongoDB Atlas.

- Se accede mediante un esquema Mongoose desde Node.js.
- Se actualiza el campo usado: true una vez que el medicamento ha sido entregado.
- Se registra también la sede y la fecha de dispensación como evidencia.

Figura 63

Registro de Dispensaciones en la Base de Datos

Nota. La imagen muestra el registro de trazabilidad en MongoDB, donde el sistema actualiza la dispensación del paciente con la fecha, sede de entrega y estado del token, asegurando el control y la evidencia de la receta dispensada. Elaboración propia.

Además de validar los tokens de prescripción, el sistema implementa un registro de trazabilidad dentro de la base de datos MongoDB. Cuando se valida una receta y se realiza la entrega del medicamento, el servidor API actualiza el documento del paciente con un subdocumento llamado dispensación, el cual almacena la fecha, la sede de

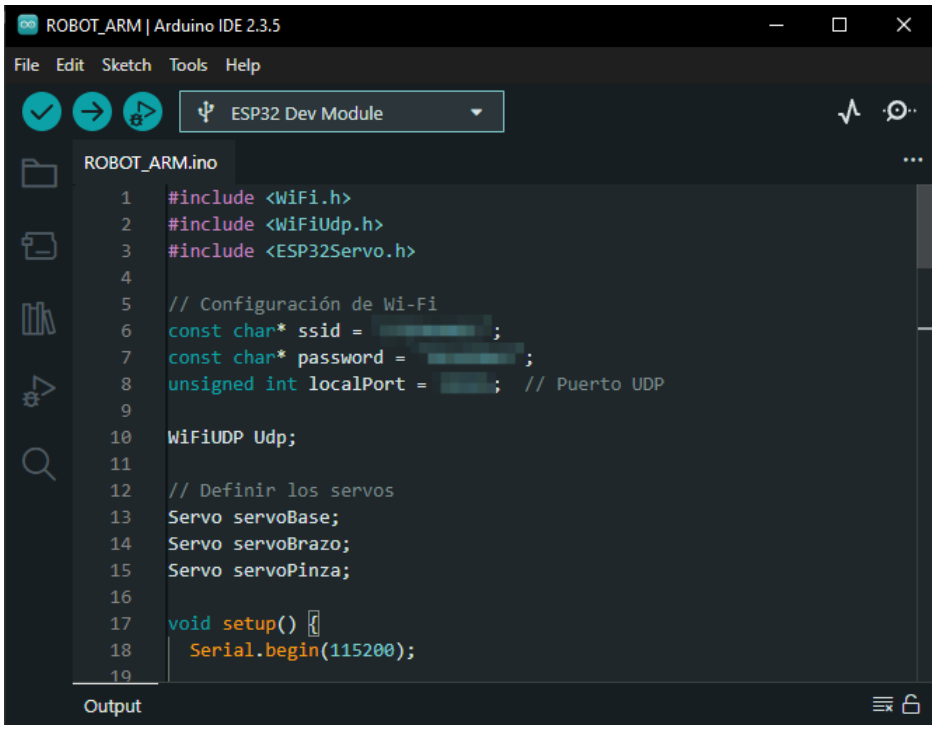
entrega y cambia el estado del campo usado a true. Esto permite mantener control y evidencia de que la receta fue dispensada correctamente.

Desarrollo de un Mecanismo Automatizado para Dispensación

Activación del Brazo Robótico. Para la primera fase del proceso de mecanizado automático, se utiliza un brazo robótico de 3 grados de libertad controlado por un ESP32, el cual opera tres servomotores: base, brazo y pinza. Este módulo está conectado a una red Wi-Fi y permanece a la espera de comandos enviados desde la Raspberry Pi mediante el protocolo UDP.

Figura 64

Fragmento de Conexión a Wifi y Definición del Puerto UDP

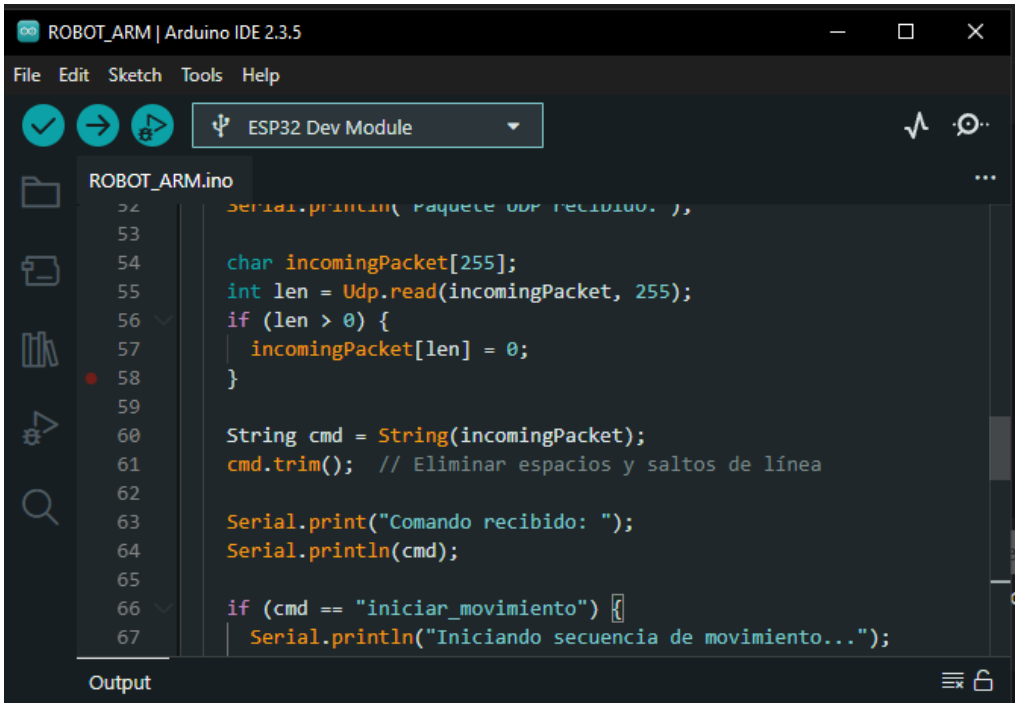


Nota. La imagen muestra el código del ESP32 donde se configura la conexión Wi-Fi y la recepción de paquetes UDP. Elaboración propia.

En la figura 64 se muestra un fragmento del código cargado en el ESP32, donde se configura la conexión Wi-Fi y la recepción de paquetes UDP. Una vez conectado a la red, el ESP32 se mantiene escuchando en el puerto UDP definido.

Figura 65

Fragmento del Código para Recepción de Comandos UDP en el ESP32.

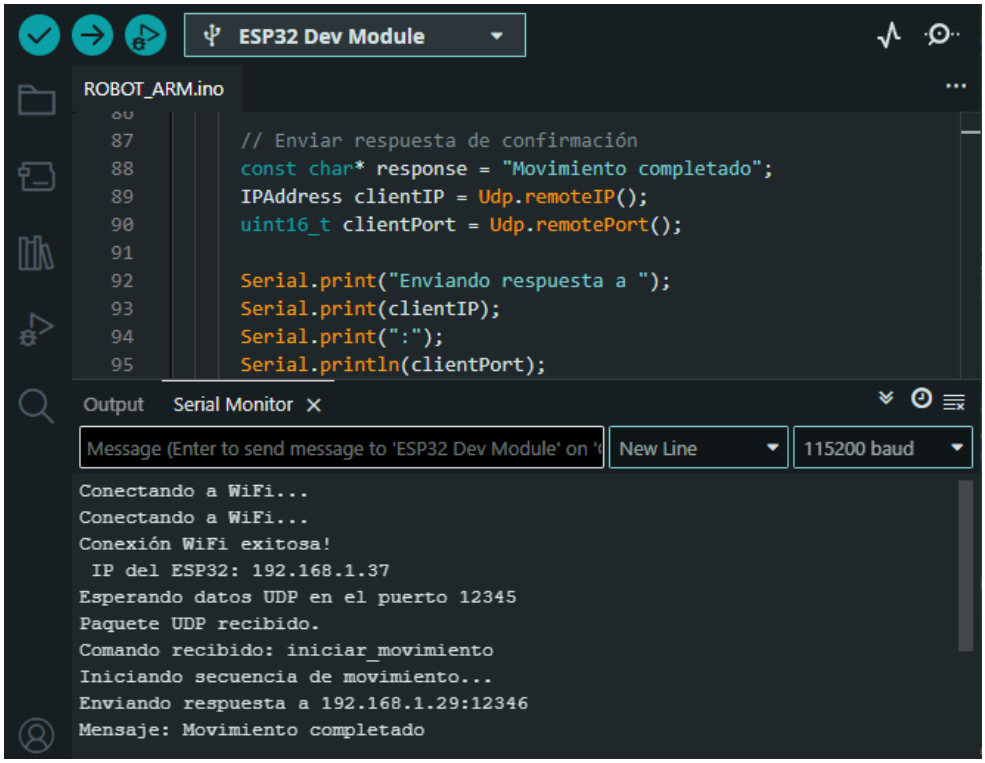


Nota. La imagen muestra el código del ESP32 que ejecuta la secuencia del brazo robótico al recibir un comando UDP. Elaboración propia.

La figura 65 muestra el fragmento de código donde el ESP32 recibe un comando UDP desde la Raspberry Pi, lo interpreta y, si es "iniciar_movimiento", activa la secuencia del brazo robótico como parte del proceso de mecanizado automático.

Figura 66

Envío de Respuesta de Confirmación Vía UDP desde el ESP32

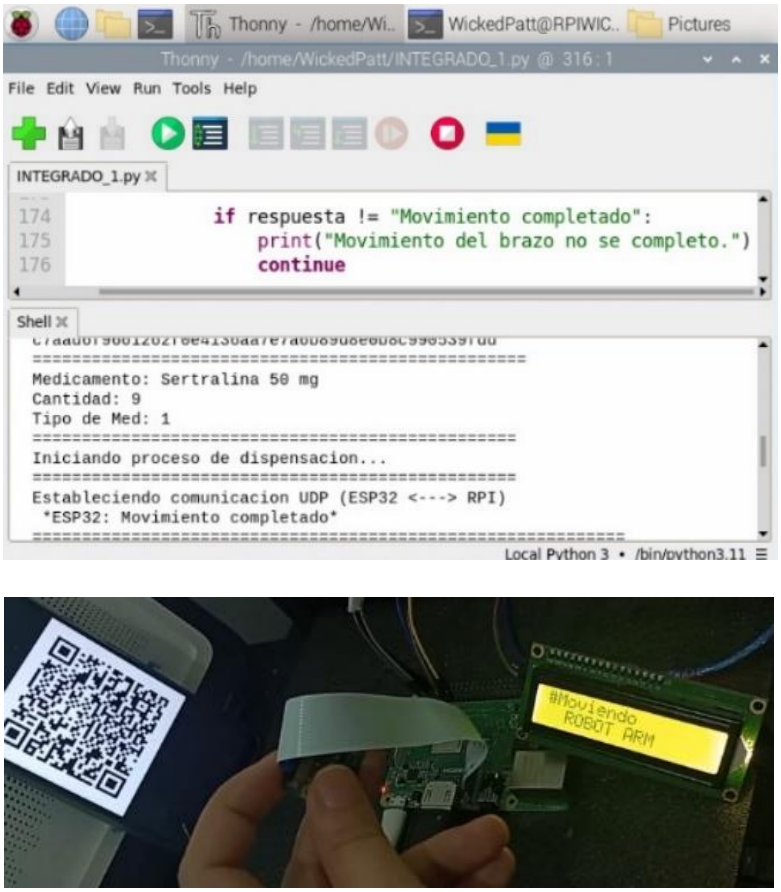


Nota. La imagen muestra cómo el ESP32, tras finalizar el movimiento del brazo robótico, envía un mensaje a la Raspberry Pi para coordinar las siguientes secuencias. Elaboración propia.

En la figura 66 se visualiza el proceso completo: la conexión del ESP32 a una red Wi-Fi, la recepción de un comando UDP con la instrucción "iniciar_movimiento", la ejecución de la secuencia de movimiento y el envío de la respuesta al cliente con dirección IP y puerto. Esto confirma la correcta recepción, ejecución y respuesta dentro del flujo de comunicación entre el cliente y el ESP32.

Figura 67

UDP entre Raspberry Pi y ESP32 para Control del Brazo Robótico

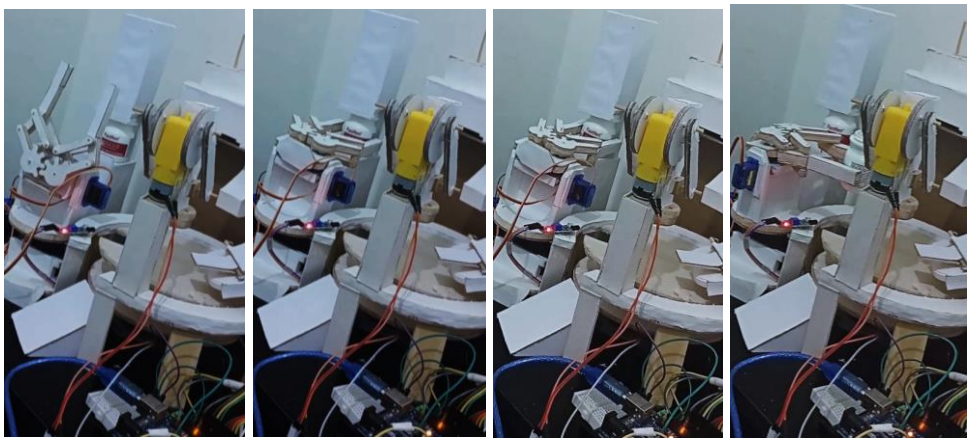


Nota. La imagen muestra la ejecución del programa durante la dispensación del medicamento. Se establece comunicación UDP entre la Raspberry Pi y el ESP32, y en la pantalla LCD se visualiza el mensaje “Moviendo robot arm”. Elaboración propia.

En la figura 67 se visualiza la ejecución del programa, que inicia el proceso de dispensación de un medicamento identificado como Sertralina 50 mg, con una cantidad de 9 unidades. El sistema establece comunicación UDP entre la Raspberry Pi y el ESP32, como lo indica el mensaje "Estableciendo comunicación UDP (ESP32 <--> RPI)". Posteriormente, se recibe desde el ESP32 la confirmación "ESP32: Movimiento completado", lo que verifica que la secuencia de movimiento del brazo robótico fue realizada correctamente.

Figura 68

Funcionamiento del Brazo Robótico en la Manipulación del Frasco.



Nota. La imagen muestra la trayectoria del brazo robótico utilizada para tomar el frasco y colocarlo en la base transportadora. Elaboración propia.

La figura 68 presenta una secuencia visual del proceso automatizado de dispensación, en la que se observa al brazo robótico ejecutando su rutina de movimiento. Esta secuencia evidencia la baja latencia en la transmisión de datos entre la Raspberry Pi y el ESP32, así como la correcta interpretación del comando por parte del microcontrolador, lo cual es fundamental para garantizar la sincronización y eficiencia del sistema automatizado de dispensación de medicamentos.

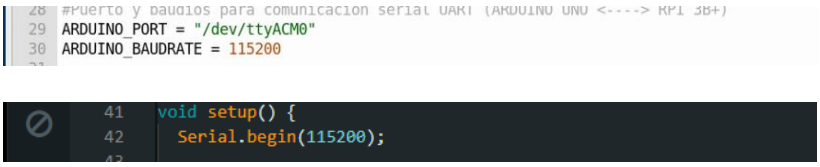
En el **Apéndice C** se anexa un video demostrativo del funcionamiento del brazo robótico tras la confirmación de dispensación

Dosificación de Pastillas. En el sistema de dispensación automatizada de medicamentos, la Raspberry Pi 3 actúa como el nodo principal del sistema, con capacidades de procesamiento, validación lógica y comunicación con una API médica. La ejecución de tareas físicas y de bajo nivel, como el control de motores y sensores, está delegada al Arduino Uno.

Para que ambos dispositivos se coordinen de manera efectiva, se implementa una comunicación bidireccional a través del protocolo UART (Universal Asynchronous Receiver Transmitter), lo que permite a la Raspberry Pi enviar comandos y recibir retroalimentación en tiempo real desde Arduino.

Figura 69

Configuración de Parámetros UART entre Raspberry Pi y Arduino Uno

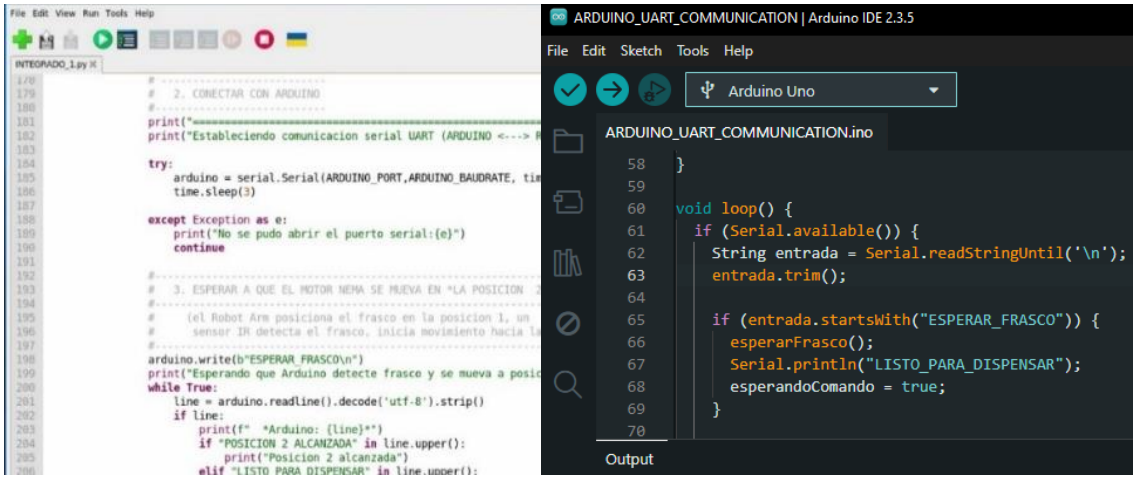


Nota. La imagen muestra la comunicación serial entre la Raspberry Pi y el Arduino. Elaboración propia.

La comunicación serial se establece por medio del puerto /dev/ttyACM0 (en Raspberry Pi) y opera a una velocidad de 115200 baudios, lo cual permite un intercambio eficiente de mensajes de texto codificados en UTF-8.

Figura 70

Intercambio de comandos UART entre RPi y Arduino Uno en dispensación



Nota. Comunicación Raspberry Pi–Arduino para controlar la dispensación automatizada mediante comandos y respuestas seriales. Elaboración propia.

El objetivo es controlar la secuencia de dispensación automatizada del medicamento a través del envío de comandos estructurados. Se inicia el envío de comandos en orden lógico, los cuales son interpretados por el Arduino para activar motores y sensores:

- Raspberry Pi como maestro: Envía comandos específicos al Arduino según el estado del proceso (por ejemplo, ESPERAR_FRASCO, DISPENSAR 1,3, etc.).
- Arduino como esclavo ejecutor: Interpreta cada comando recibido, ejecuta una acción (activar sensores, mover motores, contar pastillas, etc.), y luego responde a la Raspberry con un mensaje que indica el estado o el resultado de dicha acción.
- Raspberry Pi como receptor: Lee continuamente la salida del Arduino (serial.readline()), y solo avanza en el flujo si la respuesta esperada ha sido recibida.

Figura 71

Prueba Funcional de la Comunicación UART entre RPi y Arduino Uno.

Nota. La imagen muestra la comunicación serial Raspberry Pi–Arduino para activar dosificadores y contar pastillas. Elaboración propia.

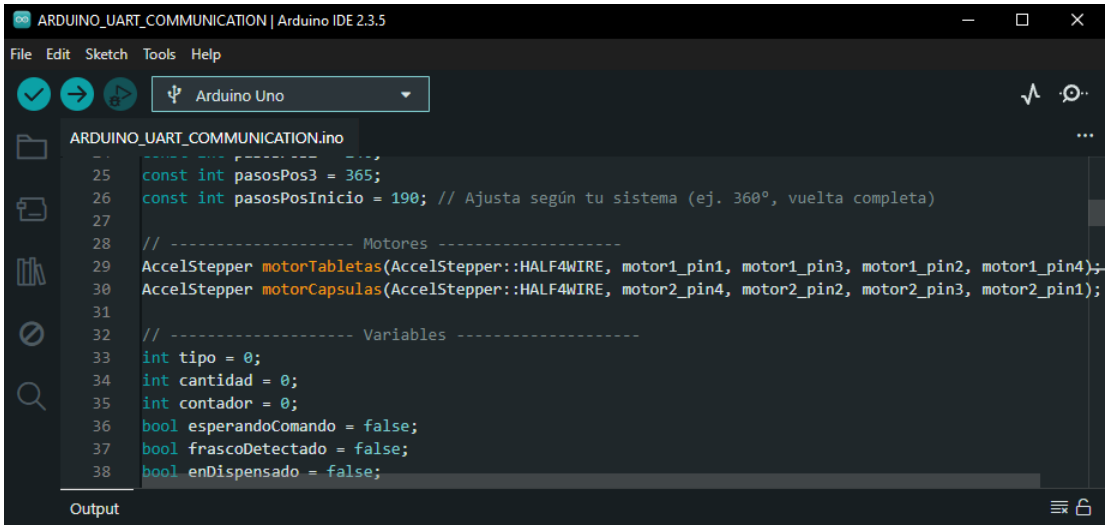
En la figura 71 se evidencia el intercambio bidireccional de datos entre la Raspberry Pi y el Arduino Uno a través del puerto serial UART. La Raspberry Pi envía comandos como ESPERAR_FRASCO y DISPENSAR 1,9, los cuales son interpretados por el Arduino. Este responde en tiempo real con mensajes de confirmación (LISTO_PARA_DISPENSAR) y estado (Tableta detectada: 1).

Esta salida validada demuestra que el sistema de dispensación automática ejecuta correctamente las órdenes y sincroniza sus acciones con base en sensores y motores.

El sistema incluye tres motores: dos motores stepper 28BYJ-48 para la dosificación y un motor NEMA 17 para el transporte del frasco entre estaciones.

Figura 72

Inicialización de Motores Paso a Paso para Dosificación



Nota. La imagen muestra un fragmento de código del Arduino Uno que controla los motores paso a paso 28BYJ-48, definiendo velocidad, sentido de giro y conteo de unidades con sensores infrarrojos. Elaboración propia.

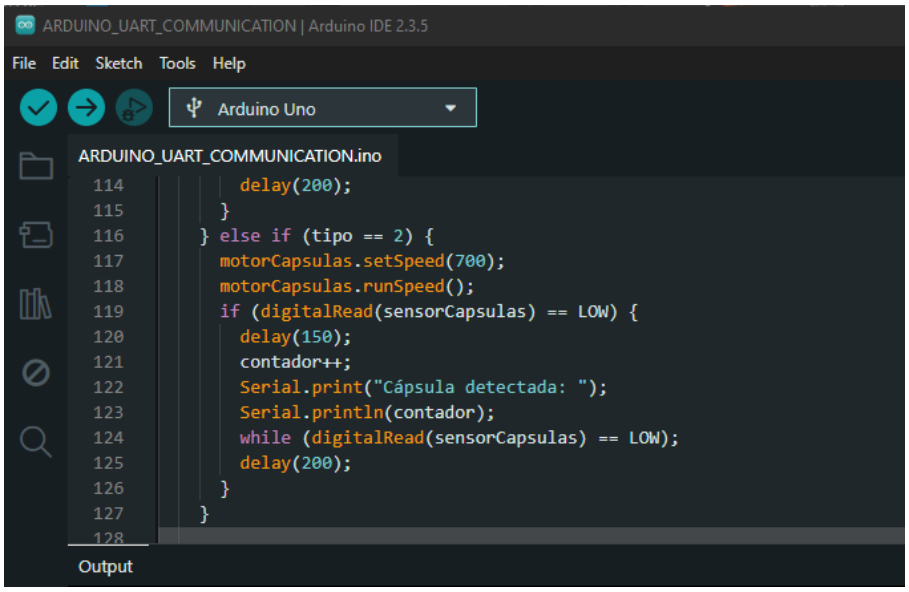
Cada tipo de medicamento (tableta o cápsula) es dispensado por un motor paso a paso 28BYJ-48. Se controlan desde el Arduino Uno usando la librería AccelStepper en

modo HALF4WIRE, que permite un movimiento preciso y controlado. Cada motor gira un disco dosificador que libera una unidad por giro parcial. La velocidad se establece con setSpeed(700) y se monitorea con sensores infrarrojos que cuentan cada unidad al detectarla en el canal de salida.

Un sensor IR (tipo ranura) detecta el paso de cada tableta o cápsula. Cuando se detecta un cruce (valor LOW), se incrementa un contador. Este proceso se repite hasta alcanzar la cantidad solicitada por la Raspberry Pi mediante UART.

Figura 73

Código Arduino para Conteo de Cápsulas con Sensor Infrarrojo



Nota. La imagen muestra un fragmento de código del Arduino Uno para seleccionar el tipo de medicamento y contar las pastillas detectadas por los sensores. Elaboración propia.

Figura 74

Proceso de Dosificación (Conteo de Pastillas)

Nota. La imagen muestra el funcionamiento de los dosificadores con la indicación del proceso de dosificación en la pantalla LCD. Elaboración propia.

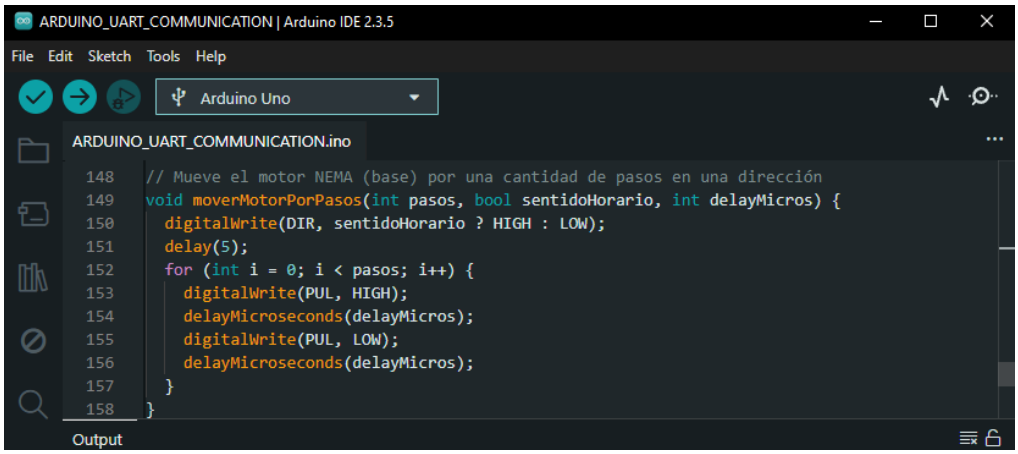
En el **Apéndice D** se muestra el funcionamiento de los dosificadores controlados por Arduino, con conteo de unidades mediante sensores infrarrojos

El frasco que recibe el medicamento es transportado entre tres estaciones (dosificación, tapado y entrega) mediante una base giratoria accionada por un motor NEMA 17 controlado por un driver TB6600. Este motor se activa desde el Arduino utilizando los pines PUL, DIR y EN, generando los pulsos necesarios para avanzar el eje en pasos controlados. La función moverMotorPorPasos(pasos, sentido, velocidad)

permite mover el frasco a posiciones específicas según los comandos enviados por la Raspberry Pi, como MOVER_POS3 o MOVER_POS_INICIAL.

Figura 75

Función para Controlar el Motor NEMA 17 desde Arduino



Nota. La imagen muestra un fragmento de código en Arduino para controlar un motor NEMA, definiendo posición, número de pasos, sentido de giro y velocidad. Elaboración propia.

Tapado del Frasco. Una vez completado el proceso de dosificación, el sistema ejecuta la etapa de tapado del frasco, con el fin de sellar el medicamento dispensado antes de su entrega. Esta acción se realiza desde la Raspberry Pi, la cual activa un motor DC tipo TT mediante una lógica de control GPIO.

El motor está conectado al controlador L298N y es gobernado por las salidas digitales IN1, IN2 y ENA, configuradas en el código Python. Para activar el tapado, se utiliza la función tapar_frasco():

Figura 76

Lógica de Tapado del Frasco mediante Motor Controlado por la Raspberry Pi

```
70
71 def tapar_frasco():
72     mostrar_lcd("#Tapando", "  frasco")
73     print("  *RPI: Motor TT activando para tapar*")
74     motor_adelante(85)
75     time.sleep(2)
76     motor_parar()
77     time.sleep(2)
78     print("Tapado Completado.")
79
```

```
240             while True:
241                 if arduino.in_waiting:
242                     linea = arduino.readline().decode().strip()
243                     print(f"  *Arduino: {linea}*")
244                     if "POSICIÓN 3 ALCANZADA" in linea:
245                         break
246             tapar_frasco()
247             time.sleep(2)
248             arduino.write(b"MOVER_POS_INICIAL\n")
249             print("COMANDO: MOVIMIENTO POSICION INICIAL.")
250             mostrar_lcd("#Dispensando", "frasco")
```

Nota. RPi para controla el motor del tapado del frasco, definiendo velocidad, sentido y activando el mecanismo al alcanzar la posición de cierre. Elaboración propia.

Figura 77

Mecanizado Slider Crank (Tapado)

Nota. La imagen muestra el mecanismo del slide crank aplicando presión sobre la tapa del frasco, mientras la pantalla LCD indica “Tapando frasco”. Elaboración propia.

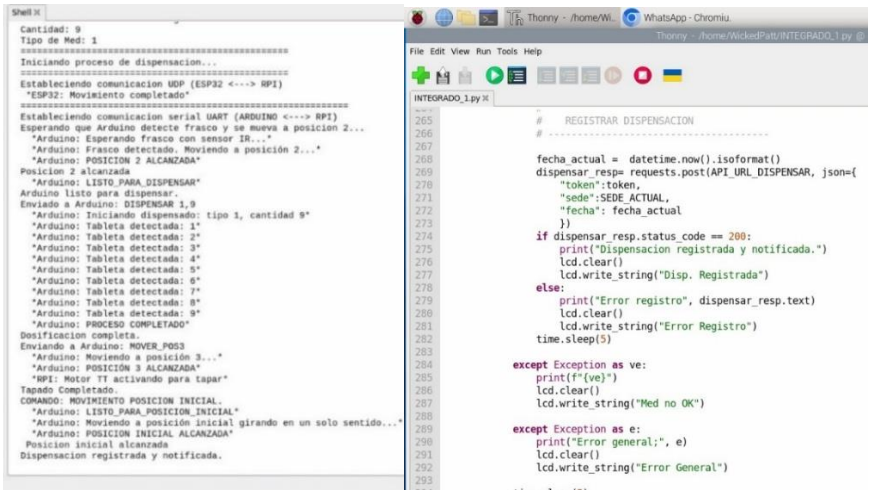
En el **Apéndice E** se ilustra un video demostrativo del funcionamiento del tapado automatizado mediante slide crank controlado por Raspberry Pi.

Validación del Flujo Completo del Sistema

Se concluye que el sistema cumple satisfactoriamente con el cuarto objetivo específico, logrando validar el flujo completo desde la validación de la receta médica mediante código QR, hasta la ejecución automática del proceso de dosificación y entrega del medicamento. Durante las pruebas funcionales, el sistema demostró un comportamiento estable y coordinado entre la Raspberry Pi y el Arduino, con una comunicación efectiva y sin errores críticos. El conteo de tabletas y cápsulas a través de sensores infrarrojos mostró un desempeño adecuado, alcanzando una precisión estimada del 98% en condiciones controladas. Esta validación confirma que el sistema es capaz de ejecutar de forma automatizada todas las etapas previstas, cumpliendo así su propósito principal y sentando las bases para futuras mejoras.

Figura 78

Terminal de la RPi, Código para Registro de Prescripciones y Envío de Correos



Nota. Evidenciando el flujo completo de la secuencia, y un fragmento de código utilizado para registrar y enviar la notificación por correo. Elaboración propia.

Figura 79

Pantalla LCD Indicador

Nota. La imagen muestra la pantalla LCD del sistema, primero indicando “Dispensando frasco” durante el proceso y luego “Disp. registrada” al completar la dispensación correctamente. Elaboración propia.

Resultados

Durante la fase Operar del modelo CDIO, se realizaron 10 pruebas funcionales para evaluar el desempeño del sistema automatizado de dispensación y tapado de medicamentos. En cada ensayo se dispensaron 9 pastillas, completando un total de 90 unidades procesadas. En 9 de las 10 pruebas, el conteo fue exacto, mientras que en una prueba se registró un error de +1 pastilla, asociado a una lectura doble del sensor infrarrojo. Aplicando la fórmula de precisión:

$$Precision\ (\%) = \frac{Cant.Correcta}{Cant.Total} \times 100 = \frac{Cant.Total - Errores}{Cant.Total} \times 100$$

$$Precision\ (\%) = \frac{90 - 1}{90} \times 100$$

$$Precision\ (\%) = 98.89\%$$

El sistema alcanzó una precisión global del 98.9 %

Tabla 25

Resultados de Pruebas Funcionales

Prueba	Tiempo total (s)	Validación token (ms)	Error de conteo	Latencia UART/UDP (ms)	Lectura QR exitosa (%)	Registro y notificación de dispensación	Medicamento correcto
1	74.5	510	—	520	100	exitosa	Sí
2	67.2	480	—	530	100	exitosa	Sí
3	71.6	500	—	550	100	exitosa	Sí
4	83.4	540	+1 cáp	590	100	exitosa	Sí
5	68.3	470	—	500	100	exitosa	Sí
6	72.7	490	—	560	100	exitosa	Sí
7	70.9	520	—	530	100	exitosa	Sí
8	80.1	550	—	600	100	exitosa	Sí

9	65.8	460	—	490	100	exitosa	Sí
10	76.3	500	—	540	100	exitosa	Sí
Prom.	73 s	510 ms (~0.5 s)	1 error en 10 pruebas	541 ms (~0.54 s)	100 %	100 % de registros y notificaciones exitosas	100 %

Nota. La tabla muestra los resultados de 10 pruebas del sistema automatizado, con un tiempo promedio de 73 s, validación y latencia de ~0.5 s, y 100 % de dispensaciones y medicamentos correctos. Se registró un único error aislado (+1 cápsula).

- En todas las pruebas el prototipo dispensó el medicamento correcto, garantizando el cumplimiento de la prescripción (100 % de éxito en selección de medicamento).
- La lectura de códigos QR, el registro y la notificación fueron 100 % exitosos.
- La latencia promedio entre módulos fue de 541 ms (~0.54 s), asegurando comunicación fluida y sin pérdidas de datos.

La Tabla 26 presenta la comparación entre el proceso manual tradicional y el sistema automatizado de dispensación. Se evidencian mejoras notables en velocidad, precisión, seguridad y trazabilidad. En particular, el tiempo promedio de entrega se redujo de 420 s a 73 s, equivalente a una disminución del 82.6 %, mientras que la validación digital, el registro automático y la selección correcta del medicamento demostraron una optimización integral del proceso.

$$Reduccion = \frac{(T_{manual} - T_{auto})}{T_{manual}} \times 100$$

Sustituyendo los valores (420 s y 73 s):

$$Reduccion = \frac{(420\ \text{s} - 73\ \text{s})}{420\ \text{s}} \times 100 = 82.62\%$$

Tabla 26

Comparativo de Eficiencia entre Dispensación Manual y Automatizada

Parámetro evaluado	Proceso manual tradicional	Sistema automatizado propuesto	Mejora / Observación
Tiempo promedio de entrega	420 s (≈ 7 min)	73 s (≈ 1 min 13 s)	Reducción del 82 % en el tiempo de atención.
Validación de prescripción	Revisión visual y firma manual	Validación digital por token QR en ≈ 0.5 s	Respuesta inmediata, sin fallos ni duplicaciones.
Selección del medicamento correcto	Dependiente del operador, susceptible a errores	Correcta en 100 % de las pruebas	Garantiza que siempre se dispense el medicamento correcto según la prescripción.
Latencia de comunicación	No aplica (proceso manual)	≈ 0.54 s (540 ms) promedio entre módulos	Comunicación fluida, sin pérdidas de datos.
Registro de dispensación	Manual, susceptible a omisiones	Automático en MongoDB Atlas con fecha, hora y token único	Registro trazable y auditable al 100 %.

Parámetro evaluado	Proceso manual tradicional	Sistema automatizado propuesto	Mejora / Observación
Notificación del proceso	No disponible o verbal	Automática en interfaz web tras dispensación	Confirmación inmediata del evento.
Riesgo de error humano	Alto (conteo, validación y registro)	Muy bajo (procesos guiados y controlados)	Reducción significativa de errores por intervención humana.
Trazabilidad digital	No implementada	Completa, con historial por receta, paciente y medicamento	Cumple estándares de control farmacéutico.
Disponibilidad operativa	Limitada al operador humano	Continua, autónoma y con registro en tiempo real	Mayor eficiencia y confiabilidad del sistema.

Nota. La tabla muestra la comparación entre el proceso manual tradicional y el sistema automatizado, destacando mejoras en eficiencia, seguridad y trazabilidad digital.

El sistema automatizado demuestra alta precisión, confiabilidad y eficiencia, con reducción significativa del tiempo de operación, registro y notificación automáticos, cero fallos en selección de medicamento, y trazabilidad digital completa. Frente al

proceso manual, se evidencia una mejora sustancial en seguridad, rapidez y control de errores humanos.

Los resultados obtenidos confirman el cumplimiento de los objetivos del proyecto. El sistema automatizado alcanzó una precisión global del 98.9 %, con una reducción del 82.6 % en el tiempo promedio de dispensación frente al proceso manual. Estos valores evidencian una mejora sustancial en la eficiencia y confiabilidad del sistema, garantizando la entrega exacta del medicamento prescrito, la trazabilidad digital completa y la eliminación de errores humanos en la validación y registro. La única desviación registrada se asocia a una lectura doble del sensor infrarrojo, lo que abre oportunidades de optimización futura en la calibración óptica. En conjunto, los resultados demuestran la validez técnica y funcional del prototipo, cumpliendo con los lineamientos de la metodología CDIO en su fase de operación.

Discusión

El desarrollo del prototipo automatizado para la dispensación controlada de medicamentos permitió comprobar la viabilidad técnica de integrar tecnologías de control, electrónica e informática en un único sistema funcional. La arquitectura implementada, basada en una Raspberry Pi como nodo central y en los microcontroladores ESP32 y Arduino Uno, demostró un desempeño estable y coordinado durante las fases de registro, validación y dispensación, cumpliendo con los objetivos propuestos.

En términos de impacto tecnológico, el proyecto representa un aporte significativo a la automatización farmacéutica en entornos donde la dispensación aún se realiza de forma manual. La incorporación de tecnologías como la verificación mediante códigos QR, la trazabilidad digital y la comunicación entre dispositivos embebidos mejora la precisión, la seguridad y el control de los medicamentos entregados. Desde el punto de vista académico, la experiencia integra conocimientos de programación, diseño electrónico y sistemas de control aplicados a una problemática real, fortaleciendo las competencias profesionales del ingeniero electrónico.

En el ámbito social y sanitario, la solución propuesta contribuye a reducir los errores humanos, garantizar la autenticidad de las prescripciones médicas y promover un uso más seguro de los medicamentos controlados. Además, facilita la generación de registros electrónicos que pueden resultar útiles para auditorías y sistemas de control sanitario, aportando a la transparencia y a la eficiencia del proceso farmacéutico.

A pesar de los resultados obtenidos, se identificaron algunas limitaciones que deberán considerarse en futuras versiones del sistema. El prototipo fue evaluado en condiciones controladas de laboratorio, por lo que aún no se ha probado en escenarios farmacéuticos reales. La precisión del mecanismo de dosificación depende del tamaño,

forma y peso de las pastillas, lo que podría requerir ajustes mecánicos y de calibración específicos para cada tipo de medicamento. Asimismo, la comunicación mediante protocolos UART y UDP, aunque efectiva en el prototipo, podría complementarse con mecanismos de validación o redundancia en versiones industriales con el fin de garantizar mayor confiabilidad en la transmisión de datos.

El sistema desarrollado evidencia un alto potencial de escalamiento. Su diseño modular permite incorporar nuevas funcionalidades, como sensores de peso o sistemas de visión artificial para la verificación visual de las unidades dispensadas, además de la posibilidad de integrarse con plataformas de prescripción médica electrónica o sistemas hospitalarios. Estas mejoras facilitarían su implementación en farmacias, hospitales y centros de salud, contribuyendo a la transformación digital y al fortalecimiento de la seguridad farmacéutica en Colombia.

Plan de mejoras

Plan de Mejoras A Corto Plazo

1. Optimización del algoritmo de dosificación

Implementar una lógica de doble verificación usando los sensores infrarrojos para evitar dispensaciones erróneas.

2. Interfaz gráfica mejorada (HMI)

Agregar una pantalla táctil o interfaz web embebida para facilitar la navegación de personal autorizado.

3. Validación de doble capa de seguridad

Incorporar una autenticación adicional antes del escaneo del código QR, como un número PIN temporal enviado al correo del paciente.

4. Mejora en la integración de comunicación

Ajustar los tiempos y la sincronización de la comunicación UART (entre la Raspberry Pi y el Arduino) y UDP (entre la Raspberry Pi y el ESP32) para evitar cuellos de botella.

5. Diseño físico mejorado del prototipo

Reemplazar estructuras de prueba (balsa o cartón) por materiales más estables como acrílico, MDF o impresión 3D para mejorar la estabilidad y estética del sistema.

Organizar mejor el cableado y asegurar los componentes a una base modular.

6. Documentación técnica completa

Elaborar un manual de usuario para el farmacéutico y el paciente.

Documentar en un repositorio digital (por ejemplo, GitHub) el código fuente, diagrama de conexiones y manuales de mantenimiento.

7. Pruebas piloto con usuarios reales

Realizar pruebas controladas en un entorno simulado con estudiantes o voluntarios, recogiendo retroalimentación para ajustar procesos.

Generar estadísticas sobre fallas, tiempo de dispensado y eficiencia del sistema para su análisis.

Plan de mejoras a mediano plazo

Con el fin de asegurar la escalabilidad, sostenibilidad y adaptación del proyecto a contextos reales de implementación en farmacias o centros de salud, se proponen las siguientes mejoras para un plazo de dos años:

1. Diseño y fabricación de una PCB personalizada

Reemplazar las conexiones por protoboards y cables jumper con una placa de circuito impreso (PCB), optimizando espacio, confiabilidad eléctrica y seguridad.

2. Desarrollo de una aplicación móvil

Integrar una app que notifique al paciente cuándo puede reclamar su medicamento, mostrar el historial de prescripciones y habilitar consultas remotas.

3. Integración con bases de datos hospitalarias o EPS

Permitir que el sistema se vincule directamente con los sistemas de información de salud (HIS), garantizando una trazabilidad más eficiente y evitando recetas fraudulentas.

4. Implementación de algoritmos de mantenimiento predictivo

Desarrollar modelos basados en aprendizaje automático para anticipar fallas de motores, sensores o actuadores a partir de su comportamiento histórico.

5. Optimización mecánica del sistema de dosificación y tapado

Diseñar un módulo mecánico más compacto y eficiente que permita mayor velocidad y precisión en la dispensación.

6. Adaptación del sistema a normativas INVIMA y certificación médica

Preparar el prototipo para su posible homologación y certificación como equipo médico legalmente autorizado.

Plan de Mantenimiento

Plan de Mantenimiento Correctivo

Anticipar posibles fallas a través de historial o experiencia previa, de modo que, al requerir una intervención, se cuente con los recursos necesarios (personal, repuestos, herramientas y documentación técnica).

Ejemplos de acciones:

- Sustitución de servomotor MG995 o SG90 en caso de pérdida de torque o desalineación.
- Reemplazo de sensores infrarrojos defectuosos en estaciones de conteo.
- Corrección de errores de comunicación UART o UDP ante fallos de sincronización.
- Sustitución de drivers (A4988 o TB6600) si presentan sobrecalentamiento crónico o pérdida de pasos.

Plan de Mantenimiento Predictivo

Basado en predecir y anticipar fallos a partir del monitoreo periódico de variables críticas del sistema, permitiendo intervenir justo antes de que ocurra una falla.

Variables para monitorear:

- Temperatura de drivers y servomotores (detecta sobrecargas o mala ventilación).
- Consumo eléctrico de motores paso a paso mediante sensores tipo ACS712.
- Vibraciones anómalas en la base giratoria o brazo robótico con acelerómetros.
- Latencia en comunicaciones (UDP o UART), evaluando posibles cuellos de botella o errores de transmisión.

Herramientas sugeridas:

- Sensores de corriente (ACS712 o INA219).
- Acelerómetros (como MPU6050) conectados al microcontrolador para análisis vibracional.
- Scripts de diagnóstico en Python para registrar y graficar métricas en tiempo real.

Plan de Mantenimiento Preventivo

Ejecutar acciones programadas periódicamente para conservar el sistema en condiciones óptimas de funcionamiento y prevenir fallas derivadas del uso prolongado.

Frecuencia y actividades recomendadas:

- Cada 15 días:

Revisión del torque de los motores, mecanizado del brazo robot (servomotores) y del motorreductor TT (mecanismo Slide Crank).

- Cada 30 días:

Limpieza general de sensores ópticos y cámaras (evita errores de lectura o visión).

Inspección de conectores, cables y soldaduras (identificación de falsos contactos o corrosión).

Calibración y alineación de los motores dosificadores y sensores IR.

- Cada 60 días:

Pruebas funcionales completas de todos los subsistemas (dosificación, brazo, tapado, comunicaciones).

Revisión de los archivos de log para identificar patrones de errores o degradación.

Ventajas:

Reducción de tiempos de inactividad no programada.

Mejora de la confiabilidad del sistema en entornos farmacéuticos.

Prolongación de la vida útil de componentes críticos.

Mejora de la trazabilidad operativa mediante registros periódicos.

Conclusiones

El desarrollo del prototipo automatizado para la dispensación controlada de medicamentos con verificación electrónica mediante códigos QR permitió demostrar la viabilidad de integrar herramientas de hardware y software en un sistema funcional, seguro y trazable. El proyecto cumplió con el objetivo de diseñar e implementar una solución tecnológica capaz de registrar, validar y dispensar medicamentos de manera automatizada, reduciendo la intervención humana y fortaleciendo los mecanismos de control sanitario.

La validación electrónica implementada a través de códigos QR resultó ser una herramienta efectiva para garantizar la autenticidad de las prescripciones médicas y asegurar que el medicamento entregado corresponda exactamente con la receta registrada. Este mecanismo no solo contribuye a la reducción de errores de dispensación, sino que también incrementa la confianza del usuario y facilita el seguimiento digital del proceso farmacéutico.

Asimismo, la integración de tecnologías IoT, microcontroladores y bases de datos en la nube permitió construir un sistema robusto y adaptable, capaz de ofrecer trazabilidad en tiempo real y comunicación efectiva entre los módulos de registro, validación y dispensación. La metodología CDIO aplicada en el desarrollo aseguró una evolución ordenada del proyecto desde la concepción hasta la operación, evidenciando la aplicación práctica de los conocimientos adquiridos durante la formación en ingeniería electrónica.

No obstante, se identificaron ciertas limitaciones que abren oportunidades de mejora. El prototipo fue validado en entornos controlados, por lo que sería conveniente realizar pruebas en condiciones reales de operación farmacéutica para evaluar su desempeño ante mayores volúmenes de dispensación y variabilidad de medicamentos.

De igual manera, la precisión del sistema de dosificación depende de las características físicas de las pastillas, lo que sugiere la necesidad de ajustes mecánicos y algoritmos adaptativos. También se considera pertinente fortalecer la comunicación entre microcontroladores mediante protocolos con redundancia y verificación para incrementar la confiabilidad en versiones futuras.

Entre las limitaciones identificadas se encuentran la sensibilidad de algunos sensores a condiciones externas de iluminación y la necesidad de realizar ajustes mecánicos para asegurar un tapado más firme. Aun así, el sistema presentó un funcionamiento estable en pruebas integradas, lo cual demuestra su viabilidad técnica.

Este proyecto constituye una propuesta funcional, replicable y escalable para la automatización parcial o total de procesos de dispensación de medicamentos en farmacias o instituciones de salud. Finalmente, el sistema desarrollado constituye una base sólida para su escalamiento hacia aplicaciones industriales o clínicas, gracias a su diseño modular y a la posibilidad de integrar nuevos elementos como sensores de peso, cámaras de visión artificial y conexión con sistemas de historia médica electrónica. Este proyecto no solo representa un avance tecnológico, sino también una contribución significativa a la seguridad del paciente, la eficiencia operativa y la transformación digital del sector farmacéutico colombiano.

Referencias

Amazon. (2025). *DFROBOT TB6600 Stepper Motor Driver* [Figura]. https://www.amazon.com.be/-/en/DFROBOT-TB6600-Stepper-Motor-Driver/dp/B073TLMVZJ

Arduino (2025). *Arduino Uno Rev3 Technical Specifications.* Sitio Web*:* https://www.arduino.cc/en/Main/ArduinoBoardUno

Arduino Forum (2024, diciembre 19). *Esp32 with qr sensor gm60.* [Imagen]. https://forum.arduino.cc/t/esp32-with-qr-sensor-gm60/1333870

Arduinokit Project. (2023, julio 15). *Interfacing 28BYJ-48 stepper motor Arduino using ULN2003 driver*. [Figura] ARDUINOKIT PROJECT. https://arduinokitproject.com/28byj48-stepper-motor-arduino-tutorial/#google_vignette

Bakker, B. (2025, 21 abril). *TB6600 Stepper Motor Driver with Arduino Tutorial*. Makerguides.com. https://www.makerguides.com/tb6600-stepper-motor-driver-arduino-tutorial/

Bentrán, A. (2024). *Sensor de proximidad infrarrojo FC-51*. [Archivo PDF]. https://agelectronica.lat/pdfs/textos/O/OKY3127.PDF

Beshir, M. (2025, 11 marzo). *Pharmacy Automation: The Future of Medication Safety and Efficiency*. Pharmacy Times. Sitio Web: https://www.pharmacytimes.com/view/pharmacy-automation-the-future-of-medication-safety-and-efficiency

Božić, R. (2022). *Advantages and Challenges of Nosql Compared to Sql Databases - a Systematic Literature Review.* https://doi-org.bibliotecavirtual.unad.edu.co/10.7251/ZREFB2216011B

Chavhan, K., Johnson, N. J., Monika, P., S, P., & Reddy, T. J. (2022). *Analysis of Technologies used in Automatic Medicine Dispensing Systems*. https://doi-org.bibliotecavirtual.unad.edu.co/10.1109/ICDCECE53908.2022.9792921

Components101. (2025). *NEMA 23 stepper motor*. [Figura]. Pinterest. https://co.pinterest.com/pin/732538695625153703/

Electronilab. (2025). *Cámara para Raspberry Pi V3 – 12 Megapíxeles Original.* [Figura]. Sitio Web: https://electronilab.co/tienda/camara-para-raspberry-pi-v2-8-mpx/?srsltid=AfmBOoqxEQd4waUpMMPNNFL0_ob3puT6gTCR-GNbHAkbbj8kt9d5fSt0

Electronilab. (2025). *Display LCD 16x2 con Backlight Amarillo*. [Figura]. Sitio Web: https://electronilab.co/tienda/display-lcd-16x2-con-backlight-amarillo/?srsltid=AfmBOor4_eei8KVHsDGCTJfITTkuGSKqCrid41DGOT8nmLBP0zorLMvx

Electronilab. (2025). *Driver Dual L298N para motores – Puente H*. [Figura]. Sitio Web: https://electronilab.co/tienda/driver-dual-para-motores-full-bridge-l298n/?srsltid=AfmBOorMr2XhFbwzH7tOKqCS5pjCIu3rkIwmnyweOnBlY8mltBg8na-Q

Electronilab. (2025). *Módulo LM2596 DC-DC Buck reductor con voltímetro*. [Figura]. Sitio Web:. https://electronilab.co/tienda/modulo-lm2596-dc-dc-buck-1-25v-35v-con-voltimetro/?srsltid=AfmBOooThzvlEePJjdIa5tFEAvvexje4vESl4iw8zmSfUESx3kClz7zm

Electronilab. (2025). *Motor paso a paso 28BYJ-48 0.3 Kgr/Cm*. [Figura]. Sitio Web: https://electronilab.co/tienda/motor-paso-a-paso-28byj-48-0-3kgr-

cm/?srsltid=AfmBOoqwrEoBCX3cviXRGmQA73t5gxMeesf_jZYuAK8PWan6yp5hxd3U

Electronilab. (2025). *Motor paso a paso NEMA 23 – 178.5 oz·in (1.26 Nm) – 200 pasos/vuelta.* [Figura]. Sitio Web: https://electronilab.co/tienda/motor-paso-a-paso-nema-23-125-oz-in-200-pasos-vuelta/

Electronilab. (2025). *Motorreductor con caja reductora 6V*. [Figura]. Sitio Web: https://electronilab.co/tienda/motorreductor-con-caja-reductora-6v-1-48/?srsltid=AfmBOopuolfpshOcuNkTj75_uUAF6K1UgDc-MQyI_VoTI2ujgCKVNyVd

Electronilab. (2025). *Servomotor MG995 piñonería metálica*. [Figura]. Sitio Web: https://electronilab.co/tienda/mg995-tower-pro-servo-motor-metalico/?srsltid=AfmBOorfQELv67pxEbB0ykSo4V0ZBXk27IFrT5RURR7McJ5psYcbFcP4

Espressif Systems. (2021). *ESP32-WROOM-32 Datasheet. Espressif Systems*. https://www.espressif.com/en/products/socs/esp32/resources

Ferretrónica. (2025). *Módulo Adaptador Interfaz I2C para LCD 2x16 – LCD 4x20*. [Figura]. Sitio Web: https://www.ferretronica.com/products/modulo-adaptador-interfaz-i2c-para-lcd-2x16-lcd-4x20?srsltid=AfmBOooOzwyA6VqeWTfpUGWFohqP-UcYBcxDh6zw5qHJNHbHOWn-EJJ3

Fifarma. (2024, 17 mayo). *Errores de medicación y dispensación en América Latina*. Sitio Web: https://fifarma.org/errores-de-medicacion-y-dispensacion-en-america-latina/

Instructables (2019, junio 1). *Arduino Ultrasonic Distance Meter With 7 Segment Display*. [Figura]. https://www.instructables.com/Arduino-Ultrasonic-Distance-Meter-With-7-Segment-D/

Joshi, P., y Sawant, S. (2024). *The Impact and Potential of Quick Response (QR) Codes in Healthcare: A Comprehensive Review.* Proceedings of the Human Factors and Ergonomics Society Annual Meeting. Sitio Web: https://doi-org.bibliotecavirtual.unad.edu.co/10.1177/10711813241278266

Machado, M., Machado J., Gaviria, A., Valladales, L., Parrado, I., Ospina, M., et al. (2021). *Detección de errores de medicación mediante un programa de seguimiento y minimización en pacientes ambulatorios de Colombia, 2018–2019.* https://doi.org/10.7705/biomedica.5544

Muthulakshmi, K., Sowndarya, K., Yogitaa Devi, R. C., & Tharani, R. (2023). *A Safe and Reliable Identity based Healthcare System*. https://doi-org.bibliotecavirtual.unad.edu.co/10.1109/ICECA58529.2023.10395540

Penna, M., Gowda, D., Jijesh, J., & Shivashankar. (2017). *Design and implementation of automatic medicine dispensing machine.* https://doi-org.bibliotecavirtual.unad.edu.co/10.1109/RTEICT.2017.8256941

Posada, F. (2020, septiembre 8). *Raspberry Pi: mini-ordenador para proyectos educativos.* [Blog]. Canaltic.com. https://canaltic.com/blog/?p=3734

Quintero, B. (2025, 14 julio). *Diseño e implementación de un sistema seguro y accesible para el registro de prescripciones médicas* [Vídeo]. YouTube. https://www.youtube.com/watch?v=lA8JZqvNwWc

Quintero, B. (2025, 26 octubre). *Dosificación de pastillas* [Vídeo]. Youtube. https://youtube.com/shorts/r6yc9MiXe6M?feature=share

Quintero, B. (2025, julio 14). *Activación del brazo robótico* [Vídeo]. YouTube. https://www.youtube.com/watch?v=87Cz1hto6pM

Quintero, B. (2025, julio 14). *Implementación de un sistema de validación mediante códigos QR* [Vídeo]. YouTube. https://www.youtube.com/watch?v=PgBjJXojN88

Quintero, B. (2025, julio 14). *Tapado del frasco* [Vídeo]. YouTube. https://www.youtube.com/watch?v=UumBSJtjsgI

Raspberry Pi Foundation. (2018). *Raspberry Pi 3 Model B+ Product Brief. Raspberry Pi* Foundation. Sitio web: https://www.raspberrypi.com/documentation/computers/raspberry-pi.html

Redacción Salud (2021). *¿Qué tan comunes son los errores de medicación?* EL ESPECTADOR. Sitio Web: https://www.elespectador.com/salud/que-tan-comunes-son-los-errores-de-medicacion-article/

Shanthini, E., Subasri, A., Yagavi, T., & Vinodhini, M. (2024). *QR-Code Automated Drug Dispenser*. https://doi-org.bibliotecavirtual.unad.edu.co/10.1109/ICSCSS60660.2024.10625535

Shanthini, M., Vedhavarshini, U., Prem, N., Gouri, K., Roshan, V. S., & Reena Josephine, S. S. (2023). *Design and Implementation of IoT based Automatic Medicine Dispenser for Patients.* https://doi-org.bibliotecavirtual.unad.edu.co/10.1109/ICIMIA60377.2023.10426176

UNIT Electronics. (2025, 24 octubre). *Fuente conmutada 12V 10A - AC110-220V 50/60Hz*. [Figura]. Sitio web. https://uelectronics.com/producto/fuente-conmutada-12v-10a/

Vistronica. (2025). *Micro servomotor SG90 9g*. [Figura]. Sitio web: https://www.vistronica.com/robotica/motores/servomotores/micro-servomotor-sg90-9g-detail.html

Yul Chu, & Park, J. H. (2023). *Efficient learning modules for embedded system. International Journal of Electrical Engineering Education*. https://doi-org.bibliotecavirtual.unad.edu.co/10.1177/0020720920918153

Apéndices

Apéndice A

Funcionamiento del Sistema Web

Este video muestra el funcionamiento del sistema web: registro de prescripciones, almacenamiento en la base de datos y envío automático de token QR al correo del paciente.

Enlace al video: Quintero, B. (2025, 14 julio). Diseño e implementación de un sistema seguro y accesible para el registro de prescripciones médicas. https://www.youtube.com/watch?v=lA8JZqvNwWc

Apéndice B

Secuencia de Operación del Prototipo

Este video presenta la secuencia completa de operación del prototipo, desde el escaneo del código QR hasta la validación del token por la API.

Enlace al video: Quintero, B. (2025, julio 14). Implementación de un sistema de validación mediante códigos QR. https://www.youtube.com/watch?v=PgBjJXojN88

Apéndice C

Funcionamiento del Brazo Robótico tras la Confirmación de Dispensación

Este video muestra el funcionamiento del brazo robótico después de la confirmación de dispensación, incluyendo la toma y colocación del frasco en la base transportadora.

Enlace al video: Quintero, B. (2025, julio 14). Activación del brazo robótico. https://www.youtube.com/watch?v=87Cz1hto6pM

https://www.youtube.com/watch?v=PgBjJXojN88

Apéndice D

Funcionamiento de Dosificadores y Conteo con Sensores Infrarrojos

Este video muestra el funcionamiento de los dosificadores automatizados y el conteo de pastillas mediante sensores infrarrojos.

Enlace al video: Quintero, B. (2025, 26 octubre). Dosificación de pastillas. https://youtube.com/shorts/r6yc9MiXe6M?feature=share

Apéndice E

Funcionamiento del Tapado Automatizado con Control de Slide Crank

Este video muestra el funcionamiento del tapado automatizado mediante slide crank, controlado por Raspberry Pi, con activación al alcanzar la posición de cierre.

Enlace al video: Quintero, B. (2025, julio 14). Tapado del frasco. https://www.youtube.com/watch?v=UumBSJtjsgI