**Desarrollo e implementación de un sistema web de inventarios y validación de códigos de barras**

**Ángel Omar Flores Pérez, Daniela Méndez Raíz**

**Tecnológico de Estudios Superiores de Villa Guerrero**

**L.I Emmanuel Carbajal Bravo**

**14 de octubre del 2025**

# Agradecimiento

# Resumen

En este documento de Residencia profesional

**Índice General**

[Agradecimiento 2](#_Toc216688939)

[Resumen 3](#_Toc216688940)

[Introducción 16](#_Toc216688941)

[Descripción de la empresa u organización y del puesto o área del trabajo el estudiante 18](#_Toc216688942)

[Descripción de empresa 18](#_Toc216688943)

[Puesto o Área de trabajo del Estudiante 19](#_Toc216688944)

[Problemas para resolver, priorizándolos 21](#_Toc216688945)

[Objetivos (General y Específicos) 22](#_Toc216688946)

[Objetivo General 22](#_Toc216688947)

[Objetivos Específicos 22](#_Toc216688948)

[Alcances 22](#_Toc216688949)

[Limitaciones 23](#_Toc216688950)

[Justificación 24](#_Toc216688951)

[Marco Teórico (fundamentos teóricos) 26](#_Toc216688952)

[Metodología CASCADA 29](#_Toc216688953)

[Fases de la metodología 30](#_Toc216688954)

[Fase 1 30](#_Toc216688955)

[Análisis 30](#_Toc216688956)

[Fase 2 31](#_Toc216688957)

[Diseño 31](#_Toc216688958)

[Fase 3 33](#_Toc216688959)

[Implementación 33](#_Toc216688960)

[Fase 4 33](#_Toc216688961)

[Verificación 33](#_Toc216688962)

[Fase 5 34](#_Toc216688963)

[Mantenimiento 34](#_Toc216688964)

[Cronograma de actividades 35](#_Toc216688965)

[Fase 1 43](#_Toc216688966)

[Análisis 43](#_Toc216688967)

[1. Herramienta de recolección de información. 43](#_Toc216688968)

[2. Generar la propuesta 43](#_Toc216688969)

[3. Definir los requisitos funcionales del sistema. 44](#_Toc216688970)

[4. Analizar el lector de código de barras como protocolos soportados, configuración. 44](#_Toc216688971)

[5. Establecer el flujo de trabajo en la línea de producción como el escaneo, Validación, Visualización y registro de inventarios. 45](#_Toc216688972)

[6. Determinar roles y permisos (administrador, operador). 46](#_Toc216688973)

[Fase 2 47](#_Toc216688974)

[Diseño 47](#_Toc216688975)

[1. Crear diagramas de caso de uso 47](#_Toc216688976)

[2. Diagramas secuenciales 48](#_Toc216688977)

[3. Arquitectura cliente-servidor 50](#_Toc216688978)

[4. Backend en PHP (Diagrama E-R) 51](#_Toc216688979)

[5. Diagrama Relacional 52](#_Toc216688980)

[6. Frontend para el Login 53](#_Toc216688981)

[Componentes de la ventana: 53](#_Toc216688982)

[7. Frontend para el panel principal del administrador 55](#_Toc216688983)

[Componentes de la ventana: 55](#_Toc216688984)

[8. Frontend para la ventana escanear del administrador 56](#_Toc216688985)

[Componentes de la ventana: 56](#_Toc216688986)

[9. Frontend para seleccionar el producto que se escaneara e indicar el número de Lote del administrador 57](#_Toc216688987)

[Componentes de la ventana: 57](#_Toc216688988)

[10. Frontend para el botón escanear del administrador 59](#_Toc216688989)

[Componentes de la ventana: 59](#_Toc216688990)

[11. Frontend para el botón registros del administrador 60](#_Toc216688991)

[Componentes de la ventana: 60](#_Toc216688992)

[12. Frontend para descargar el reporte de los productos del administrador 61](#_Toc216688993)

[Componentes de la ventana: 61](#_Toc216688994)

[13. Frontend para el botón añadir del administrador 62](#_Toc216688995)

[Componentes de la ventana 62](#_Toc216688996)

[14. Frontend para el botón de reportes del administrador 63](#_Toc216688997)

[Componentes de la ventana 63](#_Toc216688998)

[15. Frontend para el botón de generar reportes de acuerdo a la preferencia de impresión. 64](#_Toc216688999)

[Componentes de la ventana 64](#_Toc216689000)

[16. Frontend para la ventana inventarios del administrador 65](#_Toc216689001)

[Componentes de la ventana 65](#_Toc216689002)

[17. Frontend para la ventana de reportes del administrador. 67](#_Toc216689003)

[Componentes de la ventana 67](#_Toc216689004)

[18. Frontend para descargar el reporte de los productos del inventario del administrador 68](#_Toc216689005)

[Componentes de la ventana: 68](#_Toc216689006)

[19. Frontend para la ventana de usuarios registrados del administrador 69](#_Toc216689007)

[Componentes de la ventana 69](#_Toc216689008)

[20. Frontend para la ventana principal de inventarios del supervisor. 71](#_Toc216689009)

[Componentes de la ventana 71](#_Toc216689010)

[21. Frontend para la ventana principal del supervisor de calidad (Este usuario con rol de supervisor es el único y exclusivo que puede registrar usuarios de su área con rol de operador). 72](#_Toc216689011)

[Componentes de la ventana 72](#_Toc216689012)

[22. Frontend para el panel escanear del supervisor 73](#_Toc216689013)

[Componentes de la ventana 73](#_Toc216689014)

[23. Frontend para seleccionar el producto que se escaneara e indicar el número de Lote del supervisor 74](#_Toc216689015)

[Componentes de la ventana 74](#_Toc216689016)

[24. Frontend para el botón escanear del supervisor 75](#_Toc216689017)

[Componentes de la ventana 75](#_Toc216689018)

[25. Frontend para el botón de registros escaneados del supervisor 76](#_Toc216689019)

[Componentes de la ventana 76](#_Toc216689020)

[26. Frontend para la generación de reportes de los productos del botón escanear del supervisor 77](#_Toc216689021)

[Componentes de la ventana. 77](#_Toc216689022)

[27. Frontend para el botón añadir del Supervisor 78](#_Toc216689023)

[Componentes de la ventana 78](#_Toc216689024)

[28. Frontend para el botón de reportes de los productos añadidos del supervisor 79](#_Toc216689025)

[Componentes de la ventana 79](#_Toc216689026)

[29. Frontend para generar los reportes de los productos añadidos del supervisor 80](#_Toc216689027)

[Componentes de la ventana. 80](#_Toc216689028)

[30. Frontend para la ventana principal de los inventarios de cada área. 81](#_Toc216689029)

[Componentes de la ventana 81](#_Toc216689030)

[31. Frontend para el operador 83](#_Toc216689031)

[Componentes de la ventana: 83](#_Toc216689032)

[32. Frontend para seleccionar el producto que se desea escanear del operador. 84](#_Toc216689033)

[Componentes de la ventana: 84](#_Toc216689034)

[33. Frontend para escanear el producto seleccionado del operador 85](#_Toc216689035)

[Componentes de la ventana: 85](#_Toc216689036)

[34. Frontend para el registro de los escaneos realizados por parte del operador. 86](#_Toc216689037)

[Componentes de la ventana: 86](#_Toc216689038)

[35. Frontend para elegir las preferencias de impresión del reporte. 87](#_Toc216689039)

[Componentes de la ventana. 87](#_Toc216689040)

[36. Arquitectura del sistema 88](#_Toc216689041)

[Fase 3 89](#_Toc216689042)

[Implementación 89](#_Toc216689043)

[1. Elegir el SGBD a utilizar 89](#_Toc216689044)

[2. Codificación de la base de datos 90](#_Toc216689045)

[Tabla usuarios 90](#_Toc216689046)

[Login (Acceso y control de área y rol) 90](#_Toc216689047)

[Código ejecutado en phpMyAdmin 91](#_Toc216689048)

[Tablas Inventarios 91](#_Toc216689049)

[Tabla Categorías 91](#_Toc216689050)

[Código ejecutado en phpMyAdmin 92](#_Toc216689051)

[Tabla Campos 92](#_Toc216689052)

[Código ejecutado en phpMyAdmin 93](#_Toc216689053)

[Tabla Registros 93](#_Toc216689054)

[Código ejecutado en phpMyAdmin 94](#_Toc216689055)

[Tabla Productos\_Escanear 94](#_Toc216689056)

[Código ejecutado en phpMyAdmin 94](#_Toc216689057)

[Tabla Registro\_Escaneos 94](#_Toc216689058)

[Código ejecutado en phpMyAdmin 95](#_Toc216689059)

[Tabla Historial 95](#_Toc216689060)

[3. Configurar el lector SICAR Checko2B para enviar datos al sistema web. 96](#_Toc216689061)

[4. Programar el backend para recibir, validar y almacenar los datos. 99](#_Toc216689062)

[Conexión de la base de datos 99](#_Toc216689063)

[Inicio de Sesión (Index.html) 100](#_Toc216689064)

[5. Programar el frontend para mostrar información al administrador. 113](#_Toc216689065)

[6. Programar el frontend para mostrar información al supervisor. 113](#_Toc216689066)

[7. Programar el frontend para escanear del operador. 113](#_Toc216689067)

[8. Configurar la base de datos en XAMPP. 113](#_Toc216689068)

[9. Integrar la lógica de activación de alarmas desde el lector. 113](#_Toc216689069)

[Fase 4 114](#_Toc216689070)

[Verificación 114](#_Toc216689071)

[1. Pruebas de la base de datos 114](#_Toc216689072)

[2. Pruebas unitarias de cada módulo (escaneo, validación, almacenamiento). 114](#_Toc216689073)

[3. Pruebas de integración del sistema completo 114](#_Toc216689074)

[4. Pruebas de rendimiento de la velocidad de respuesta, carga de datos. 114](#_Toc216689075)

[5. Pruebas de error 114](#_Toc216689076)

[6. Validación con usuarios 114](#_Toc216689077)

[Fase 5 114](#_Toc216689078)

[Mantenimiento 114](#_Toc216689079)

[1. Corrección de errores 114](#_Toc216689080)

[2. Hacer actualizaciones 114](#_Toc216689081)

[3. Manual de usuario 114](#_Toc216689082)

[4. Manual técnico 114](#_Toc216689083)

**Índice de figuras**

[Figura 1 36](#_Toc212545435)

[Figura 2 37](#_Toc212545436)

[Figura 3 38](#_Toc212545437)

[Figura 4 39](#_Toc212545438)

[Figura 5 40](#_Toc212545439)

[Figura 6 41](#_Toc212545440)

[Figura 7 42](#_Toc212545441)

[Figura 8 43](#_Toc212545442)

[Figura 9 44](#_Toc212545443)

[Figura 10 45](#_Toc212545444)

[Figura 11 46](#_Toc212545445)

[Figura 12 48](#_Toc212545446)

[Figura 13 49](#_Toc212545447)

[Figura 14 50](#_Toc212545448)

[Figura 15 51](#_Toc212545449)

[Figura 16 52](#_Toc212545450)

[Figura 17 53](#_Toc212545451)

[Figura 18 54](#_Toc212545452)

[Figura 19 56](#_Toc212545453)

[Figura 20 57](#_Toc212545454)

[Figura 21 58](#_Toc212545455)

[Figura 22 60](#_Toc212545456)

[Figura 23 62](#_Toc212545457)

[Figura 24 64](#_Toc212545458)

[Figura 25 65](#_Toc212545459)

[Figura 26 66](#_Toc212545460)

[Figura 27 67](#_Toc212545461)

[Figura 28 68](#_Toc212545462)

[Figura 29 69](#_Toc212545463)

[Figura 30 70](#_Toc212545464)

[Figura 31 71](#_Toc212545465)

[Figura 32 72](#_Toc212545466)

[Figura 33 73](#_Toc212545467)

[Figura 34 74](#_Toc212545468)

[Figura 35 75](#_Toc212545469)

[Figura 36 77](#_Toc212545470)

[Figura 37 79](#_Toc212545471)

[Figura 38 80](#_Toc212545472)

[Figura 39 81](#_Toc212545473)

[Figura 40 82](#_Toc212545474)

[Figura 41 84](#_Toc212545475)

[Figura 42 86](#_Toc212545476)

[Figura 43 87](#_Toc212545477)

**Índice de Tablas**

[Tabla 1 33](#_Toc212545623)

# Introducción

En el entorno competitivo actual, la eficiencia en los procesos internos y la calidad en la presentación de los productos son factores determinantes para la permanencia y crecimiento de las empresas. En SlimPop, se han identificado problemáticas que afectan directamente la relación con sus clientes y cadenas comerciales, principalmente vinculadas a la legibilidad y correcta asignación de los códigos de barras en los productos terminados y embolsados, estas inconsistencias generan devoluciones e inconformidades, impactando negativamente en la confianza y satisfacción de los intermediarios comerciales.

Asimismo, se ha detectado una dificultad significativa en la gestión y control de inventarios, ya que actualmente se realizan mediante registros manuales en formatos impresos, este método no solo implica pérdida de documentación, sino también un considerable gasto de tiempo en la búsqueda de información en archivos físicos, lo que limita la eficiencia operativa.

Ante este panorama, se plantea el desarrollo e implementación de un sistema web integral, diseñado para optimizar la validación de códigos de barras y garantizar la entrega de productos con la calidad esperada, además de proporcionar herramientas modernas para la administración de inventarios. Con ello, SlimPop fortalecerá sus procesos internos, reducirá errores y ofrecerá soluciones tecnológicas que respalden su compromiso con la excelencia y la satisfacción de sus clientes.

**Capitulo II Generalidades del**

**proyecto**

# Descripción de la empresa u organización y del puesto o área del trabajo el estudiante

## Descripción de empresa

SLIMPOP es una empresa privada dedicada a la producción y venta de palomitas explotas con aire caliente hechas con su propio maíz producida de las tierras más fértiles de Estados Unidos, debido a la tendencia de éxito de la industria de palomitas en USA, en el año 2013 el director César Villalobos y Norm Krug, fundador de Preferred Popcorn USA deciden inaugurar una planta en México para traer este delicioso snack a nuestro país.

Así nace Preferred Popcorn México con sólo 10 personas en su equipo, hoy en día ya son más de 150 colaboradores enamorados de la marca y continúan creciendo en equipo e innovando en México con las primeras palomitas saludables explotadas con aire caliente de ingredientes naturales y con los mejores estándares del mercado.

Luego de 12 años el objetivo sigue siendo imaginar y desarrollar novedosos snacks que te sorprendan a ti como equipo, al mercado y al consumidor cuidando cada detalle de los procesos de calidad para que lo disfrutes creando momentos perfectos, siendo una empresa joven con actualizaciones y tendencias nuevas en sus productos.

## Puesto o Área de trabajo del Estudiante

* **Departamento de T.I (Tecnologías de la información)**

Las actividades que realiza el estudiante en el departamento T.I son principalmente del soporte técnico de la empresa atendiendo a los colaboradores cuando se requiere mantenimiento de los equipos de cómputo y laptops, así como las actualizaciones de software de equipos conforme se requiera, resolviendo también problemas técnicos que surgen en el transcurso de los días.

De igual forma el departamento se lleva a cabo el mantenimiento de la red, como añadir nuevos dispositivos para el control y amplificación de espacios con conexión, administrar el acceso a las redes que se manejan.

* **Mantenimiento de los equipos de impresión**

El mantenimiento y reposición de tinta o tóner de las impresoras se realiza cada dos semanas para asegurar que el colaborador obtenga una buena calidad de impresión en los distintos documentos que desea imprimir.

* **Implementación de mejoras continuas**

La implementación de mejoras continuas se lleva a cabo por parte del estudiante a la hora de realizar un nuevo software desarrollado desde cero, observando las diferentes problemáticas que lleguen a surgir durante el transcurso de los meses para poder darle solución y resultados efectivos a la empresa, implementando así el desarrollo del sistema web de inventarios y validación lectores de códigos la cual benefician a las áreas comunes y a producción.

* **Configuración de nuevos dispositivos**

La configuración de los dispositivos nuevos consiste en actualizar la versión más reciente de tal manera en la que se deben de conectar al servidor de red para mantenerlos protegidos, finalmente se asigna un dispositivo al personal autorizado con su carta responsiva.

* **Configuración de conexiones remotas**

La configuración de las conexiones remotas es una de las actividades que realiza el estudiante en esta área, puesto que mediante esta herramienta se realizan mantenimientos remotos a los equipos que se encuentran dentro de planta o mediante Team Viewer que es una de las herramientas que permite la manipulación de otro equipo de manera remota, de tal manera que para la configuración de los VPN para los equipos encontrados dentro de planta o fuera.

# Problemas para resolver, priorizándolos

La empresa SLIMPOP se dedicada a la producción y venta de palomitas, dentro de su proceso de producción cada producto debe ser revisado antes de ser puesto a la venta, verificando aspectos principales como su presentación y código de barras, este código funciona como una identificación única que permite llevar un control preciso del producto, sin embargo, en la actualidad la empresa no lleva este control de la verificación de códigos lo que genera errores en la identificación de los productos, afectando directamente el control de inventario, la logística y la trazabilidad de las ventas, por lo que si un código es incorrecto puede ocasionar pérdidas de tiempo, errores en la facturación y desajustes en el stock disponible, dificultando la correcta gestión de la producción y distribución.

Por otro lado, algunas de las áreas de la empresa presentan dificultades en la gestión y control de sus inventarios, ya que estos se realizan de forma manual, por lo que esta práctica ha ocasionado desorganización y falta de precisión en los registros de equipos y materiales en general, generando complicaciones al asignar un dispositivo electrónico, o en la asignación temporal de otro material, además, se desconoce con exactitud el stock disponible y cuantos están en uso o fuera de servicio debido a fallas eléctricas u otros motivos, afectando así la toma de decisiones y en el mantenimiento eficiente de los recursos.

Por lo tanto, el control de estos productos y materiales necesitan ser organizados y registrados de forma digital que permitan un control automatizado y eficiente tanto en el proceso de empaquetado y validación de productos como en la gestión de los inventarios de las áreas.

# Objetivos (General y Específicos)

## Objetivo General

Desarrollar un sistema web de inventarios y validación de códigos de barras que permita optimizar y reducir errores al momento de escanear el código de barras del producto, así mismo tener un control en sus inventarios de materiales y equipos de cómputo de la empresa SLIMPOP.

## Objetivos Específicos

* Realizar un levantamiento de requerimientos con el operador de la línea de producción.
* Analizar los métodos de registro de las diferentes operaciones.
* Determinar la metodología de desarrollo de software.
* Desarrollar una interfaz interactiva para el usuario.

## Alcances

* Implementación de un sistema que verifique la legibilidad y correspondencia de los códigos de barras en productos terminados y embolsados.
* Reducción de devoluciones e inconformidades por parte de clientes y cadenas comerciales.
* Digitalización de los registros de inventario en todas las áreas de la empresa.
* Acceso rápido y seguro a la información mediante consultas en tiempo real.
* Generación automática de reportes físicos y digitales para respaldo administrativo.
* Mejora en la trazabilidad de productos y lotes.
* Integración de un sistema web escalable que permita futuras mejoras y módulos adicionales.

## Limitaciones

* El sistema requiere acceso a internet y dispositivos compatibles para su uso.
* Posibles fallos de conectividad pueden afectar la disponibilidad del sistema.
* El proyecto se centra en la validación de códigos de barras y gestión de inventarios, sin incluir otros procesos administrativos o logísticos de la empresa.
* Se necesitará entrenamiento para que los usuarios adopten el sistema correctamente.

# Justificación

La implementación del sistema web en SlimPop se justifica por la necesidad de optimizar procesos críticos que actualmente afectan la calidad del producto y la eficiencia operativa de la empresa.

Los problemas relacionados con la ilegibilidad y asignación incorrecta de códigos de barras generan devoluciones y reclamos por parte de los intermediarios comerciales, lo que repercute directamente en la imagen de la marca y en la confianza de sus clientes.

Un sistema automatizado permitirá validar y controlar estos códigos de manera precisa, asegurando que cada producto cumpla con los estándares de calidad exigidos en el mercado.

Por otro lado, la gestión manual de inventarios mediante formatos impresos representa un riesgo constante de pérdida de información y un consumo excesivo de tiempo en la búsqueda de registros, esta práctica limita la capacidad de respuesta de la empresa y dificulta la toma de decisiones estratégicas, por lo que con un sistema web, la información podrá ser almacenada, consultada y actualizada en tiempo real, reduciendo errores y facilitando la generación de reportes confiables que respalden la administración de las distintas áreas.

En consecuencia, el desarrollo de esta solución tecnológica no solo responde a la necesidad de mejorar la eficiencia interna, sino que también fortalece la competitividad de SlimPop en el mercado, al garantizar productos con mayor calidad y procesos administrativos más ágiles, este sistema se traduce en beneficios tangibles como reducción de devoluciones, ahorro de tiempo, mejor control de inventarios y mayor satisfacción de los clientes.

**Capitulo III Marco teórico**

# Marco Teórico (fundamentos teóricos)

Antecedentes

Marco conceptual

Sistema web

Código de barras

Inventarios

Gestor de inventarios

Software

Lector de códigos

SICAR CHECKO2B

Programación

Código

HTML

CSS

PHP

JavaScript

Librerías

Bootstrap

XAMPP

PHPmyAdmin

FTP

CDMON

Hosting

Servidor

MySQL

Visual Studio Code

**Capitulo IV Desarrollo**

## Metodología CASCADA

La metodología cascada se define por su proceso secuencial paso a paso en la salida de cada fase alimenta a la siguiente por lo que su proceso es riguroso del desarrollo del software fluyendo de tal manera en la que no se retrocede a ninguna fase, sin embargo su estructura es clara y fácil de entender, de tal manera que el proyecto sea fácil de gestionar, dejando las tareas claramente definidas y con un documento limpio, esta metodología tradicional cuenta con 5 faces que a continuación se mencionan, siendo estas fundamentales para el desarrollo del sistema web dividiendo en secuencias donde cada fase debe de completarse en su totalidad.

**Análisis:** En esta fase se recopilan, documentan y analizan todos los requisitos del proyecto.

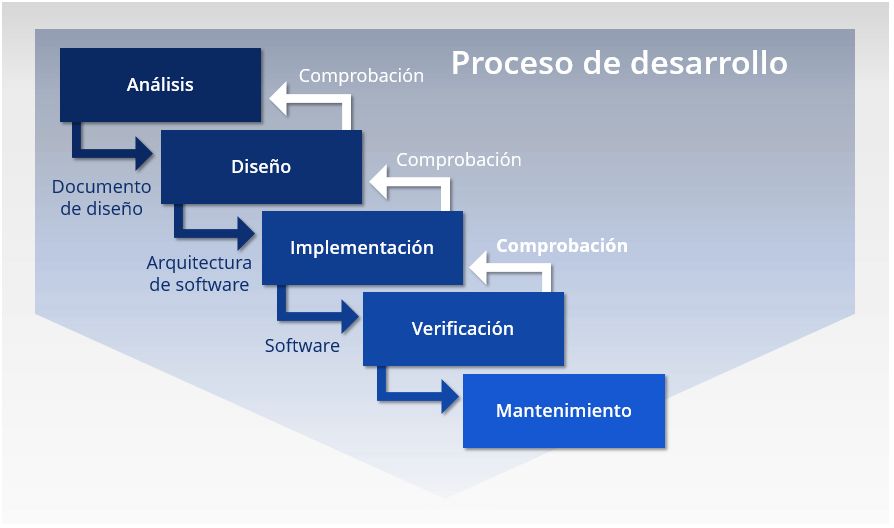
**Diseño:** En función a los requisitos, se crea un plan de diseño completo, es decir se incluye la arquitectura del sistema, diseños de interfaz, modelos de datos y más.

**Implementación:** Durante la implementación, se realizan el desarrollo o construcción propiamente dicha, donde cada componente se construye según el plan de diseño.

**Verificación:** También conocida como pruebas, esta etapa garantiza que el proyecto cumpla con los requisitos iniciales, cualquier otro error o problema se identifica y resuelve.

**Mantenimiento:** Tras la entrega del proyecto entra esta fase de mantenimiento donde aquí se aplican actualizaciones, optimizaciones y correcciones de errores basándose en la opinión de los usuarios o cambios del sistema. (Menzheha\_SaaSJet, 2025)

Figura 1

*Metodología CASCADA*

*Nota: La imagen representa las fases de la metodología CASCADA que se deben de seguir durante el desarrollo del proyecto.*

## Fases de la metodología

## Fase 1

## Análisis

* Herramienta de recolección de información.
* Generar la propuesta.
* Definir los requisitos funcionales del sistema.
* Analizar el lector de código de barras como protocolos soportados, configuración, documentación técnica.
* Establecer el flujo de trabajo en la línea de producción como el escaneo, Validación, Visualización y registro de inventarios.
* Determinar roles y permisos (administrador, operador, supervisor).
* Analizar la infraestructura de red y servidor.
* Identificar riesgos y limitaciones.

## Fase 2

## Diseño

* Crear diagramas de caso de uso.
* Diagramas secuenciales.
* Arquitectura cliente-servidor.
* Backend en PHP (Diagrama E-R).
* Diagrama Relacional.
* Frontend para el Login.
* Frontend para el panel principal del administrador.
* Frontend para la ventana escanear del administrador.
* Frontend para seleccionar el producto que se escaneara e indicar el número de Lote del administrador.
* Frontend para el botón escanear del administrador.
* Frontend para el botón registros del administrador.
* Frontend para descargar el reporte de los productos del administrador.
* Frontend para el botón añadir del administrador.
* Frontend para el botón de reportes del administrador.
* Frontend para el botón de generar reportes de acuerdo a la preferencia de impresión.
* Frontend para la ventana inventarios del administrador.
* Frontend para la ventana de reportes del administrador.
* Frontend para descargar el reporte de los productos del inventario del administrador.
* Frontend para la ventana de usuarios registrados del administrador.
* Frontend para la ventana de retito y devoluciones del administrador.
* Frontend para la ventana de devoluciones del administrador.
* Frontend para la ventana del historial del administrador.
* Frontend para la ventana principal de inventarios del supervisor.
* Frontend para el panel escanear del supervisor.
* Frontend para seleccionar el producto que se escaneara e indicar el número de Lote del supervisor.
* Frontend para el botón escanear del supervisor.
* Frontend para el botón de registros escaneados del supervisor.
* Frontend para la generación de reportes de los productos del botón escanear del supervisor.
* Frontend para el botón añadir del Supervisor.
* Frontend para el botón de reportes de los productos añadidos del supervisor.
* Frontend para generar los reportes de los productos añadidos del supervisor.
* Frontend para la ventana principal de los inventarios de TI del supervisor.
* Frontend para el botón de registrar productos de infraestructura de la ventana inventarios del supervisor.
* Frontend para el botón de registrar productos equipos de cómputo de la ventana de inventarios del supervisor.
* Frontend para la ventana de generar reportes del inventario del supervisor.
* Frontend para el botón de activos de material TI y generar reportes para el supervisor.
* Frontend para el inventario de suministros y consumibles para el supervisor.
* Frontend para la ventana de retito y devoluciones del supervisor.
* Frontend para la ventana de devoluciones del supervisor.
* Frontend para la ventana del historial del supervisor.
* Arquitectura del sistema.

## Fase 3

## Implementación

* Elegir el SGBD a utilizar
* Codificación de la base de datos
* Configurar el lector SICAR Checko2B para enviar datos al sistema web.
* Programar el backend para recibir, validar y almacenar los datos.
* Programar el frontend para mostrar información al administrador, supervisor y operador.
* Configurar la base de datos en XAMPP.
* Integrar la lógica de activación de alarmas desde el lector.

## Fase 4

## Verificación

* Pruebas de la base de datos.
* Pruebas unitarias de cada módulo (escaneo, validación, almacenamiento) e inventarios.
* Pruebas de integración del sistema completo.
* Pruebas de rendimiento de la velocidad de respuesta, carga de datos.
* Pruebas de error.
* Validación con usuarios.

## Fase 5

## Mantenimiento

* Corrección de errores.
* Hacer actualizaciones.
* Manual de usuario.
* Manual técnico.

## Cronograma de actividades

Tabla 1

*Planificación de cada una de las actividades que se deben de realizar.*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Fase** | **Tarea** | **Asignado a** | **Septiembre** | | | **Octubre** | | | | **Noviembre** | | | | **Diciembre** | | | | **Enero** | | | |
| **Identificación** | ***S2*** | ***S3*** | ***S4*** | ***S1*** | ***S2*** | ***S3*** | ***S4*** | ***S1*** | ***S2*** | ***S3*** | ***S4*** | ***S1*** | ***S2*** | ***S3*** | ***S4*** | ***S1*** | ***S2*** | ***S3*** | ***S4*** |
| **Análisis** | Herramienta de recolección de información. | Daniela Méndez    Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Generar la propuesta. | Daniela Méndez    Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Definir los requisitos funcionales del sistema. | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Analizar el lector de código de barras como protocolos soportados, configuración, documentación técnica. | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Establecer el flujo de trabajo en la línea de producción como el escaneo, Validación, Visualización y registro de inventarios. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Determinar roles y permisos (administrador, operador, supervisor). | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Analizar la infraestructura de red y servidor. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Identificar riesgos y limitaciones. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Diseño** | Crear diagramas de caso de uso. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Diagramas secuenciales. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Arquitectura del cliente | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Backend en PHP (Diagrama E-R) | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Diagrama relacional | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para el login | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para el panel principal del administrador | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para la ventana escanear del administrador | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para seleccionar el producto que se escaneara e indicar el número de lote del administrador | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para el botón escanear del administrador | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para el botón de registro del administrador | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para descargar el reporte de los productos del administrador | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para el botón añadir del administrador | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para el botón de reportes del administrador | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para el botón de generar reportes de acuerdo a la preferencia de impresión | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para para la ventana inventarios del administrador | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para la ventana de reportes del administrador | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para descargar el reporte de los productos del inventario del administrador | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para la ventana de usuarios registrados del administrador. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para la ventana de retito y devoluciones del administrador. | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para la ventana del historial del administrador. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para la ventana principal de inventarios del supervisor. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para la ventana principal del supervisor de calidad (Este usuario con rol de supervisor es el único y exclusivo que puede registrar usuarios de su área con rol de operador). | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para el panel escanear del supervisor. | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para seleccionar el producto que se escaneara e indicar el número de Lote del supervisor. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para el botón escanear del supervisor. | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para el botón de registros escaneados del supervisor. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para la generación de reportes de los productos del botón escanear del supervisor. | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para el botón añadir del Supervisor. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para el botón de reportes de los productos añadidos del supervisor. | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para generar los reportes de los productos añadidos del supervisor. | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para la ventana principal de los inventarios de cada área. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para el operador | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para seleccionar el producto que se desea escanear del operador. | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para escanear el producto seleccionado del operador | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para el registro de los escaneos realizados por parte del operador. | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Frontend para elegir las preferencias de impresión del reporte. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Implementación** | Elegir el SGBD a utilizar | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Codificación de la base de datos | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Configurar el lector SICAR Checko2B para enviar datos al sistema web. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Programar el backend para recibir, validar y almacenar los datos. | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Programar el frontend para mostrar información al administrador, supervisor y operador. | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Configurar la base de datos en XAMPP. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Verificación** | Pruebas de la base de datos. | Daniela Méndez    Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Pruebas unitarias de cada módulo (escaneo, validación, almacenamiento) e inventarios. | Daniela Méndez    Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Pruebas de integración del sistema completo. | Daniela Méndez    Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Pruebas de rendimiento de la velocidad de respuesta, carga de datos. | Daniela Méndez    Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Pruebas de error. | Daniela Méndez    Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Validación con usuarios. | Daniela Méndez    Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Mantenimiento** | Corrección de errores. | Daniela Méndez    Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Hacer actualizaciones. | Daniela Méndez    Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Manual de usuario. | Omar Flores |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Manual técnico. | Daniela Méndez |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

*Nota:* Un cronograma es una herramienta para la gestión del proyecto que detalla las acciones y fechas necesarias para la creación del sistema web, describiendo los puntos necesarios para garantizar que el sistema web satisfaga las expectativas del cliente.

## Fase 1

## Análisis

### Herramienta de recolección de información.

* Levantamiento de requerimientos

Para poder llevar a cabo el desarrollo del sistema web se necesitó hacer un levantamiento de requerimientos con el operador de la línea de producción de las palomitas SLIMPOP y con los gerentes de cada área para recabar la información fundamental de sus inventarios, revisando a si los manuales técnicos del lector de códigos de barras SICAR Checko2B que se consideran una opción a usar, verificando también su compatibilidad y las especificaciones de los mismos.

### Generar la propuesta

Se propone el desarrollo de un sistema web que automatice el proceso de escaneo de productos en la línea de producción y así mismo integrar este módulo de escaneo en el inventariado de cada área, el sistema recibirá los datos del lector de códigos de barras SICAR Checko2B, validará los códigos escaneados y registrará la información en una base de datos.

El administrador podrá acceder a un panel de control donde visualizará y editar los campos del inventario en general de todas las áreas que tiene la empresa, el registro de los usuarios, las actualizaciones y eliminación de los productos, así como también podrá hacer escaneos, reportes detallados por lote, fecha, hora, código del producto, por otra parte el Supervisor solo tendrá accesos limitados a los inventarios, como el escaneo, generar reportes, registro de inventarios de su área en especifica, finalmente el operador solo escaneara productos, esta solución busca reducir errores, mejorar y optimizar el tiempo.

### Definir los requisitos funcionales del sistema.

El sistema estará contando con requisitos específicos que son fundamentales para el funcionamiento.

* Escaneo de códigos de barras desde el lector.
* Validación automática del código escaneado.
* Registro de productos válidos y errores en la base de datos.
* Gestión de inventarios por cada área.
* Generación de reportes por fecha, usuario y tipo de producto.
* Control de acceso por roles definidos.

### Analizar el lector de código de barras como protocolos soportados, configuración.

Tabla 2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Nombre del lector | Lenguaje | Función | Compatibilidad | Componentes | Librerías | Interfaces compatibles |
| SICAR Checko2B | JavaScript + CSS | El lector actúa como si fuera un teclado cada vez que escanea un código, lo “escribe” en el campo activo de la ventana. | Si es adecuado para el sistema en JavaScript solo se necesita configurar en modo teclado para que funcione sin complicaciones. | Frontend (JavaScript)  Backend (PHP con XAMPP)  Lector Checko2B  Entrada de datos vía teclado USB o Bluetooth. | SweetAlert2 para alertas visuales de éxito/error  Script para la secuencia de instrucciones ejecutadas en un entorno especifico.  API para enviar datos al backend. | Interfaz serie RS-232 Teclado USB Interfaz serie virtual USB Bluetooth. |

*Lector de códigos de barras SICAR Checko2B y sus características principales.*

*Nota:* Se muestran las características principales del lector de código de barras SICAR Checko2B.

### Establecer el flujo de trabajo en la línea de producción como el escaneo, Validación, Visualización y registro de inventarios.

El flujo de trabajo del sistema será el siguiente:

**Modulo escaneo**

* El producto pasa frente al lector de códigos de barras SICAR Checko2B.
* El lector escanea el código de barras del producto.
* El lector envía el código escaneado al sistema web, actuando como teclado (USB/Bluetooth).
* JavaScript en el frontend captura el código y lo envía al servidor mediante una solicitud HTTP.
* El backend (PHP) recibe el código y lo valida contra la base de datos MySQL.
* Si el código es válido lo registra como exitoso y continua su proceso, pero si el código es inválido emite una alerta sonora y visual, la cual el programa marca como fallido.
* El sistema guarda el registro del escaneo.
* El administrador accede al panel web para consultar registros históricos filtrar por fechas, productos y generar reportes.

**Modulo inventario**

* El administrador accede a los inventarios para registrar, actualizar, eliminar, generar reportes y consultar usuarios registrados.

### Determinar roles y permisos (administrador, operador).

El sistema contará con un Login que permitirá el acceso a tres tipos de usuarios la cual será el administrador, Supervisor y finalmente el operador.

***Administrador:*** Usuario con acceso completo al sistema, podrá consultar todos los registros de los escaneos e inventarios, actualizará y eliminará, generar reportes por fecha, producto, lote, operador y gestionar usuarios.

***Supervisor:*** Usuario con acceso limitado al sistema, podrá consultar registros de su área y actualizarlas, generar reportes por fecha, escanear el producto.

***Operador:*** Usuario encargado de supervisar y realizar el escaneo en la línea de producción, solo tendrá acceso un módulo que corresponde al de escanear, donde podrá visualizar el estado de los escaneos en tiempo real, sin embargo, no podrá modificar ni consultar reportes históricos.

La autenticación se realizará mediante credenciales únicas, y se implementará control de acceso basado en roles para garantizar la seguridad de la información.

1. **Infraestructura de Red y Servidor**

El sistema funcionará en una red local (LAN) donde el lector SICAR Checko2B se conectará por USB o Bluetooth a una computadora, el servidor local estará configurado con XAMPP, el cual alojará tanto el sistema web como la base de datos, los dispositivos dentro de la red podrán acceder permitiendo el acceso multiusuario con roles definidos, lo que facilitará la gestión de permisos y tareas dentro del sistema, la infraestructura está preparada para escalar a un entorno en la nube sin afectar la operatividad del sistema.

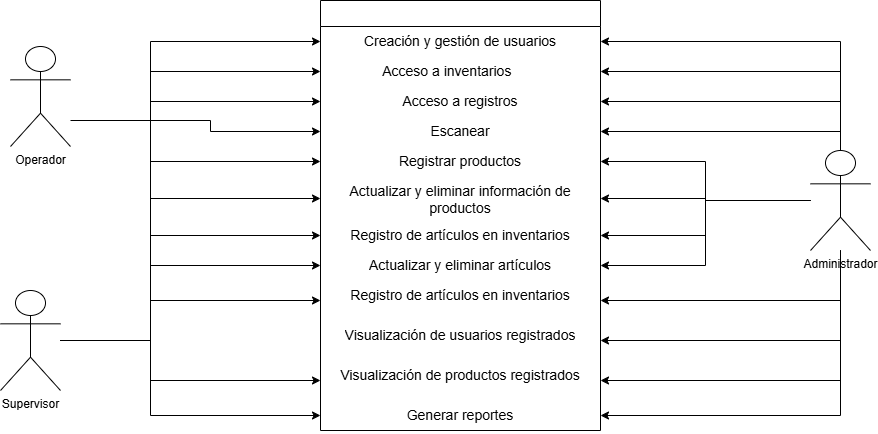
## Fase 2

## Diseño

### Crear diagramas de caso de uso

Figura 2

*Diagrama del caso de uso del sistema web POPCODE*

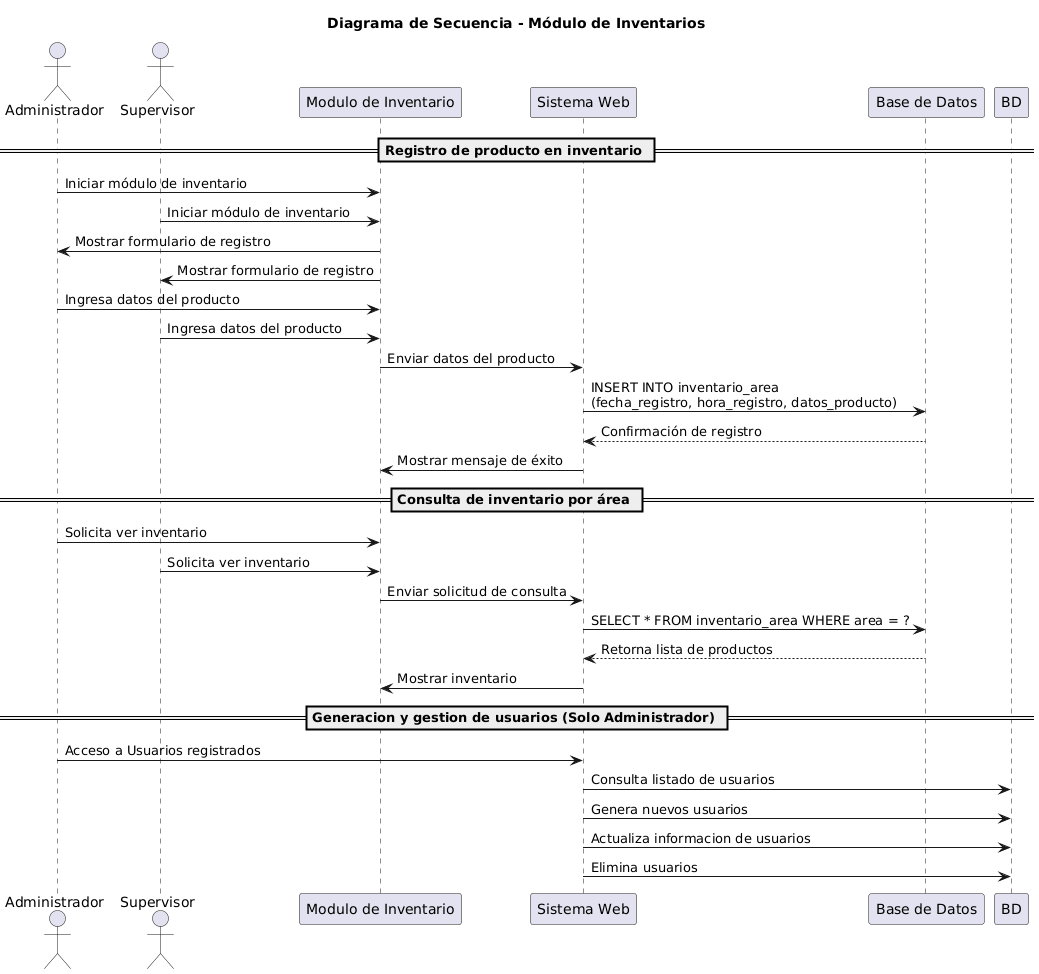


*Nota:* El diagrama representa una secuencia de lo que es la funcionalidad principal del sistema desde la perspectiva del usuario por cada rol, capturando los requisitos del sistema, y facilitar la comunicación.

### Diagramas secuenciales

Figura 3

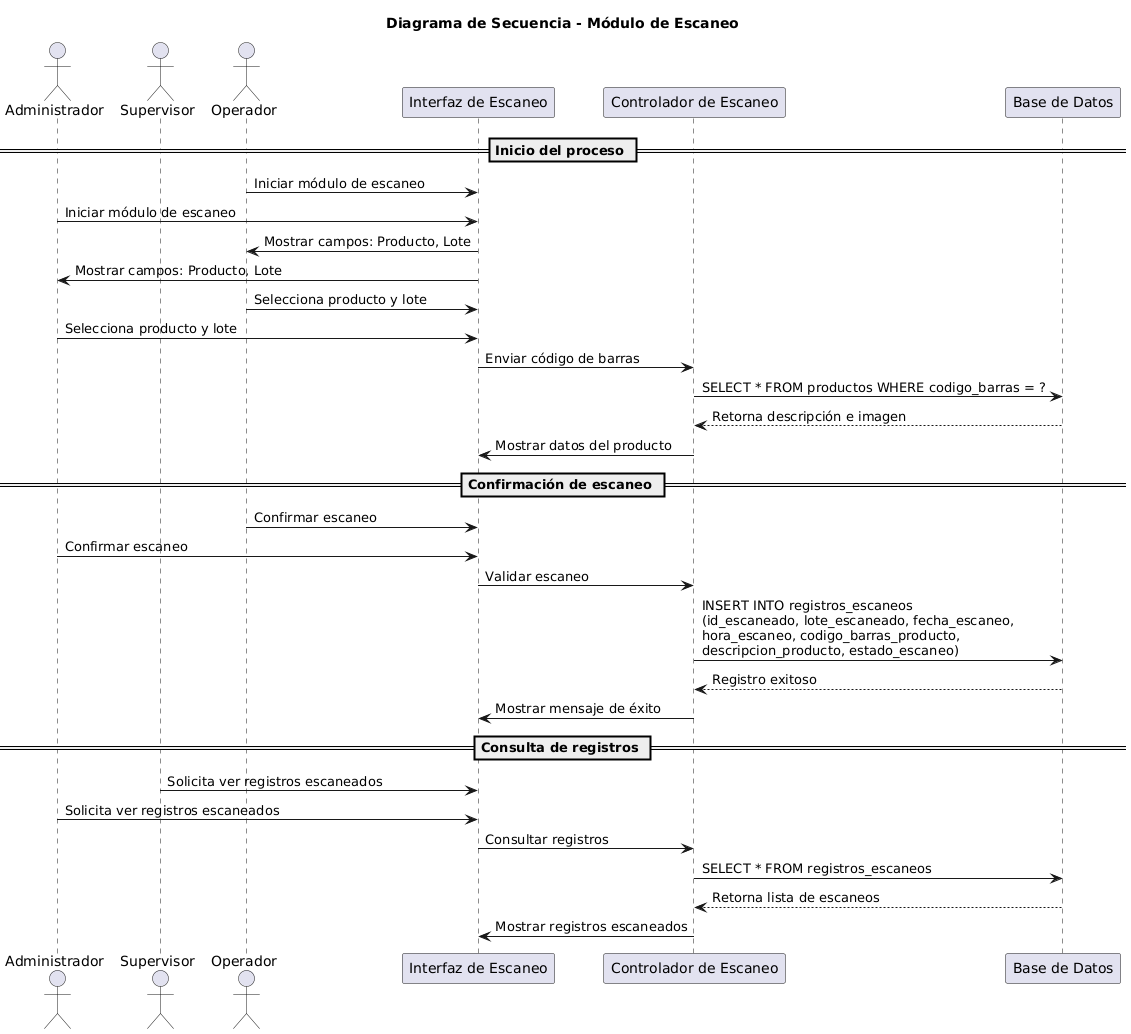
*Diagrama secuencial del módulo de inventarios*



*Nota:* El diagrama secuencial hace una representación de cómo es que se lleva el proceso del sistema y como interactúa, visualiza y documenta la interacción ordenada de objetos a lo largo del tiempo, mostrando cómo los mensajes son intercambiados entre ellos para cumplir con un caso de uso específico.

Figura 4

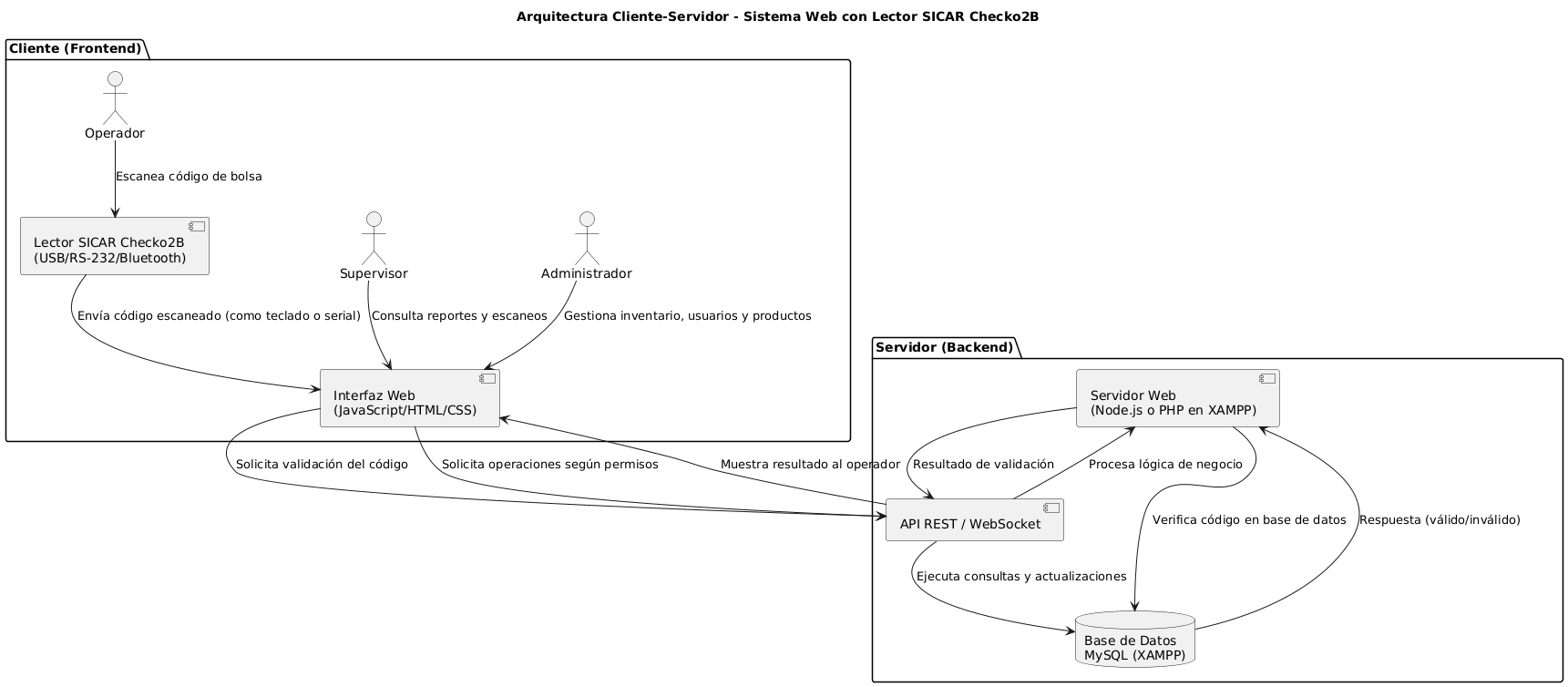
*Diagrama secuencial del módulo de escaneo de productos*



*Nota:* El diagrama secuencial se representa como el proceso del sistema y debe de interactuar, visualiza y documentar la interacción ordenada de objetos a lo largo del tiempo.

### Arquitectura cliente-servidor

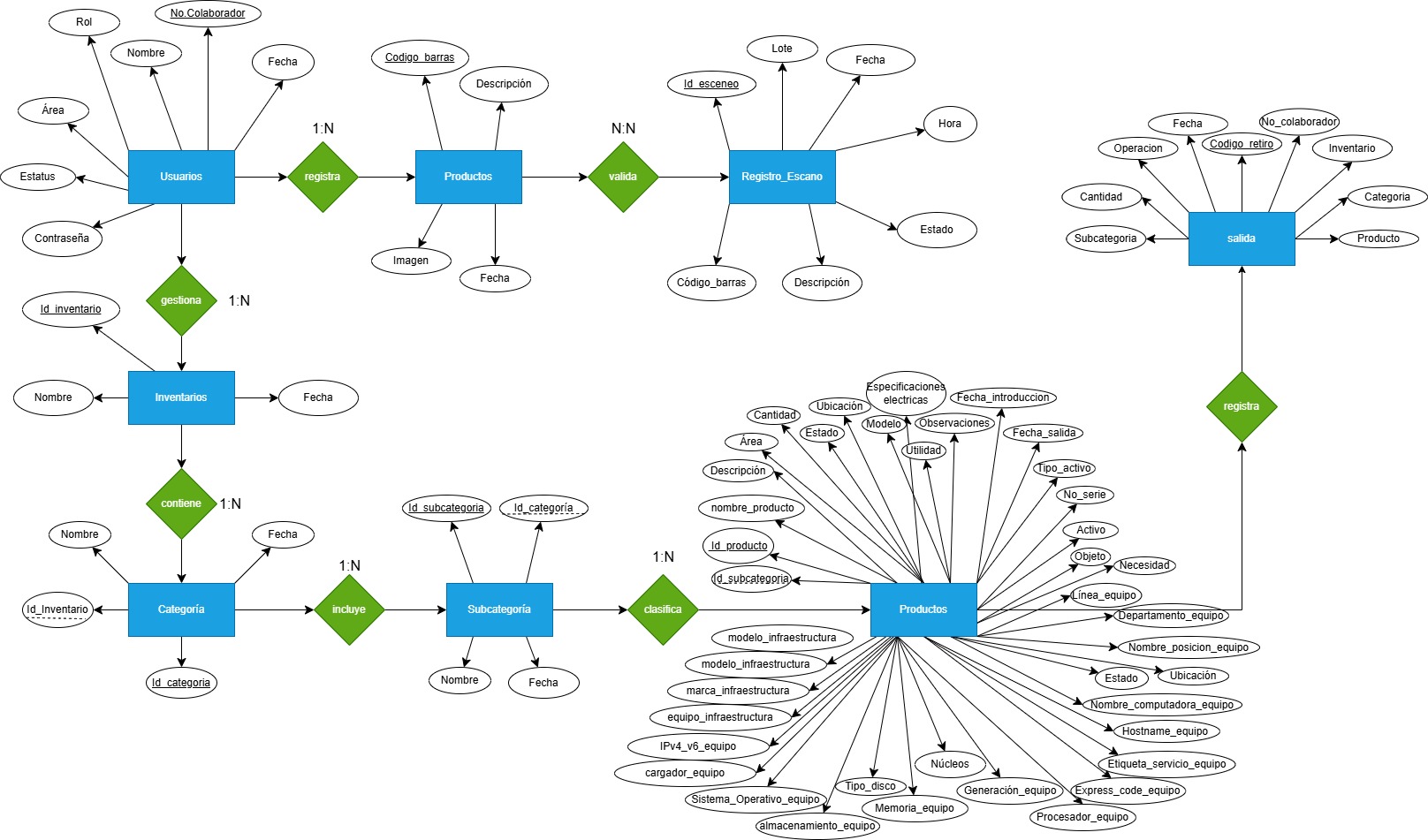
Figura 5

*Arquitectura de cliente-servidor del sistema web POPCODE*

*Nota:* La arquitectura del cliente-servidor se divide y organiza el sistema en responsabilidades que le corresponden como al cliente que solicita información y los servidores la proporcionan y gestionan los datos de manera centralizada.

### Backend en PHP (Diagrama E-R)

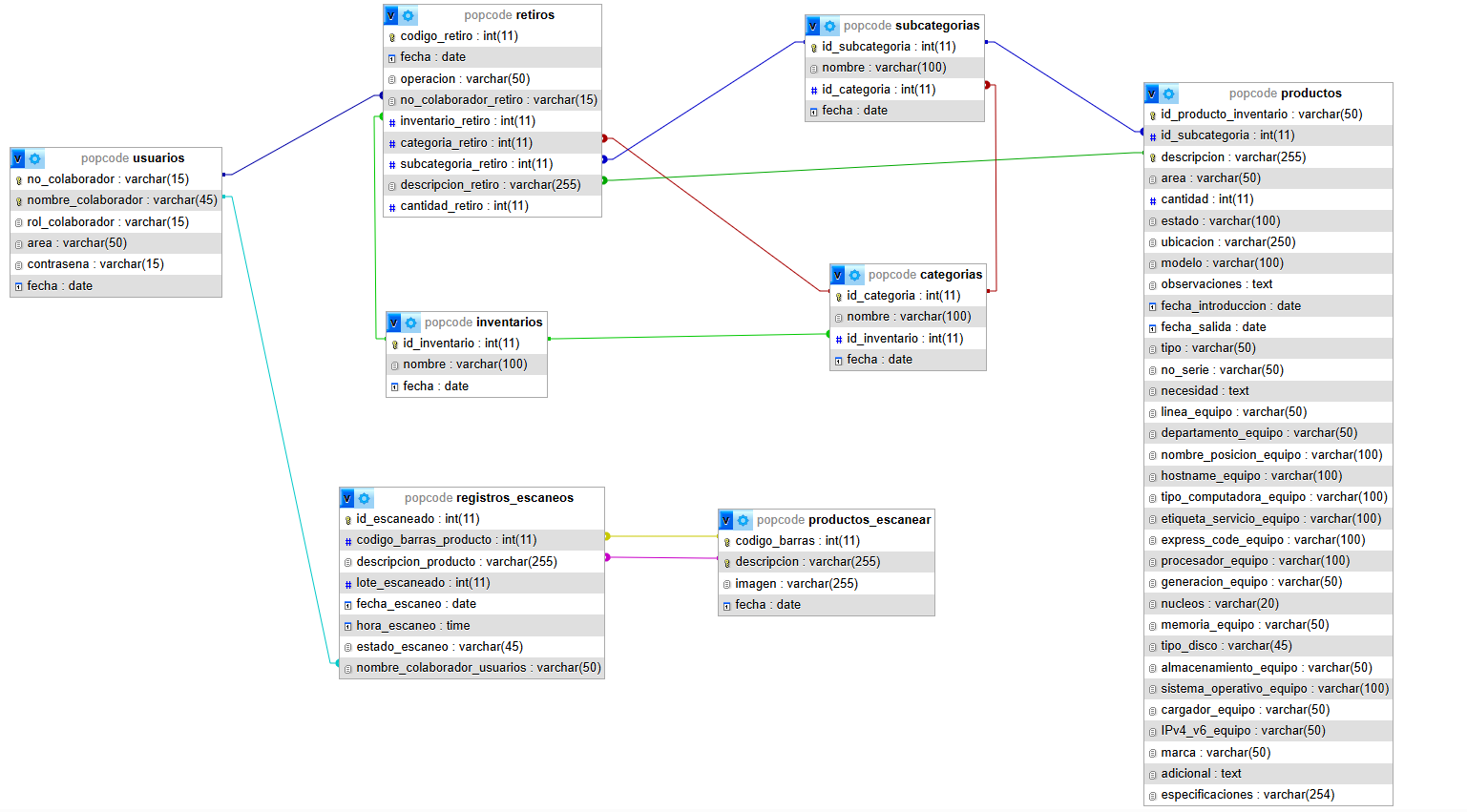
Figura 6

*Diseño del diagrama entidad relación correspondiente a los campos de la Base de Datos.*

*Nota:* El diagrama de entidad relación permite ver de manera organizada y estructurada la información y los campos a utilizar en la base de datos como las relaciones, los atributos y las claves primarias como foránea.

### Diagrama Relacional

Figura 7

*Diagrama relacional correspondiente las tablas de la base de datos a utilizar.*

*Nota:* El diagrama relacional permite visualizar las tablas de manera organizada y conectada entre sus claves correspondientes.

### Frontend para el Login

Figura 8

*Interfaz gráfica de la ventana del Login para acceder al sistema.*

### Componentes de la ventana:

* Campo texto “Usuario”: Campo para introducir el nombre del colaborador.
* Campo de texto “Contraseña”: Campo para introducir la contraseña del usuario y que en la base de datos la almacene de manera cifrada.
* Botón “Iniciar Sesión”: Botón que te va a permitir ingresar al sistema.



*Nota:* Se visualiza la interfaz del inicio de sesión.

Figura 9

*La interfaz gráfica muestra un mensaje de bienvenida.*

*Nota:* Mensaje de bienvenida al sistema y presentación.

### Frontend para el panel principal del administrador

Figura 10

*La interfaz gráfica del panel principal del administrador.*

### Componentes de la ventana:

* Botón “Escanear”: Acceder a módulo de escaneo.
* Botón “Inventarios”: Acceso a los inventarios con los que cuenta el administrador de esa área.
* Botón “Reportes”: Acceso a panel de selección del reporte que desea descargar ya sea de sus inventarios o de sus escaneos.
* Botón “Cerrar sesión”: Cierra sesión
* Botón “Usuarios”: Visualización de los usuarios que están registrados en el sistema.

*Nota:* Se visualiza la interfaz principal que tendrá el sistema a la hora de iniciar sesión.

### Frontend para la ventana escanear del administrador

Figura 11

*La interfaz gráfica de la ventana escanear del administrador.*

### Componentes de la ventana:

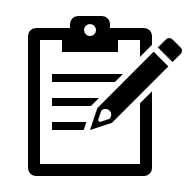
* Botón “Escaneo”: Botón para ingresar a el apartado de escaneo.
* Botón “Registrar”: Botón para registrar nuevos productos para su próximo escaneo.
* Botón “Registrados”: Botón para acceder al apartado de productos registrados.
* Botón “Cerrar sesión”: Botón para cerrar la sesión.

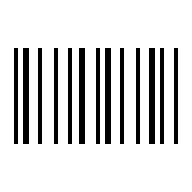


**¿Qué deseas realizar?**

**¡Bienvenido, \_\_\_\_\_\_\_\_\_!**

**Cerrar Sesión**

Añadir

Escaneo

*Nota:* En esta interfaz se estará mostrando una siguiente ventana que permite desplazarse por las siguientes interfaces de escaneo, registro y productos registrados.

### Frontend para seleccionar el producto que se escaneara e indicar el número de Lote del administrador

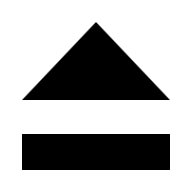
Figura 12

*La interfaz gráfica de la ventana que permite seleccionar el producto a escanear y el número de lote.*

### Componentes de la ventana:

* ComboBox “Producto a escanear”: Selección de producto que se escaneara mostrando una lista de productos.
* Campo de texto “No. Lote”: Ingreso del número del lote que se escaneara por medio de una nueva subventana que permite ingresar el tipo de producto “T, TP”, Turno “Matutino (A), Vespertino (B), Nocturno (C)”, Línea “1,2,3,4” y generar automáticamente y se ingresa al campo de texto
* Campo de texto: “Quien realiza la operación”: Permite mostrar a la persona quien realiza la operación
* Botón “Cancelar”: Botón para cancelar la operación y volver a la sección anterior.
* Botón “Continuar”: Botón para continuar el proceso.

**Forma, Rectángulo

El contenido generado por IA puede ser incorrecto.**

**¿Qué producto escanearas hoy?**

Palomitas XXXX 16G

Palomitas XXXX 16G

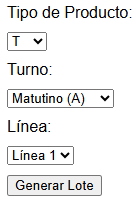
Producto a escanear

No. Lote

Continuar

Cancelar

Quien realiza la operación



*Nota:* En esta interfaz permite que el administrador seleccione el producto que desea escanear indicando el número de lote que tendrán los productos escaneados.

### Frontend para el botón escanear del administrador

Figura 13

*La interfaz gráfica del botón escanear donde se muestras los datos del producto escaneado.*

### Componentes de la ventana:

* Campo de texto “Código de barras”: Este campo permite mostrar el código de barras que se está escaneando.
* Campo de texto “Descripción”: Este campo se llena automáticamente al ingresar el código de barras.
* Botón “Regresar”: Botón para regresar al panel anterior del producto a escanear.
* Botón “Registros”: Botón para acceder a los registros de esca neos.

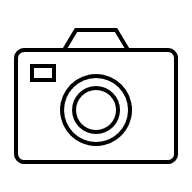
**Imagen que contiene Patrón de fondo

El contenido generado por IA puede ser incorrecto.**

Código de barras:

Descripción:

Imagen:

****

Regresar

Registros

IMG 600x900

*Nota:* En esta interfaz se mostrará los datos ya escaneados según su correspondencia.

### Frontend para el botón registros del administrador

Figura 14

*La interfaz gráfica del botón que muestra la anterior ventana de registros.*

### Componentes de la ventana:

* Tabla de “Registro de escaneos”: Tabla donde se almacenan todos los escaneos conforme a ID, Lote, Escaneo, Fecha, Hora, Código de barras, Descripción y Estado.
* Botón “Generar reporte”: Acceso a el apartado de preferencias del reporte.
* Botón “Regresar”: Retorno al panel anterior.

*Nota:* La interfaz que va a permitir generar los reportes de los productos que ya están escaneados.

### Frontend para descargar el reporte de los productos del administrador

Figura 15

*La interfaz gráfica muestra la selección de descarga del reporte por fecha.*

### Componentes de la ventana:

* Campo de texto “Fecha inicio”: Fecha inicial para conteo de fechas.
* Campo de texto “Fecha final”: Fecha final para conteo de fechas.
* CheckBox “Con Lote”: Para incluir los lotes en el reporte.
* CheckBox “Con Horas”: Para incluir la hora en el reporte.
* Botón “Generar”: Para la generación del reporte

**Imagen que contiene Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.**

**Preferencias de reporte**

**Orden**

Fecha: Desde Hasta

Con Lote Con Hora

Cancelar

Generar

Operador

Producto

Formato:

PDF

XLSX

*Nota:* En esta interfaz se puede elegir la fecha de preferencia para poder generar el reporte, así como el lote, el producto y los formatos.

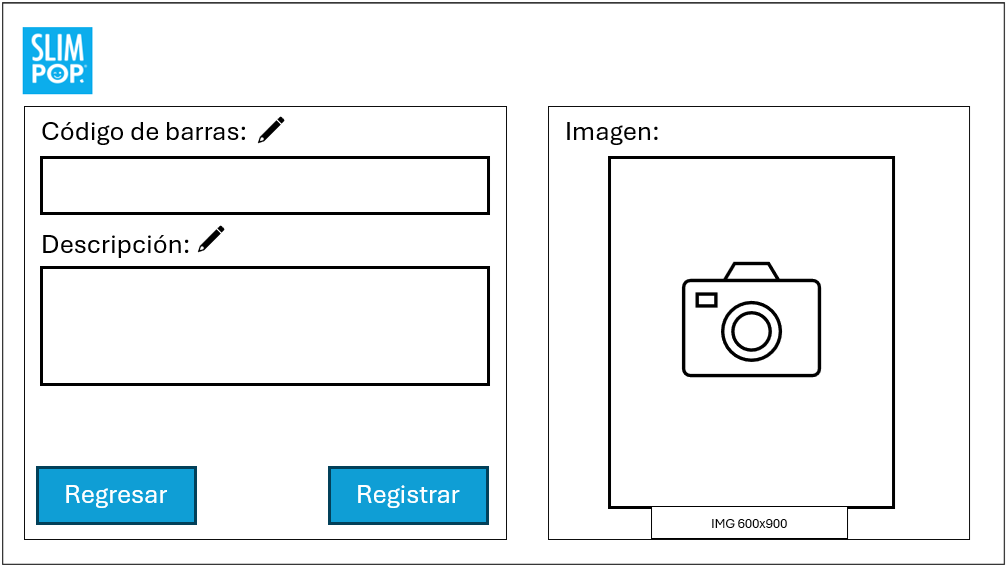
### Frontend para el botón añadir del administrador

Figura 16

*La interfaz gráfica muestra el botón donde se deben de registrar los productos.*

### Componentes de la ventana

* Campo de texto “Código de barras”:
* Campo de texto “Descripción”:
* Botón “Regresar”: Botón para regresar al panel anterior.
* Botón “Registrar”: Botón para registrar el producto.
* Botón “Reportes”: Permite ingresar a una nueva ventana donde se muestran os productos registrados.



**Reportes**

*Nota:* La interfaz muestra los campos para poder registrar un producto que se escaneara próximamente.

### Frontend para el botón de reportes del administrador

Figura 17

*La interfaz gráfica muestra la tabla de los productos registrados.*

### Componentes de la ventana

* Campo de texto “Filtro de datos por código”: Este campo permite buscar un producto en específico.
* Botón “Editar”: Permite editar algún producto.
* Botón “Eliminar”: Este campo elimina los productos.
* Botón “Generar reporte”: Este campo manda a la ventana de preferencia donde este permite imprimir de acuerdo a lo que se solicite.
* Tabla: Muestra los campos que anteriormente se escanearon, dando las acciones de editar algún campo o de eliminar.



*Nota:* En la interfaz gráfica se muestra la tabla donde se guardan los productos que anteriormente se escanearon.

### Frontend para el botón de generar reportes de acuerdo a la preferencia de impresión.

Figura 18

*La interfaz muestra la ventana de preferencia a descargar el reporte.*

### Componentes de la ventana

* Campo de texto “Fecha inicio”: Fecha inicial para conteo de fechas.
* Campo de texto “Fecha final”: Fecha final para conteo de fechas.
* Botón “Generar”: Para la generación del reporte
* Botón “Producto”: Para imprimir por producto de preferencia
* Botón “Formato”: Imprimir en formato de preferencia.

**Imagen que contiene Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.**

**Preferencias de reporte**

**Orden**

Fecha: Desde Hasta

Cancelar

Generar

Producto

Formato:

PDF

XLSX

*Nota:* En esta interfaz se puede elegir la fecha de preferencia para poder generar el reporte, así como por producto o el formato de acuerdo a su elección.

### Frontend para la ventana inventarios del administrador

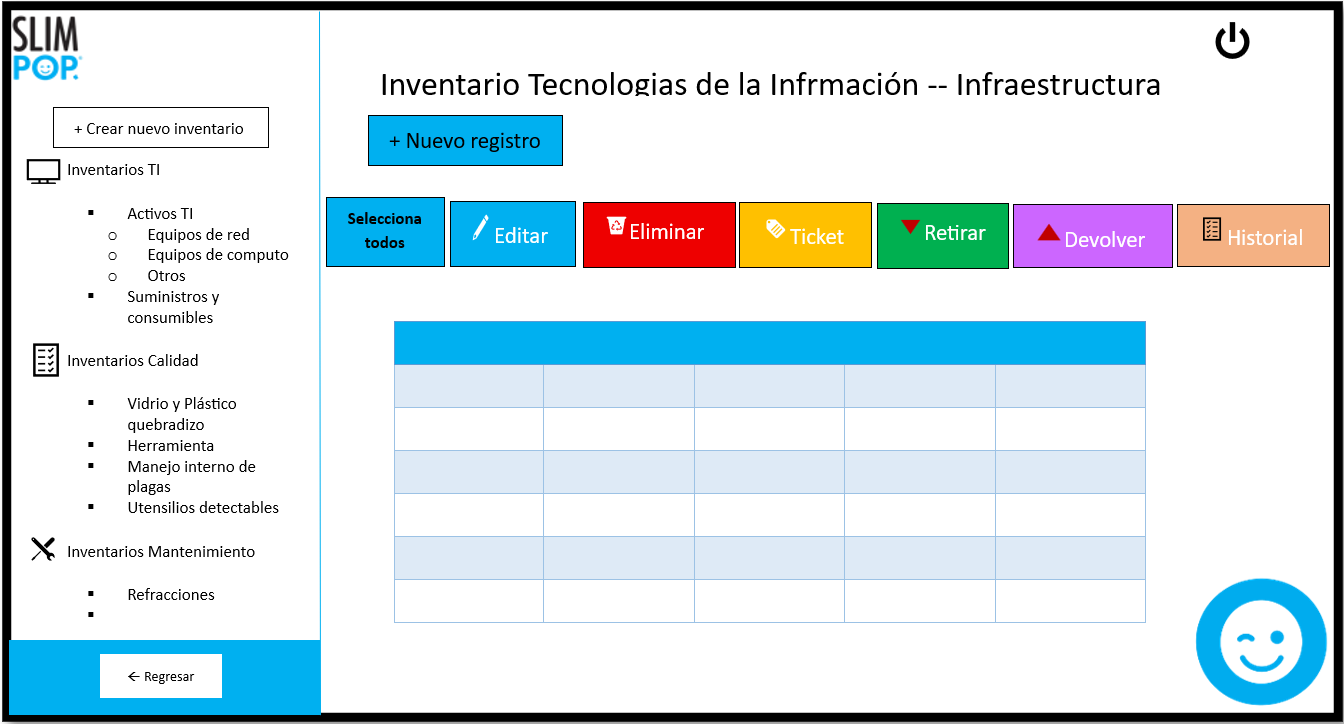
Figura 19

*La interfaz gráfica muestra los inventarios que se debe de registrar cada producto en cada área.*

### Componentes de la ventana

* Menú: Se muestran todos los inventarios de cada área donde en cada uno de estos se pueden registrar los productos para guardarlos en la base de datos.
* Botón “Crear nuevo inventario”: Funciona para crear un nuevo inventario de un área específica.
* Botón “Regresar”: Permite regresar a la página anterior.
* Botón “Nuevo registro”: Permite ingresar un nuevo registro a la base de datos.
* Botón “Seleccionar todos”: En este botón se permite seleccionar todos los registros existentes en la tabla.
* Botón “Editar”: Una vez que se selecciona una sola fila se abre el panel de editar el campo.
* Botón: “Eliminar” Una vez seleccionado se puede eliminar el registro de la tabla.
* Botón “Ticket”: Funciona para generar el ticket de cada producto imprimiendo el número del producto.
* Botón “Retirar”: Este botón funciona para hacer algún retiro de producto tan solo ingresando el código de retiro y la cantidad.
* Botón “Devolver”: Mediante este botón se puede el producto o material que anteriormente se haya retirado tan solo ingresando el código de retiro y la cantidad a devolver.
* Tabla: Permite mostrar todos los registros que se hayan realizado.

Advertencia: Si falta algún campo falta por rellenarse al dar clic en “Guardar” se emitirá una alerta visual con la leyenda “Rellene todos los campos para continuar”.



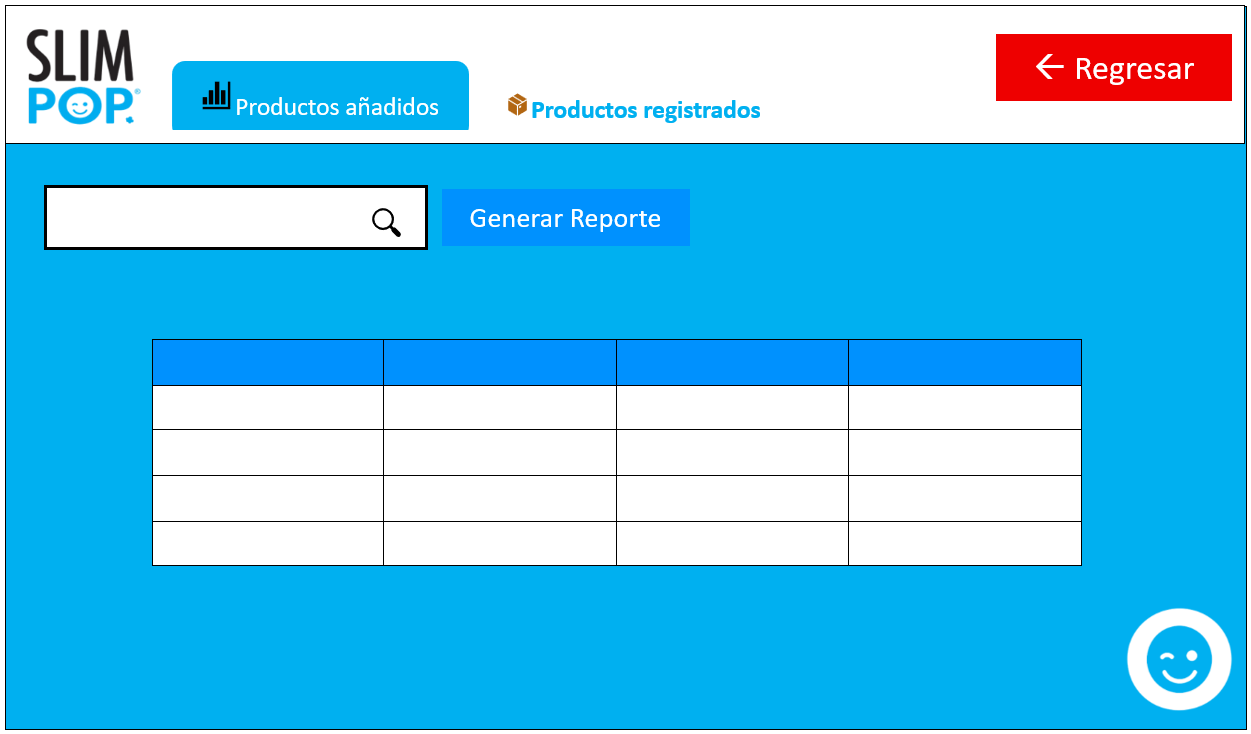
*Nota:* En esta interfaz se muestra un menú que contiene los inventarios de registro de todas las áreas, con sus respectivos movimientos que corresponden a la de editar, eliminar, ticket, retirar, devolver y el historial.

### Frontend para la ventana de reportes del administrador.

Figura 20

*La interfaz gráfica muestra el listado de productos de inventarios de cada área.*

### Componentes de la ventana

* Menú: Se muestran todos los inventarios de cada área y del módulo de escaneó clasificados y listos para poder ser impresos de acuerdo a las necesidades de usuario.
* Campo “Filtro por búsqueda”: Este permite hacer la búsqueda de algún producto más eficiente y de manera rápida.
* Botón “Generar reporte”: Este campo manda a la ventana de “Preferencias” donde este permite imprimir de acuerdo a lo que se solicite.

*Nota:* En esta interfaz se muestra la lista de productos que ya se tienen registrados por cada inventario de cada área, donde ya se puede hacer el reporte de estos mismos productos.

### Frontend para descargar el reporte de los productos del inventario del administrador

Figura 21

*La interfaz gráfica muestra la selección de descarga del reporte por fecha.*

### Componentes de la ventana:

* Campo de texto “Fecha inicio y Fecha final”: Fecha inicial para conteo de fechas.
* ComboBox “Formatos”: Se permite descargar el archivo de acuerdo a los dos tipos de formatos que se desee.
* Botón “Generar”: Para la generación del reporte

**Imagen que contiene Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.**

**Preferencias de reporte**

**Orden**

Fecha: Desde Hasta

Cancelar

Generar

Producto



Formato:

PDF

XLSX

*Nota:* En esta interfaz se puede elegir la fecha de preferencia para poder generar el reporte y el producto que se desee imprimir, pero va a cambiar las preferencias de acuerdo al inventario o registro de escaneo se esté seleccionado.

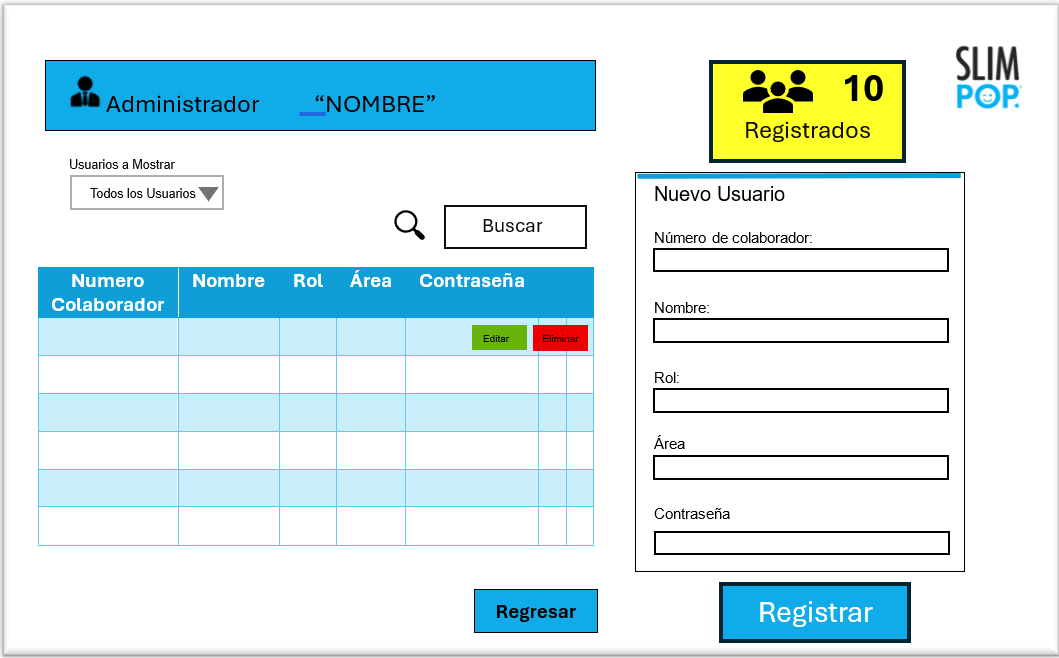
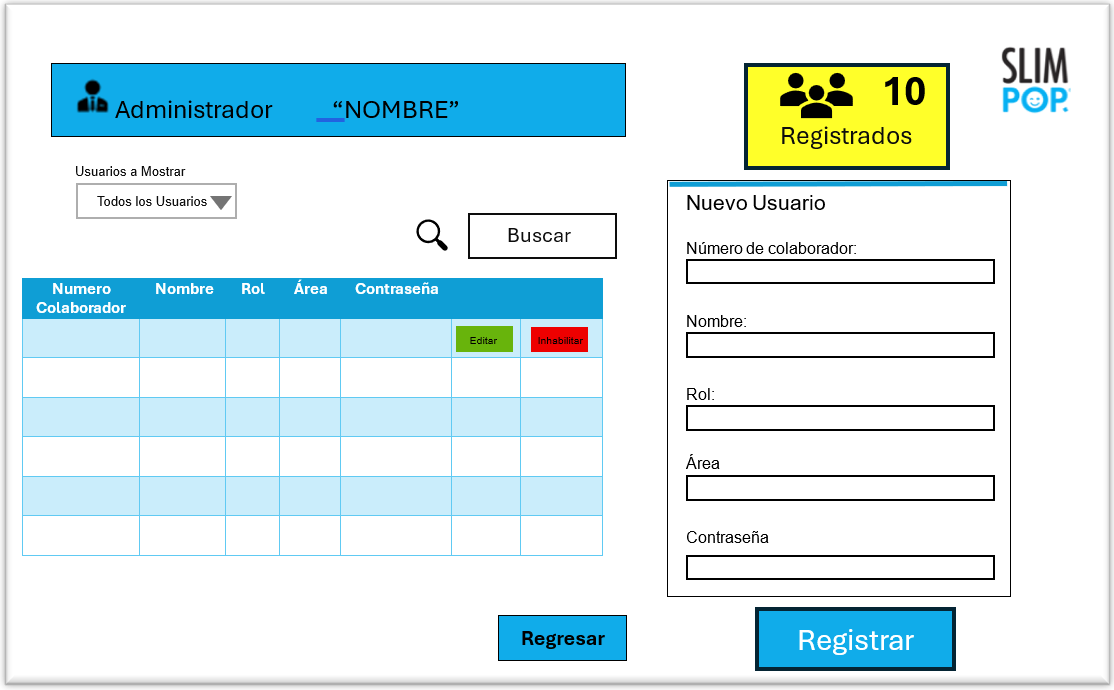
### Frontend para la ventana de usuarios registrados del administrador

Figura 22

*La interfaz gráfica muestra los registros de los usuarios existentes.*

### Componentes de la ventana

* Combo box “Usuarios a mostrar”: Selección de los tipos de usuarios que se desean mostrar como “Administrador, Supervisor, Operador”
* Campo de texto “Buscar”: Campo para realizar consultas de usuarios que están registrados por nombre y número de colaborador.
* Campo de texto “Numero de colaborador”: Ingreso del número de colaborador.
* Campo de texto “Nombre”: Campo para ingresar el nombre del colaborador.
* Campo de texto “Rol”: Ingreso del rol que tiene el colaborador.
* Campo de texto “Área”: Ingreso del área a la que pertenece el colaborador.
* Campo de texto “Contraseña”: Ingreso de la contraseña del colaborador.
* Botón “Registrar”: Botón para registrar usuarios.
* Botón “Regresar”: Botón para regresar al panel anterior.
* Botón “Cerrar sesión”: Botón que permite cerrar la sesión de un usuario.
* Tabla “Usuarios Registrados”: Tabla donde se visualizan los usuarios que se encuentran registrados.
* Tabla: Muestra la lista de todos los usuarios que se han registrado en el sistema.
* Botón “Editar”: Permite modificar o actualizar el campo de algún usuario.
* Botón “inhabilitar”: Solo permite que el usuario ya no sea visible y no esté disponible para hacer escaneos, de tal manera en la que ya no podrá iniciar sesión hasta que se active nuevamente por parte del administrador.



Nota: En esta interfaz se pueden observar los registros de los usuarios, así como también registrar nuevos y hacer una búsqueda de estos mismos.

### Frontend para la ventana principal de inventarios del supervisor.

Figura 23

*La interfaz principal del supervisor.*

### Componentes de la ventana

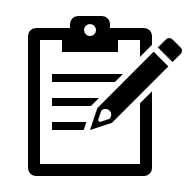
* Botón “Escanear”: Accede al módulo de escaneo.
* Botón “Inventarios”: Accede al módulo de registrar productos de los inventarios.
* Botón “Reportes”: Permite generar reportes de los inventarios.



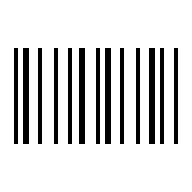
**¡Bienvenido, \_\_\_\_\_\_\_\_\_!**

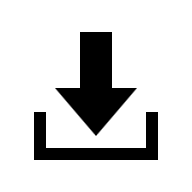
**Cerrar Sesión**

**¿Que deseas hacer hoy?**



Inventarios

Escanear



Reportes

*Nota:* Se visualiza la interfaz principal que tendrá el sistema a la hora de iniciar sesión como Supervisor.

### Frontend para la ventana principal del supervisor de calidad (Este usuario con rol de supervisor es el único y exclusivo que puede registrar usuarios de su área con rol de operador).

Figura 24

*La interfaz principal del supervisor.*

### Componentes de la ventana

* Botón “Escanear”: Accede al módulo de escaneo.
* Botón “Inventarios”: Accede al módulo de registrar productos de los inventarios.
* Botón “Reportes”: Permite generar reportes de los inventarios.
* Botón “Usuarios”: Accede al panel del registro de usuarios y el historial.

*Nota:* Se visualiza la interfaz principal que tendrá el sistema a la hora de iniciar sesión como Supervisor de Calidad.

### Frontend para el panel escanear del supervisor

Figura 25

*Esta interfaz muestra las interacciones que tiene el panel principal del Supervisor.*

### Componentes de la ventana

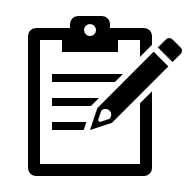
* Botón “Escanear”: Acceder a módulo de escaneo.
* Botón “Inventarios”: Acceso a los inventarios con los que cuenta el administrador de esa área.
* Botón “Reportes”: Acceso a panel de selección del reporte que desea descargar ya sea de sus inventarios o de sus escaneos.
* Botón “Cerrar sesión”: Cierra sesión.

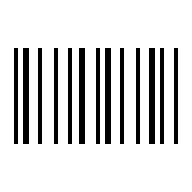


**¿Qué deseas realizar?**

**¡Bienvenido, \_\_\_\_\_\_\_\_\_!**

**Cerrar Sesión**

Añadir

Escaneo

*Nota:* En esta interfaz se muestra los botones que permiten el escanear y verificar los productos registrados.

### Frontend para seleccionar el producto que se escaneara e indicar el número de Lote del supervisor

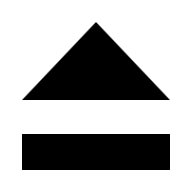
Figura 26

*Esta interfaz muestra el panel para la selección del producto que se escaneara del supervisor.*

### Componentes de la ventana

* ComboBox “Producto a escanear”: Selección de producto que se escaneara.
* Campo de texto “No. Lote”: Ingreso del número del lote que se escaneara.
* Botón “Cancelar”: Botón para cancelar la operación y volver a la sección anterior.
* Botón “Continuar”: Botón para continuar el proceso.

Forma, Rectángulo

El contenido generado por IA puede ser incorrecto.

**¿Qué producto escanearas hoy?**

Palomitas XXXX 16G

Palomitas XXXX 16G

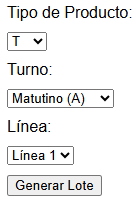
Producto a escanear

No. Lote

Continuar

Cancelar

Quien realiza la operación



*Nota:* En esta interfaz se selecciona el producto que se escaneara para hacer la comparación y se indica el número de lote.

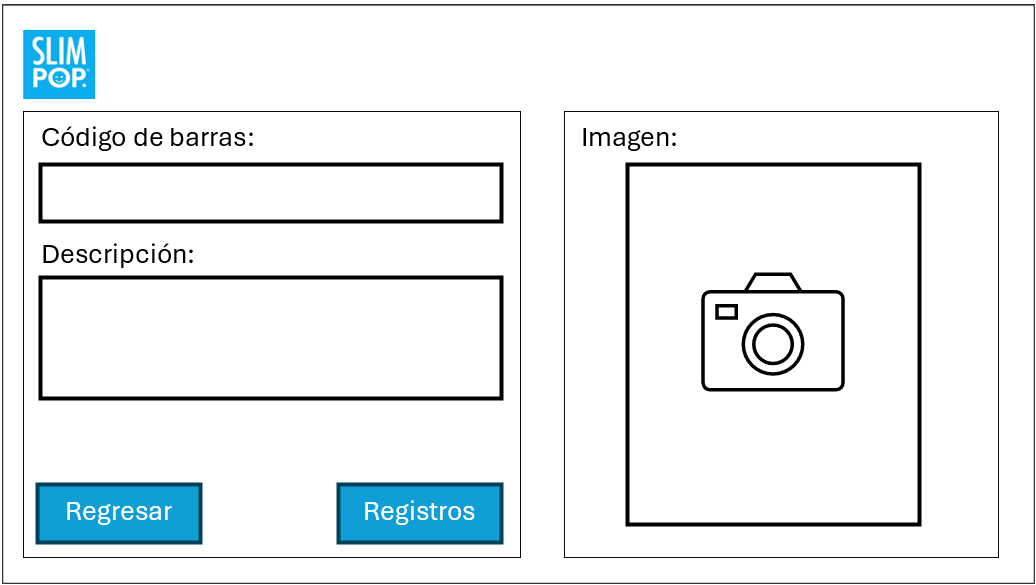
### Frontend para el botón escanear del supervisor

Figura 27

*Esta interfaz gráfica muestra la ventana para realizar los escaneos.*

### Componentes de la ventana

* Campo de texto “Código de barras”:
* Campo de texto “Descripción”:
* Botón “Regresar”: Botón para regresar al panel anterior.
* Botón “Registros”: Botón para acceder a los registros de escaneos.



*Nota:* En esta interfaz se realiza el escaneo en base a el código de barras del producto y el que se seleccionó previamente.

### Frontend para el botón de registros escaneados del supervisor

Figura 28

*Esta interfaz muestra el listado de los productos que se han registrado además de la opción para generar un reporte.*

### Componentes de la ventana

* Tabla de “Registro de escaneos”: Tabla donde se almacenan todos los escaneos conforme a ID, Lote, Escaneó Fecha, Hora, Código de barras, Descripción y Estado.
* Botón “Generar reporte”: Acceso a el apartado de preferencias del reporte.
* Botón “Regresar”: Retorno al panel anterior.



*Nota:* En esta interfaz se muestra el listado de los productos que se han escaneado y de los que se generara el reporte.

### Frontend para la generación de reportes de los productos del botón escanear del supervisor

Figura 29

Esta interfaz muestra las opciones y preferencias para la generación del reporte.

### Componentes de la ventana.

* Campo de texto “Fecha inicio”: Fecha inicial para conteo de fechas.
* Campo de texto “Fecha final”: Fecha final para conteo de fechas.
* CheckBox “Con Lote”: Para incluir los lotes en el reporte.
* CheckBox “Con Horas”: Para incluir la hora en el reporte.
* Botón “Generar”: Para la generación del reporte.

**Imagen que contiene Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.**

**Preferencias de reporte**

**Orden**

Fecha: Desde Hasta

Con Lote Con Hora

Cancelar

Generar

Operador

Producto

Formato:

PDF

XLSX

*Nota:* En esta interfaz se puede elegir la fecha de preferencia para poder generar el reporte.

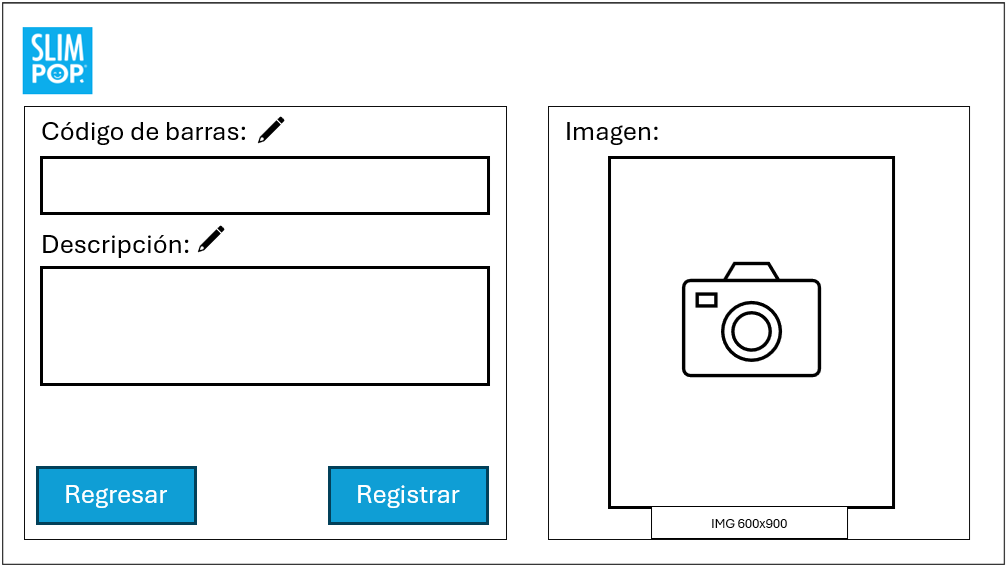
### Frontend para el botón añadir del Supervisor

Figura 30

*La interfaz gráfica muestra el botón donde se deben de registrar los productos.*

### Componentes de la ventana

* Campo de texto “Código de barras”: Permite ingresar el código de barras del producto, con una longitud de 13.
* Campo de texto “Descripción”: Permite que se ingrese la descripción del producto para así poder identificarlo en la base de datos.
* Botón “Regresar”: Botón para regresar al panel anterior.
* Botón “Registrar”: Botón para registrar el producto.
* Botón “Reportes”: Botón para realizar los reportes de los productos.



**Reportes**

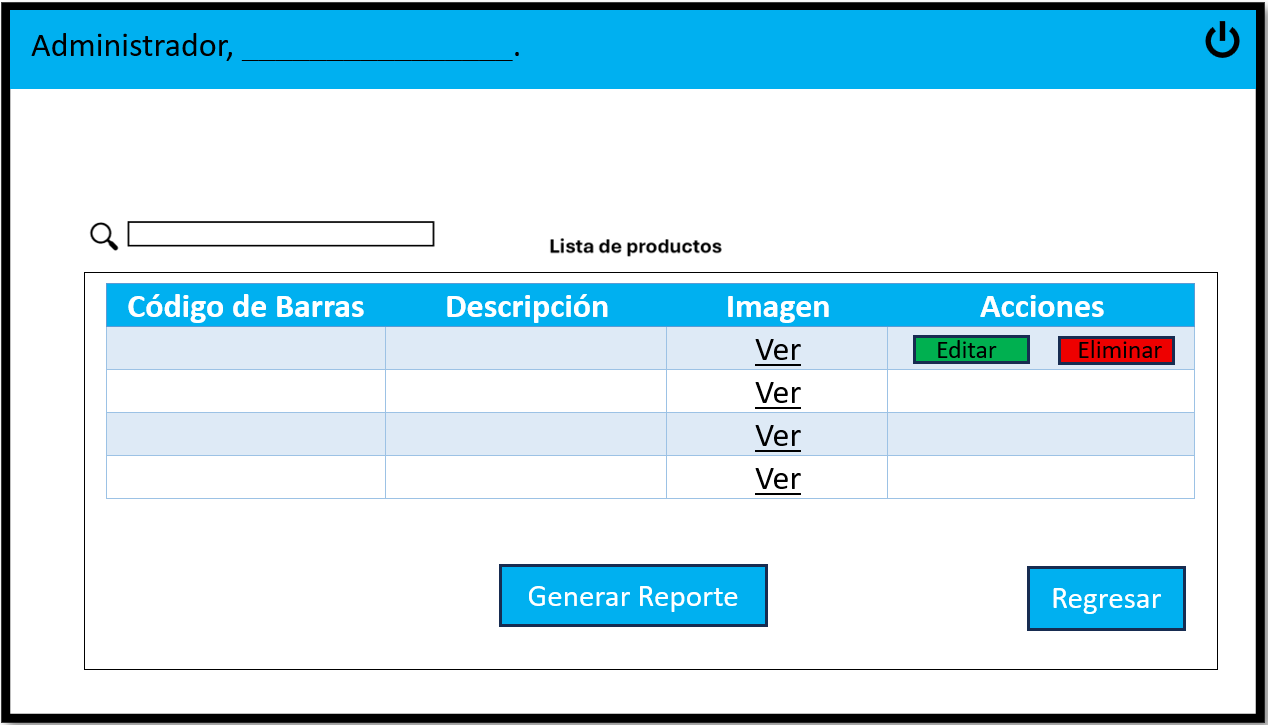
*Nota:* La interfaz muestra los campos para poder registrar un producto que se escaneara próximamente.

### Frontend para el botón de reportes de los productos añadidos del supervisor

Figura 31

*Esta interfaz muestra el listado de los productos que se han registrado además de la opción para generar un reporte.*

### Componentes de la ventana

* Tabla de “Lista de productos”: Tabla donde se almacenan todos los escaneos conforme a ID, Lote, Escaneó Fecha, Hora, Código de barras, Descripción y Estado.
* Botón “Editar”: Permite actualizar el campo seleccionado.
* Botón “Eliminar”: Botón que permite eliminar algún producto seleccionado.
* Botón “Generar reporte”: Acceso al apartado de preferencias del reporte.
* Botón “Regresar”: Retorno al panel anterior.

*Nota:* En esta interfaz se muestra el listado de los productos que se han escaneado y de los que se generara el reporte.

### Frontend para generar los reportes de los productos añadidos del supervisor

Figura 32

*Esta interfaz muestra las opciones y preferencias para la generación del reporte.*

### Componentes de la ventana.

* Campo de texto “Fecha inicio”: Fecha inicial para conteo de fechas.
* Campo de texto “Fecha final”: Fecha final para conteo de fechas.
* CheckBox “Con Lote”: Para incluir los lotes en el reporte.
* CheckBox “Con Horas”: Para incluir la hora en el reporte.
* Botón “Generar”: Para la generación del reporte.

**Imagen que contiene Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.**

**Preferencias de reporte**

**Orden**

Fecha: Desde Hasta

Cancelar

Generar

Producto



Formato:

PDF

XLSX

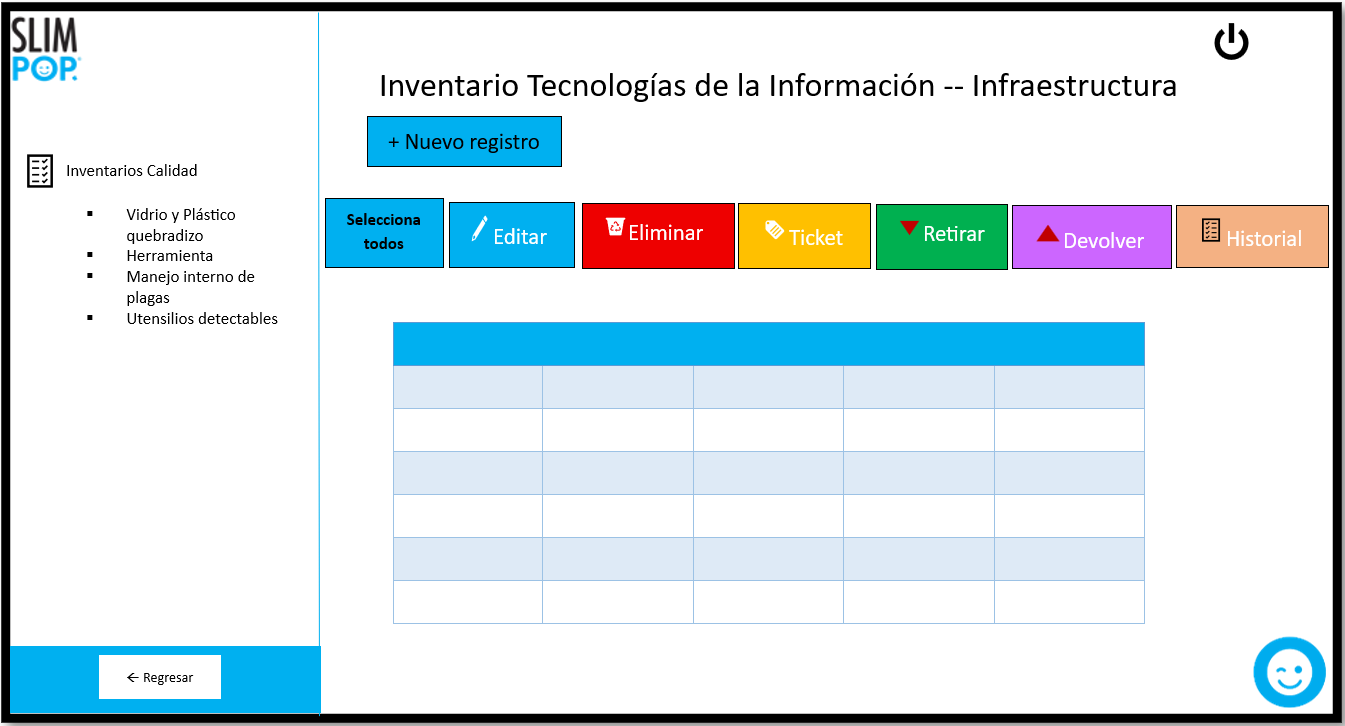
*Nota:* En esta interfaz se puede elegir la fecha de preferencia para poder generar el reporte.

### Frontend para la ventana principal de los inventarios de cada área.

Figura 33

*En esta interfaz se muestra la ventana del inventario correspondiente a su área.*

### Componentes de la ventana

* Menú: En este menú se estarán mostrando las categorías que tiene cada inventario según su área a la que corresponda.
* Botón “Regresar”: Permite regresar a la página anterior.
* Botón “Nuevo registro”: Permite ingresar un nuevo registro a la base de datos.
* Botón “Seleccionar todos”: En este botón se permite seleccionar todos los registros existentes en la tabla.
* Botón “Editar”: Una vez que se selecciona una sola fila se abre el panel de editar el campo.
* Botón: “Eliminar” Una vez seleccionado se puede eliminar el registro de la tabla.
* Botón “Ticket”: Funciona para generar el ticket de cada producto imprimiendo el número del producto.
* Botón “Retirar”: Este botón funciona para hacer algún retiro de producto tan solo ingresando el código de retiro y la cantidad.
* Botón “Devolver”: Mediante este botón se puede el producto o material que anteriormente se haya retirado tan solo ingresando el código de retiro y la cantidad a devolver.
* Tabla: Permite mostrar todos los registros que se hayan realizado.
* Advertencia: Si falta algún campo falta por rellenarse al dar clic en “Guardar” se emitirá una alerta visual con la leyenda “Rellene todos los campos para continuar”.

*Nota:* En esta interfaz se muestra lo correspondiente al registro de materiales de inventarios correspondiente a su área.

### Frontend para el operador

Figura 34

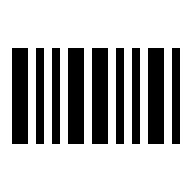
*En esta interfaz se muestra la ventana que tendrá el rol de operador que es exclusivamente para escanear los productos.*

### Componentes de la ventana:

* Botón” Escaneo”: Botón que permite ingresar a la siguiente ventana que corresponde a la de escanear los productos.
* Botón “Cerrar sesión”: Este botón permite cerrar la sesión del usuario operador y lo dirige nuevamente al login.

 ¡Bienvenido!

Cerrar Sesión



Escaneo

**¿Qué deseas realizar?**

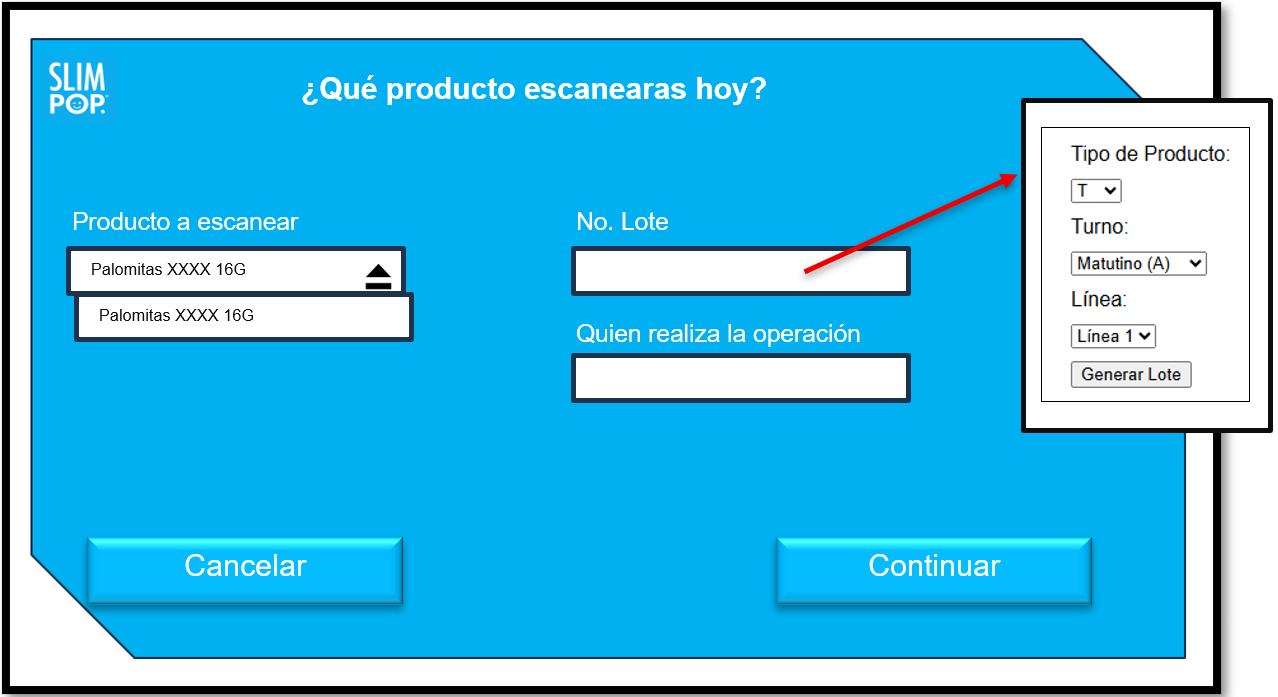
*Nota:* En esta interfaz muestra el botón que dirige a la ventana de escanear el producto.

### Frontend para seleccionar el producto que se desea escanear del operador.

Figura 35

*Esta interfaz se muestra los campos correspondientes para poder seleccionar un producto que se desee escanear.*

### Componentes de la ventana:

* ComboBox “Producto a escanear”: Selección de producto que se escaneara.
* Campo de texto “No. Lote”: Ingreso del número del lote que se escaneara.
* Botón “Cancelar”: Botón para cancelar la operación y volver a la sección anterior.
* **Botón “Continuar”: Botón para continuar el proceso.

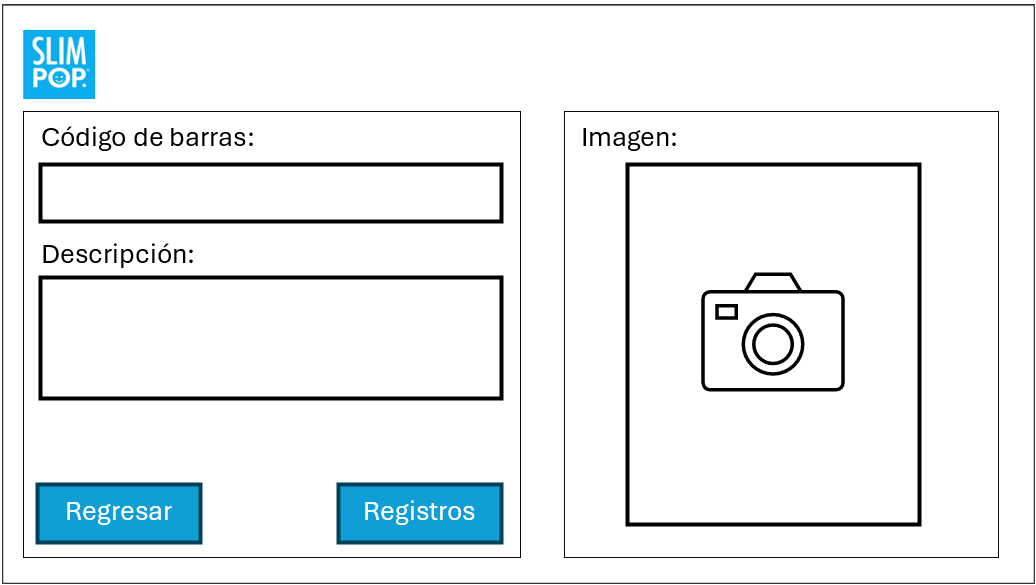
*Nota: E*sta interfaz permite Seleccionar el producto y definir el lote.

### Frontend para escanear el producto seleccionado del operador

Figura 36

*La interfaz gráfica del botón escanear donde se muestras los datos del producto escaneado.*

### Componentes de la ventana:

* Campo de texto “Código de barras”: Este campo permite mostrar el código de barras que se está escaneando.
* Campo de texto “Descripción”: Este campo se llena automáticamente al ingresar el código de barras.
* Botón “Regresar”: Botón para regresar al panel anterior del producto a escanear.
*  Botón “Registros”: Botón para acceder a los registros de escaneos

*Nota:* En esta interfaz se mostrará los datos ya escaneados según su correspondencia.

### Frontend para el registro de los escaneos realizados por parte del operador.

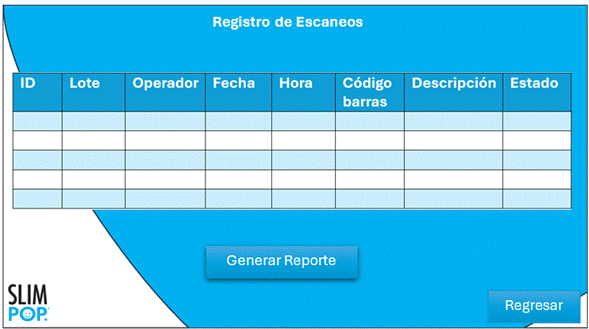
Figura 37

*La interfaz gráfica del botón que muestra la anterior ventana de registros.*

### Componentes de la ventana:

✔      Tabla de “Registro de escaneos”: Tabla donde se almacenan todos los escaneos conforme a ID, Lote, Escaneo, Fecha, Hora, Código de barras, Descripción y Estado.

✔      Botón “Generar reporte”: Acceso a el apartado de preferencias del reporte.

✔      Botón “Regresar”: Retorno al panel anterior.

*Nota:* La interfaz que va a permitir generar los reportes de los productos que ya están escaneados.

### Frontend para elegir las preferencias de impresión del reporte.

Figura 38

*Esta interfaz muestra las opciones y preferencias para la generación del reporte.*

### Componentes de la ventana.

* Campo de texto “Fecha inicio”: Fecha inicial para conteo de fechas.
* Campo de texto “Fecha final”: Fecha final para conteo de fechas.
* CheckBox “Con Lote”: Para incluir los lotes en el reporte.
* CheckBox “Con Horas”: Para incluir la hora en el reporte.
* Botón “Generar”: Para la generación del reporte.

**Imagen que contiene Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.**

**Preferencias de reporte**

**Orden**

Fecha: Desde Hasta

Con Lote Con Hora

Cancelar

Generar

Operador

Producto

Formato:

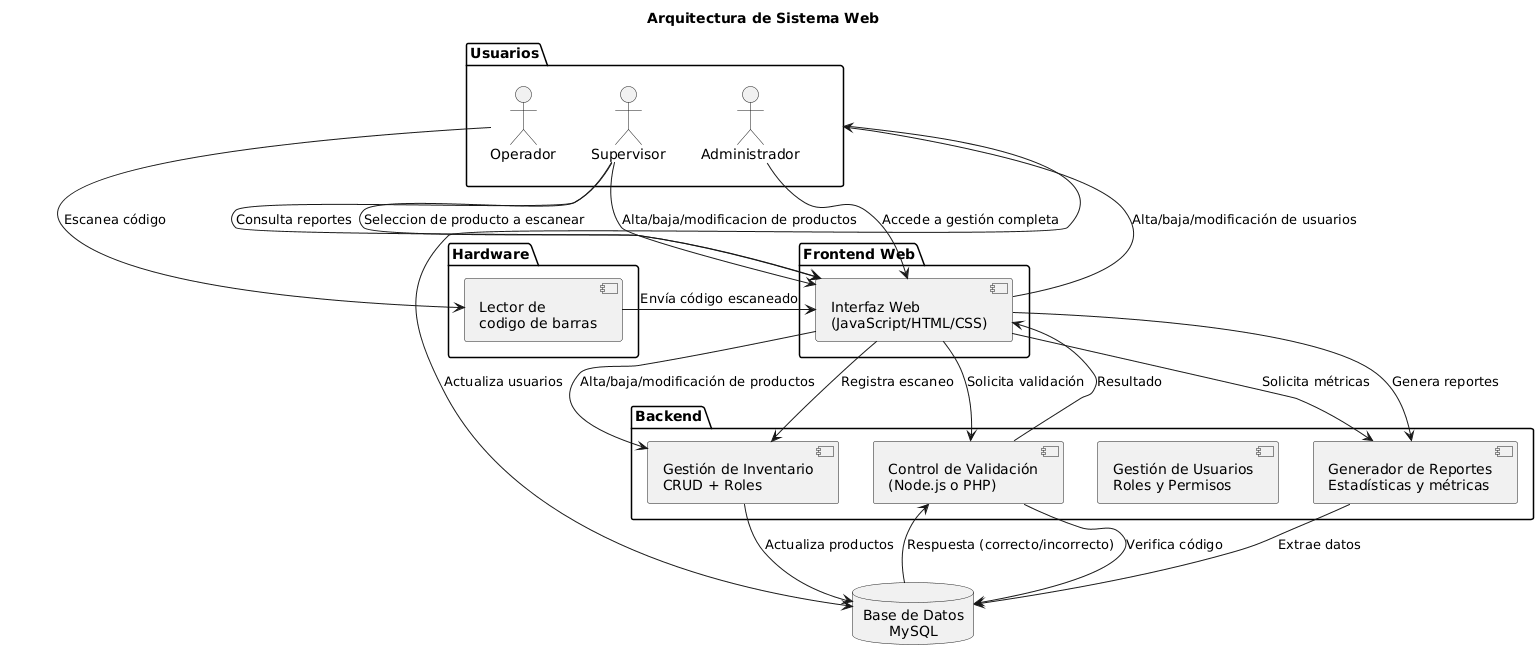
PDF

XLSX

*Nota:* En esta interfaz se puede elegir la fecha de preferencia para poder generar el reporte.

### Arquitectura del sistema

Figura 39

*La imagen representa la arquitectura del cliente*

*Nota:* En la imagen se puede observar el funcionamiento del sistema la cual tiene un orden en base a los roles y áreas de los colaboradores.

## Fase 3

## Implementación

### Elegir el SGBD a utilizar

Para el desarrollo del sistema se utilizará XAMPP, siendo una distribución de código abierto que permite instalar fácilmente un servidor web, este paquete incluye componentes esenciales como Apache, MySQL y PHP, lo que lo convierte en una herramienta gratuita ideal para el desarrollo y prueba de aplicaciones web en sistemas operativos como Windows, Linux y macOS, además, incorpora phpMyAdmin, una interfaz gráfica que facilita la gestión de bases de datos de manera sencilla y eficiente.

Además, phpMyAdmin cuenta con un amplio soporte lo que garantiza abundante documentación y soluciones a problemas comunes. Su integración nativa con MySQL y MariaDB le permite aprovechar al máximo las funciones de estos motores, permitiendo realizar tareas como la creación de tablas, la edición de registros, la asignación de permisos o la realización de copias de seguridad de manera ágil y organizada.

Por lo tanto, usar phpMyAdmin como sistema gestor de base de datos para el sistema web de inventarios y validación de códigos de barrasresulta muy conveniente ya que ofrece una interfaz gráfica sencilla e intuitiva, lo que facilita la administración de bases de datos gracias a su funcionamiento desde el navegador, y no requiere instalaciones adicionales en cada equipo permitiendo gestionar la base de datos desde cualquier lugar, siempre y cuando se cuente con las credenciales de acceso.

### Codificación de la base de datos

La base de datos utiliza la codificación utf8mb4 con collation utf8\_spanish\_ci, lo que permite almacenar correctamente caracteres especiales del idioma español y símbolos adicionales, esta configuración asegura la compatibilidad con aplicaciones web y móviles, evitando errores de interpretación en la información almacenada.

## Tabla usuarios

### Login (Acceso y control de área y rol)

La tabla usuarios está compuesta por campos que van a permitir el almacenamiento de los datos personales de los usuarios, las filas que contiene esta tabla son:

**no\_colaborador:** Toma como valor entero (INT) este siendo una clave primaria, indicando que va a tomar como valor único de la fila y del registro. En este se almacenará el número del colaborador.

**nombre\_colaborador:** Toma como valor variable el VARCHAR(45) ya que almacena series de caracteres. Este campo almacena el nombre del usuario y viene siendo una clave foránea ya que más adelante en las tablas se manda a traer este campo.

**rol\_colaborador:** Toma como valor VARCHAR(15) para almacenar caracteres. En este campo se estarán almacenando el rol de los usuarios como administradores, supervisores y operadores.

**Área:** Toma como valor VARCHAR(50) para almacenar letras y números. Este campo se estará almacenando las áreas que cuenta la empresa como Tecnologías de la información (TI), Talento Humano (TH), Mantenimiento, Calidad, Almacén.

**Contraseña:** Toma como valor VARCHAR(15) este para almacenar letras, números, caracteres y símbolos especiales. Cabe mencionar que este campo en la base de datos va a contar con un hash que lo hará más seguro.

**Fecha:** Toma como valor DATE donde permite extraer un valor de fecha de una entrada de tiempo. Almacena la fecha en que se realiza el registro.

**Estatus:** Toma como valor VARCHAR(20) para definir si el usuario esta activo, o inhabilitado.

## Código ejecutado en phpMyAdmin

CREATE TABLE `usuarios` ( `no\_colaborador` int(11) NOT NULL, `nombre\_colaborador` varchar(45) NOT NULL, `rol\_colaborador` varchar(15) DEFAULT NULL, `area` varchar(50) DEFAULT NULL, `contrasena` varchar(15) DEFAULT NULL, `fecha` date DEFAULT NULL, `estatus` varchar(20) NOT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8mb3\_general\_ci;

## Tablas Inventarios

En las tablas de los inventarios se almacenan

## Tabla Categorías

La tabla categorías permite almacenar los datos de un inventario que cuente con categorías en sus productos o materiales. Contiene 5 campos las cuales son:

**ID:** Funciona como identificador único de la tabla (PRIMARY KEY) y toma como valor entero (INT).

**Inventario\_id:** Funciona como clave secundaria para la identificación del id de la tabla de inventarios tomando como valor entero INT.

**Nombre:** Se le asigna un valor VARCHAR(120). Este campo va a permitir almacenar el nombre de las categorías.

**Campos:** Se toma como valor LONGTEXT se utilizó para almacenar grandes cantidades de datos de cadena. Mediante este campo se almacenarán las filas de las tablas.

**Fecha\_creación:** Este campo toma como valor el TIMESTAMP para convertir los valores de la zona horaria actual a UTC para su almacenamiento y viceversa para su recuperación. Almacenara la fecha en que se realice un registro.

## Código ejecutado en phpMyAdmin

CREATE TABLE `categorias` (`id` int(11) NOT NULL, `inventario\_id` int(11) NOT NULL, `nombre` varchar(120) NOT NULL, `campos` longtext CHARACTER SET utf8mb4 COLLATE utf8mb4\_bin NOT NULL DEFAULT '[]' CHECK (json\_valid(`campos`)), `fecha\_creacion` timestamp NOT NULL DEFAULT current\_timestamp()) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

## Tabla Campos

En la tabla de los campos se registran los datos al generar un nuevo inventario dando un campo identificador que será útil para la impresión de etiquetas con código de barras e identificando los artículos físicos, y de igual forma se asignan los campos que tendrán las tablas para guardar los campos registros.

**ID:** Se le asigna un valor entero (INT) con clave primaria (PRIMARY KEY), este siendo AUTO\_INCREMENT.

**Id\_categoria:** Se toma como valor entero (INT) con clave secundaria.

**Nombre:** Este campo se le asigna un valor variable VARCHAR(150). Almacenara todos los campos que se vallan registrando.

**Tipo:** En este campo se le da un valor variable VARCHAR(50). Este campo almacena los datos del archivo XML definiendo un conjunto de reglas para codificar documentos en formato legible.

**Orden:** Se le da el valor de entero (INT).

## Código ejecutado en phpMyAdmin

CREATE TABLE `campos` ( `id` int(11) NOT NULL, `id\_categoria` int(11) NOT NULL, `nombre` varchar(150) NOT NULL, `tipo` varchar(50) DEFAULT 'text', `orden` int(11) DEFAULT 0, `requerido` tinyint(1) DEFAULT 0, `meta` longtext CHARACTER SET utf8mb4 COLLATE utf8mb4\_bin DEFAULT NULL CHECK (json\_valid(`meta`)), `fecha\_creacion` timestamp NOT NULL DEFAULT current\_timestamp()) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

## Tabla Registros

**ID:** Toma como valor entero INT siendo este AUTO\_INCREMENT.

**Catgoria\_id:** Este es una clave secundaria FOREING KEY y su valor es de INT.

**Datos:** Este es un tipo de dato LONGTEXT la cual va permitir tener una longitud grande de datos.

**Fecha\_creacion:** Se le asigna un tipo de dato TIMESTAMP donde va a almacenar la fecha en la que se creó el registro.

**Fecha\_modificacion:** Su valor es de TIMESTAMP para poder almacenar la fecha en la que se modigique el registro.

**Texto\_busqueda:** Este es un valor TEXT.

## Código ejecutado en phpMyAdmin

CREATE TABLE `registros` ( `id` int(11) NOT NULL,  `categoria\_id` int(11) NOT NULL,  `datos` longtext CHARACTER SET utf8mb4 COLLATE utf8mb4\_bin NOT NULL CHECK (json\_valid(`datos`)),  `fecha\_creacion` timestamp NOT NULL DEFAULT current\_timestamp(),  `fecha\_modificacion` timestamp NOT NULL DEFAULT current\_timestamp() ON UPDATE current\_timestamp(),  `texto\_busqueda` text DEFAULT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

## Tabla Productos\_Escanear

**Codigo\_barras:** Este campo toma como valor VARCHAR(20) y es la clave primaria de la tabla. Almacenara el código de barras del producto que se escanee.

**Descripción:** Este es un valor VARCHAR(255)

**Imagen:** En este campo toma como valor VARCHAR(255)

**Fecha:** Se le da el valor de DATE para almacenar la fecha.

## Código ejecutado en phpMyAdmin

CREATE TABLE `productos\_escanear` (`codigo\_barras` varchar(20) NOT NULL, `descripcion` varchar(255) NOT NULL, `imagen` varchar(255) DEFAULT NULL, `fecha` date DEFAULT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8mb3\_general\_ci;

## Tabla Registro\_Escaneos

**Id\_escaneo:** Este es el código único que se le asigna al producto que se escaneo y es una PRIMARY KEY tomando como valor entero INT.

**Código\_barras\_producto:** almacena todos los datos escaneados y toma como valor VARCHAR(20) donde este viene siendo una clave secundaria.

**Descripción\_producto:** En este campo se almacena el producto si fue encontrado a la hora de ser escaneado. Tomando como valor VARCHAR(255).

**Lote\_escaneado:** En este campo se almacena el lote generado y tomando como valor de carácter VARCHAR(50).

**Fecha\_escaneo:** En este campo almacena la fecha en la que se escaneo el registro almacenado y toma como tipo de dato DATE.

**Hora\_escaneo:** Registra la hora en la que se realizó el esceno y es un tipo de dato TIME.

**Estado\_escaneo:** Almacena el dato de que si fue encontrado con éxito o si fue fallido por lo que este es un carácter y toma como tipo de dato VARCHAR(45).

**Nombre\_colaborador\_usuarios:** Solo almacena el nombre del usuario y cual realizo la operación tomando como valor VARCHAR (100)**.**

## Código ejecutado en phpMyAdmin

CREATE TABLE `registros\_escaneos` (`id\_escaneado` int(11) NOT NULL, `codigo\_barras\_producto` varchar(20) DEFAULT NULL, `descripcion\_producto` varchar(255) DEFAULT NULL, `lote\_escaneado` varchar(50) DEFAULT NULL, `fecha\_escaneo` date DEFAULT NULL, `hora\_escaneo` time DEFAULT NULL, `estado\_escaneo` varchar(45) DEFAULT NULL, `nombre\_colaborador\_usuarios` varchar(100) DEFAULT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8mb3 COLLATE=utf8mb3\_general\_ci;

## Tabla Historial

Id: INT

Fecha\_operación: DATATIME

Operación: ENUM()

### Configurar el lector SICAR Checko2B para enviar datos al sistema web.

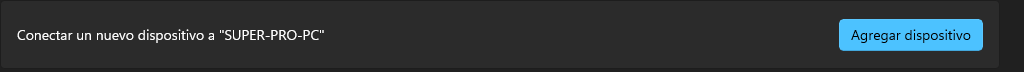
El lector de códigos de barras Checko2B es un potente lector con capacidades inalámbricas para que puedas usarlo con toda la comodidad en dispositivos móviles como tablets y celulares. Además, también puede ser usado con equipos de escritorio con conectividad bluetooth o por cable. (Aréchiga, 2024)

Para que el lector de código de barras SICAR Checko2B se conecte correctamente con el sistema web se necesita hacer una conexión vía Bluetooth o mediante cable. En este caso se va a realizar mediante bluetooth y a continuación se muestran los pasos a seguir.

Como primer paso se ingresa a la configuración de la computadora de escritorio o laptop se dirige uno al apartado de Bluetooth y dispositivos para poder agregar uno nuevo.

Figura 40

*Conectar un nuevo dispositivo bluetooth.*

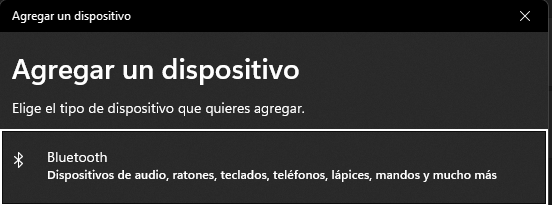


*Nota:* En la imagen se muestra el apartado donde se debe agregar un nuevo dispositivo.

Una vez dentro se le da clic a la opción de Bluetooth.

Figura 41

*Agregar el dispositivo Bluetooth.*

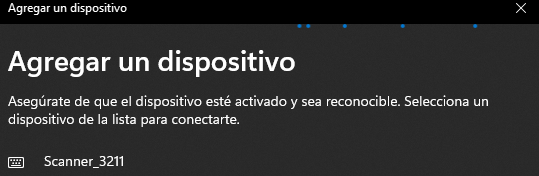


*Nota:* En la imagen se muestra la opción que debemos seguir para conectar el escáner a Bluetooth.

Posteriormente se aparece en las solicitudes el nombre del lector de código de barras SICAR Ckecko2B la cual es “Scanner\_3211”, se le da en agregar para que quede conectado.

Figura 42

*Se agrega el dispositivo a nombre de Scanner\_3211.*



*Nota:* En la imagen se puede observar el nombre del dispositivo del lector de código de barras.

Finalmente se puede observar que efectivamente ya se vinculó a la computadora y está listo para poder ser utilizado.

Figura 43

Conexión correcta del lector de códigos de barras SICAR Ckecko2B a la computadora de escritorio.



*Nota:* En la imagen se muestra la conexión correcta del lector de código de barras SICAR Checko2B en la computadora.

Figura 44

*Ilustración del Lector de código de barras SICAR Checko2B*



*Nota:* Se muestra el lector de código de barras.

### Programar el backend para recibir, validar y almacenar los datos.

## Conexión de la base de datos

La conexión de la base de datos es un archivo .php la cual funciona de tal manera en la que permite la conexión del servidor con phpMyAdmin.

Principalmente se colocan variables donde se solicita el nombre del servidor, el usuario, la contraseña y el nombre de la base de datos donde se va a estará extrayendo la información. A continuación, se muestra el código:

Figura 45

*Se muestra el código que permite la conexión con la base de datos.*

*Nota:* En la imagen se puede observar el código que permite hacer la conexión con la base de datos, solicitando el nombre del servidor, contraseña y nombre de la base de datos posteriormente se verifica la conexión.

## Inicio de Sesión (Index.html)

El índex es la parte principal del programa donde mediante credenciales se validan los usuarios con 3 roles existentes, asignándoles las acciones y permisos que le corresponden a cada uno.

Al ingresa el nombre de usuario y contraseña el sistema los valida los roles y envía los datos mediante el siguiente código:

**fetch("/validar\_usuario.php", {**

**method: "POST",**

**headers: { "Content-Type": "application/json" },**

El validar\_usuario.php es otro formulario que se encarga de validar la solicitud HTTP y sigue un proceso donde primero comienza recibiéndolos datos JSON extrayéndolos para la verificación del usuario y contraseña haciendo una conexión directamente con la base de datos, la cual este es el código que se utiliza para hacer esa conexión:

**$servername = "localhost";**

**$username = "Omar";**

**$password = "Palomitas32$";**

**$dbname = "popcode";**

Posteriormente verifica que el usuario este con estatus activado para permitir darle acceso al sistema, en caso contrario que este desactivado se debe solicitar al administrador ser activado nuevamente, para esa parte de código se muestra más adelante a detalle.

Figura 46

*En la ilustración se muestra el código completo del validad\_usuario.php.*

*Nota:* En la imagen se muestra las lenas de código donde se validan los datos del usuario como su contraseña y nombre.

Posteriormente retomando el formulario index.html, este código lo que hace es recibir y validar los datos lanzando una respuesta del servidor en un objeto JSON usable por JavaScript.

**.then(res => res.json())**

Finalmente procesa la respuesta del servidor.

**body: JSON.stringify({ usuario, contrasena })**

**})**

Una vez obteniendo respuesta del servidor da acceso al sistema según corresponda a su rol, para eso lo valida mediante el siguiente código:

**.then(res => res.json())**

**.then(data => {**

**if (data.exito) {**

**const rol = data.rol\_colaborador;**

**const area = data.area;**

**modalText.innerText = "Acceso concedido. Redirigiendo...";**

**setTimeout(() => {**

**if (rol === "Administrador") {**

**window.location.href = "/MainAdmin.php";**

**} else if (rol === "Supervisor") {**

**window.location.href = `/MainSuper${area}.php`;**

**} else if (rol === "Operador") {**

**window.location.href = "/MainOp.php";**

**} else {**

**ModlText.innerText = "Rol no reconocido.";**

**} }, 1500);**

En caso de que el usuario no exista en la base de datos manda una alerta que menciona que las credenciales son incorrectas.

**} else { modalText.innerText = data.error || "Credenciales incorrectas.";**

**setTimeout(() => modal.style.display = "none", 2000);**

**}**

**})**

**.catch(err => {**

**modalText.innerText = "Error de conexión.";**

**console.error(err);**

**setTimeout(() => modal.style.display = "none", 2000);**

**});**

**}**

**document.addEventListener("keydown", function(event)**

**{**

**if (event.key === "Enter")**

**{**

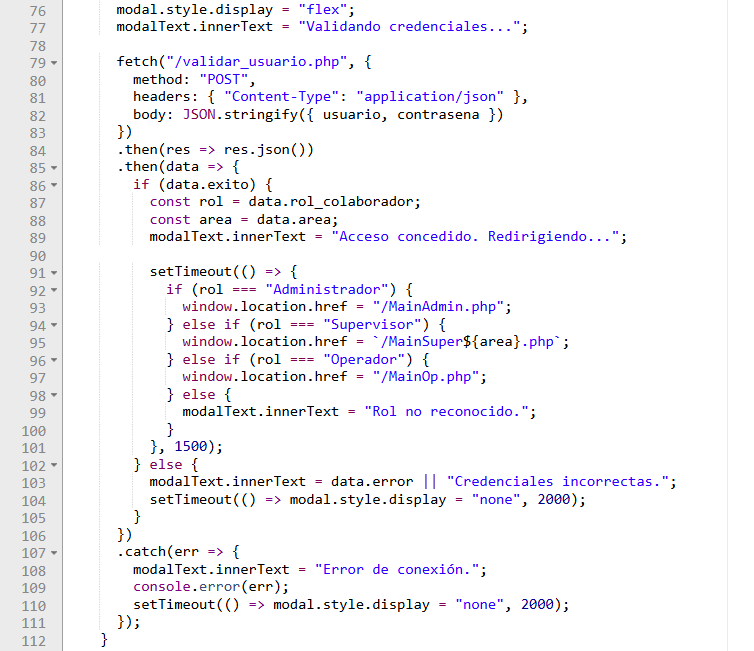
**event.preventDefault();**

**iniciarSesion();**

**}**

**});**

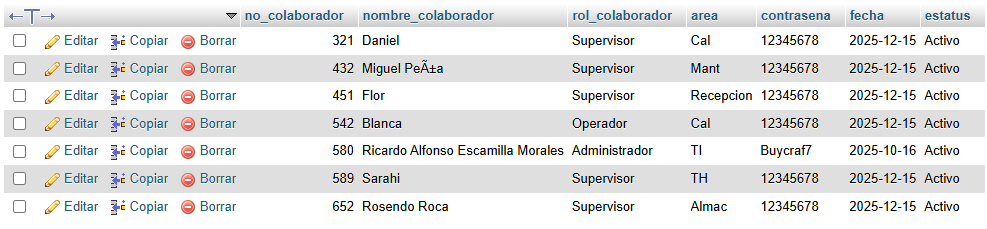
Figura 47

*******El la ilustración se muestra el digo completo de anteriormente explicado.*

*Nota:* Mediante estas líneas de código se observa la lógica que se lleva para que una vez que un usuario ingrese al sistema tengas los permisos de acuerdo al rol en la que este.

En la base de datos se guarda de esta manera los usuarios para validar su existencia. PONER CAPTURA DE LA CONTRASEÑA HASHEADA

Figura 48

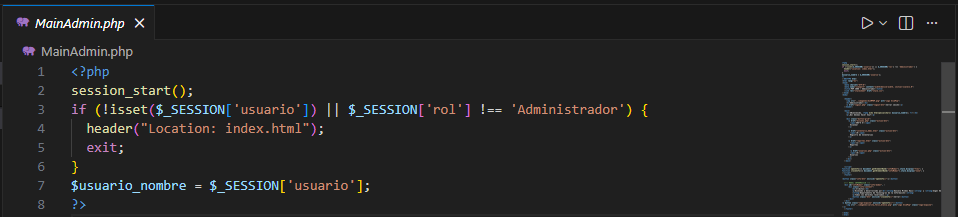
*En la imagen se muestra como se almacenan los datos en la base de datos.*

*Nota:* En la captura se muestran los datos que se tienen por ahora registrados y como se puede observar hay un solo administrador y todas las áreas entran como supervisores, a excepción del operador.

**Backend para recibir los datos, validar y almacenar del administrador.**

Una vez que se validan los datos de usuarios redirecciona de acuerdo al rol que se esté ejecutando que en este caso para el administrador es el único usuario con acceso a todos los permisos obteniendo así el control del sistema y su direccionamiento es MainAdmin.phpque corresponde a la ventana principal que estará visualizando una vez que se valida el usuario.El proceso de dirigir a su panel correspondiente se hace mediante el siguiente código la cual indica que el usuario y contraseñas al ingresar le pertenecen a un administrador.

Figura 48

*En la ilustración se muestra la conexión exitosa.*

*Nota:* Mediante esta línea de código se puede observar que a esta ventana debe de redireccionar los usuarios administradores.

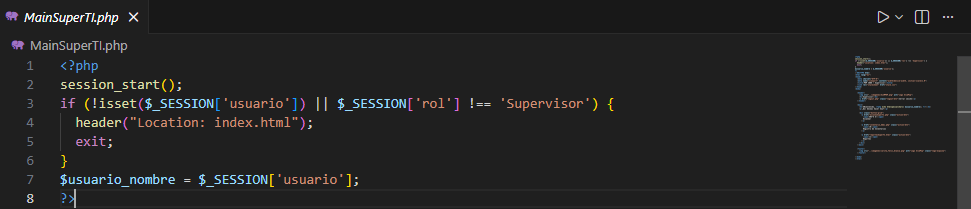
**Backend para recibir los datos, validar y almacenar del supervisor.**

Para a el rol de supervisor se maneja de acuerdo a las áreas de la empresa que se tienen contempladas y registradas las cuales serían las siguientes: Tecnologías de la información, Talento Humano, Mantenimiento, Calidad, Almacén, Recepción. Cada una de estas tienen diferentes permisos sobre los dos módulos que se manejan en el sistema.

**Tecnologías de la información**

El rol de supervisor se redirecciona de acuerdo al área en la que se realizó el registro donde lo almacena en la base de datos para así extraer la información por medio de código y validarlo y de acceso al sistema, donde se toma como direccionamiento el MainSuperTI.php mostrando así su ventana principal correspondiente.

Figura 49

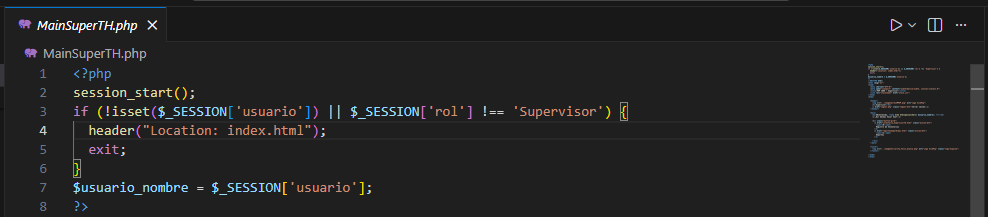
*En la imagen se muestra el acceso autorizado para ingresar al sistema correspondiente al rol de supervisor de TI.*

*Nota:* Mediante esa línea de código se puede observar que el código verifica el nombre de usuario y contraseña sean correctos para dar ingresar.

**Talento Humano**

El rol de supervisor del área de Talento Humano toma otros accesos y permisos a la hora de ingresar al sistema donde mediante le código .php realiza la verificación del rol que juega la persona que desea ingresar que en este caso su direccionamiento es MainSuperTH.php mostrando así su ventana correspondiente.

Figura 50

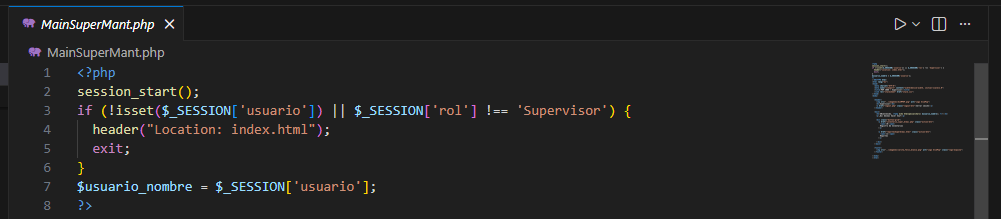
*En la imagen se muestra el código que verifica la validación del usuario, rol y contraseña.*

*Nota:* Mediante la imagen se puede observar el código que verifica el usuario para poder dar acceso a la ventana correspondiente del sistema.

**Mantenimiento**

El rol del supervisor correspondiente al área de Mantenimiento toma otros permisos de acuerdo a como se haya registrado en la base de datos ya que mediante eso se manda a traer esos campos para poder darle acceso al sistema el usuario, una vez que se verifica y comprueba la existencia direccionando al archivo MainSuperMant.php para mostrarle su ventana principal correspondiente.

Figura 51

*En la imagen se muestra el código para dar acceso al sistema web.*

*Nota:* Mediante la siguiente línea de código que se muestra en la imagen se puede observar el proceso que se debe de hacer para hacer la comprobación con la base de datos que el usuario este registrado.

**Calidad**

El rol del supervisor correspondiente al área de Calidad tiene otros permisos para poder acceder a la ventana principal del sistema ya que este le corresponde el direccionamiento de MainSuperCal.php una vez verificando su nombre, contraseña y rol en la base de datos hace la conexión exitosa para dar acceso.

Figura 52

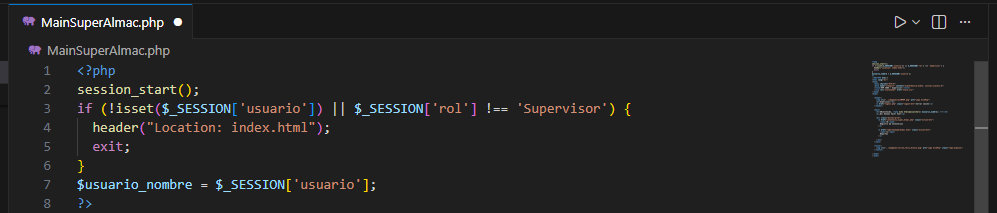
*****En la imagen se muestra el código que verifica la existencia del usuario.*

*Nota:* Mediante el siguiente código se muestra la validación de usuario solicitando el rol y si corresponde al supervisor para poder darle acceso al sistema.

**Almacén**

El rol de supervisor correspondiente al área de Almacén es otra de las áreas que se tienen contempladas para la verificación de usuarios la cual su direccionamiento es MainSuperAlmac.php si el sistema comprueba mediante la base de datos que el usuario si existe como se ingreso manda a la ventana correspondiente.

Figura 53

*En la imagen se muestra el código que permite hacer la comprobación del usuario para dar acceso al sistema.*

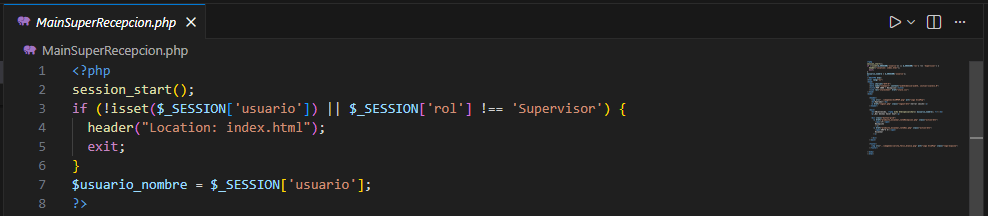
*Nota:* Mediante la imagen se puede mostrar el código que verifica que el nombre del usuario, contraseña y rol sean los correctos y que existan en la base de datos para poder ingresar.

**Backend para recibir los datos, validar y almacenar de recepción.**

**Recepción**

Este es uno de los roles especiales que entran como supervisores porque estos tienen muy limitado el acceso, pero su verificación de usuario siempre va a corresponder a la de un supervisor. Y s direccionamiento es MainSuperRecepción.php.

Figura 54

*****En la imagen se puede observar el código que verifica la existencia de un usuario con rol de recepción.*

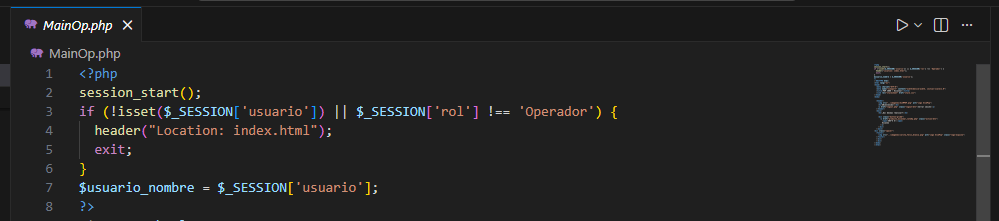
*Nota:* Mediante la siguiente línea de código se puede mostrar que realiza la verificación antes de poder dar acceso y esto es para seguridad de los datos y restringir los permisos.

**Backend para recibir los datos, validar y almacenar del operador.**

**Operador**

El operador es el usuario con los permisos limitados donde solo tiene acceso a una sola ventana que le permite realizar escaneos de productos en tiempo real. Donde el sistema detecta la validación del usuario con el nombre y contraseña para verificar a que rol pertenece y así mandar al panel correspondiente, el operador su direccionamiento es MainOp.php.

Figura 55

*****En la imagen se muestra el código que da acceso a solo operadores.*

*Nota:* Mediante la siguiente imagen se puede observar que existe un usuario con el rol de operador la cual tiene muy limitados los permisos y accesos la cual mediante el sistema detecta que un usuario se registró como operador lo manda a su direccionamiento MainOp.php.

**Backend para recibir los datos, validar y almacenar del módulo escanear del administrador, supervisor y operador.**

El módulo de escáner es el encargado de realizar el proceso de escanear el producto que pasa por producción para hacer la validación del producto, este módulo lo manejan de igual manera los 3 roles que están disponibles que son: administrador, supervisor y operador, para el rol de recepción es de distinta manera por lo que se explica más adelante.

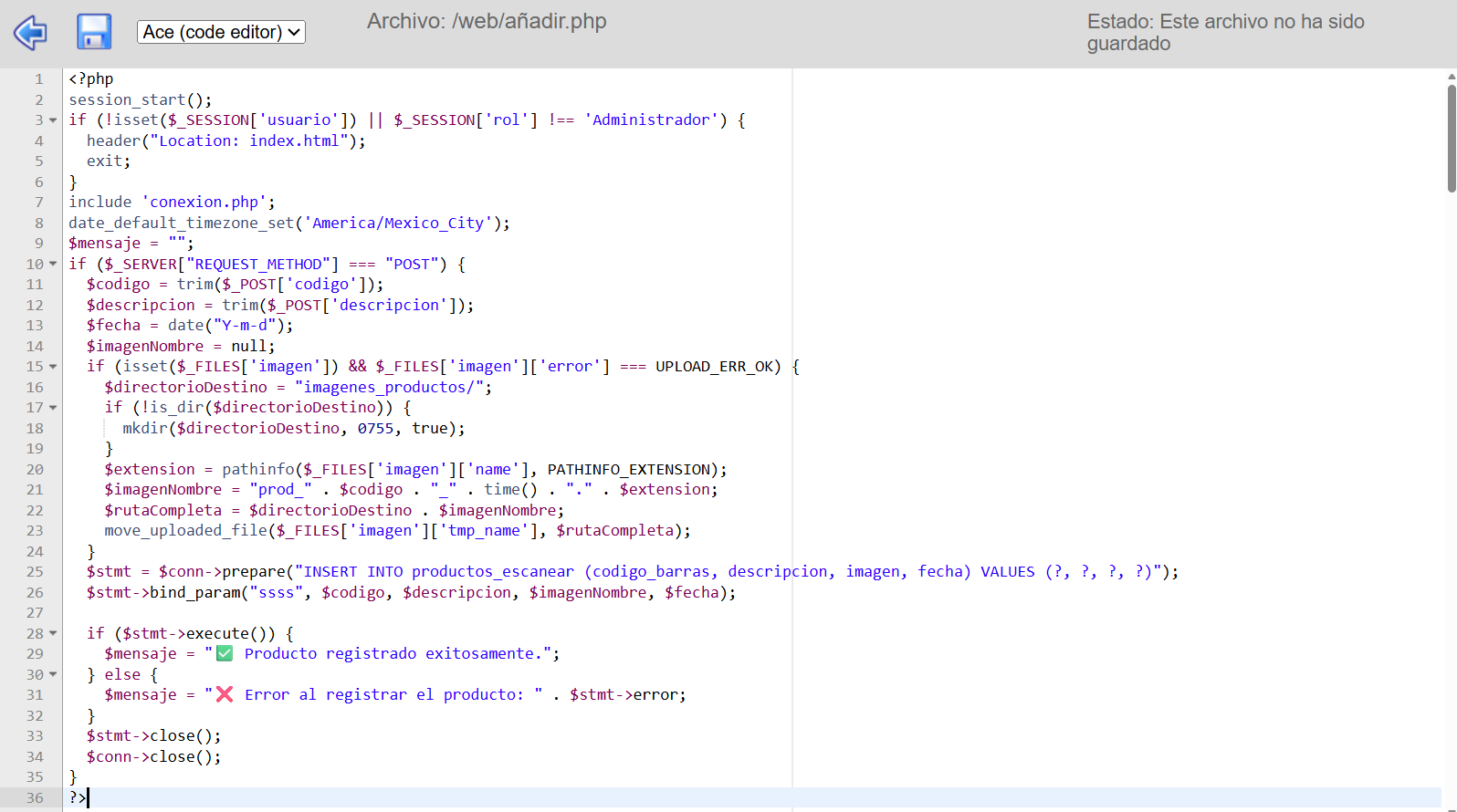
Para eso se necesita agregar un producto correctamente a la base de datos para que acepte la validación una vez que se escanea, se creó un archivo llamado añadir.php donde contiene el código que se ocupara para realizar una conexión a la base de datos, así como también realizar los registros mediante la sentencia de: INSERT INTO productos\_escanear (codigo\_barras, descripcion, imagen, fecha) VALUES (?, ?, ?, ?)"); la cual hace el registro en la base de datos.

Pero antes de realiza la sentencia se necesita hacer una conexión con la base de datos mandando atraer el archivo conexión.php, donde contiene los datos específicos para realizar conexión entre mysql y el sistema.

En primera parte se inicializan las variables mandando a traer los datos de acuerdo a como este en la base de datos, así como también cargar la imagen que se valla a colocar verificando que se haya subido correctamente y en caso de que no encuentre ningún directorio en donde almacenar la imagen crea uno por defecto asignándole un nombre y guardándola así en la base de datos.

Finalmente realiza la sentencia para almacenar los datos registrados en la tabla de la base de datos ejecutando los mensajes de “Productos registrado exitosamente”, en caso de error ejecutan el siguiente error “Error al registrar el producto” y finaliza la conexión.

Figura 56

*El código muestra las líneas donde se extrae información de la base de datos al sistema.*

*Nota:* Mediante la siguiente imagen se muestra las líneas de código de cómo se manda a traer la información de la base datos y como es que esta almacenándola.

En la base de datos se muestra la tabla donde almacena los datos que se añaden, el nombre de la tabla es productos\_escanear.

Figura 58

*Tabla de la base de datos donde almacena los registros.*

*Nota:* Se muestran los registros que se agregan a la tabla de la base de datos, cada uno de estos en su campo correspondiente.

Para el código que realiza el registro de todos los productos que se escanean se guardan en la base de datos que toma como nombre la tabla registros\_escaneos donde se manda a traer esta tabla, para eso se necesita hacer una conexión con la BD y así llevar a cabo la consulta de SELECT codigo\_barras, descripcion, imagen, fecha FROM productos\_escanear ORDER BY fecha DESC.

Figura 59

 *El código muestra la sucesión de las consultas a la base de datos.*

*Nota:* Este código PHP se usa para proteger la página, consultando productos desde la base de datos y preparar los datos para mostrarlos.

Una vez que se tienen los registros almacenados en la base de datos correctamente el siguiente archivo PHP llamado producto\_escanear\_lote.php gestiona el inicio del proceso de escaneo de productos encargándose de identificar al usuario activo mediante sesión y obteniendo la lista de productos registrados en la base de datos para procesar un formulario donde se selecciona un producto y se genera un lote guardando esta información en la sesión y redirigir a la pantalla de escaneo para llevar su proceso de registro.

En la configuración del inicio de sesión se lleva a cabo mediante cada usuario y rol en la que estén registrados, con la siguiente línea de código $usuario = $\_SESSION['usuario'] ?? 'desconocido'; se obtiene el nombre del usuario autenticado desde la sesión, y si no existe, se asigna el valor "desconocido" como respaldo.

Para obtener los productos registrados se debe de ejecutar una sentencia SQL para mandar atraer los datos que están en la base de datos para ordenarlos alfabéticamente y guardando los regustados en un arreglo $productos que luego se usa para llenar un selector en el formulario.

La sentencia es la siguiente: **SELECT descripcion FROM productos\_escanear ORDER BY descripcion ASC;.**

Para procesar el formulario que envía el usuario primero se deben de evaluar el producto que se seleccionó y el lote que le corresponda según en el día juliano que este por lo que para estos campos deben de ser obligatorios rellenar, una vez que se ha completado y validado los campos ejecuta la siguiente línea de código para que permanezca mientras se esté escaneando y así rastree el producto, lote y el operador.

**$\_SESSION['producto\_seleccionado'] = $producto;**

**$\_SESSION['lote\_generado'] = $lote;**

**$\_SESSION['operador\_actual'] = $usuario;**

Si falta algún dato manda un mensaje de alerta para que se rellenen todos los campos.

**$mensaje = "⚠️ Por favor selecciona un producto y genera el lote antes de continuar.";**

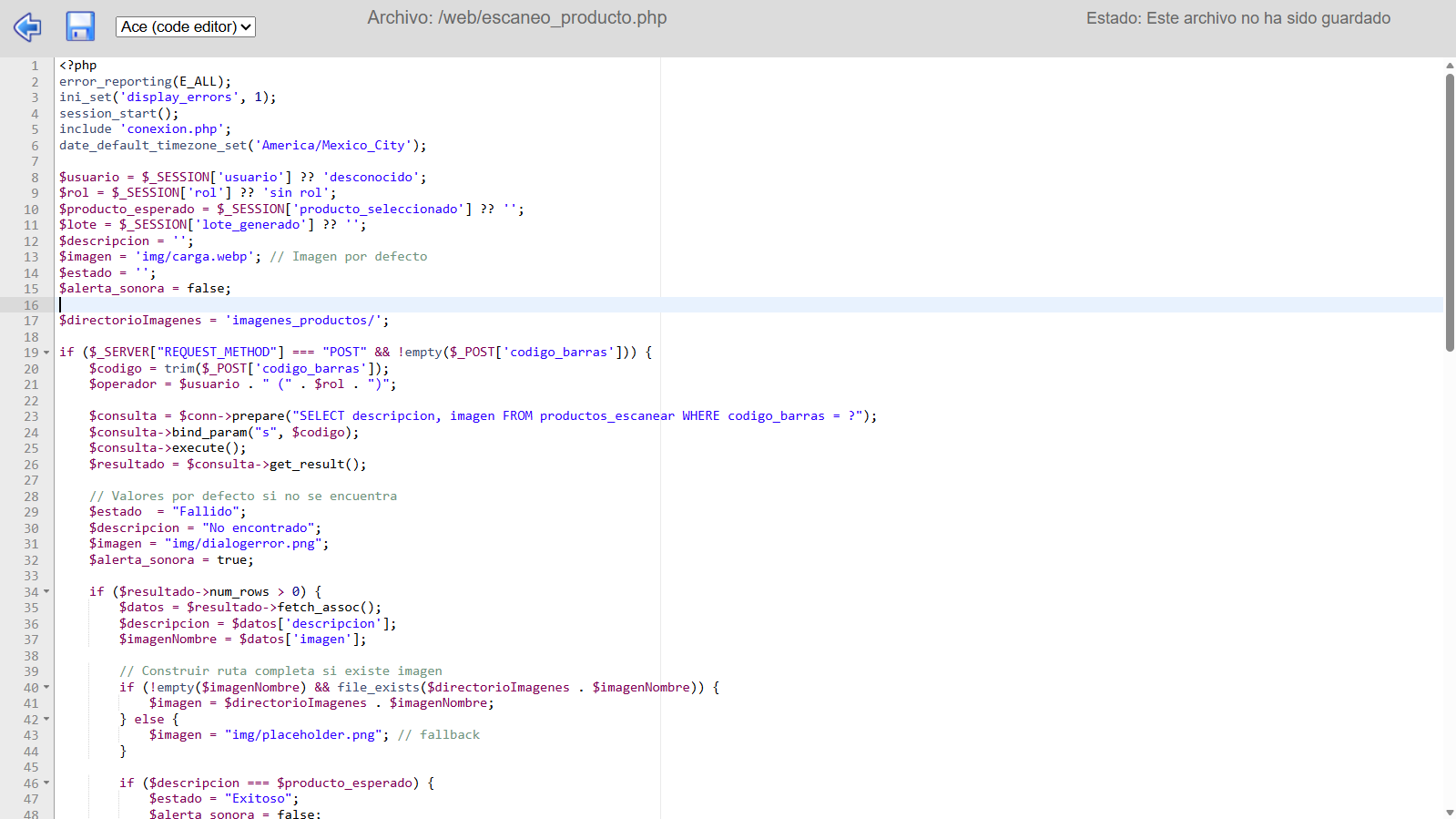
Figura 60

*El código que realiza las validaciones del usuario, producto y lote.*

*Nota:* En la imagen se puede observar las líneas de código y pasos que se siguieron para llevar un orden lógico en la programación del backend para recibir los datos desde la base de datos.

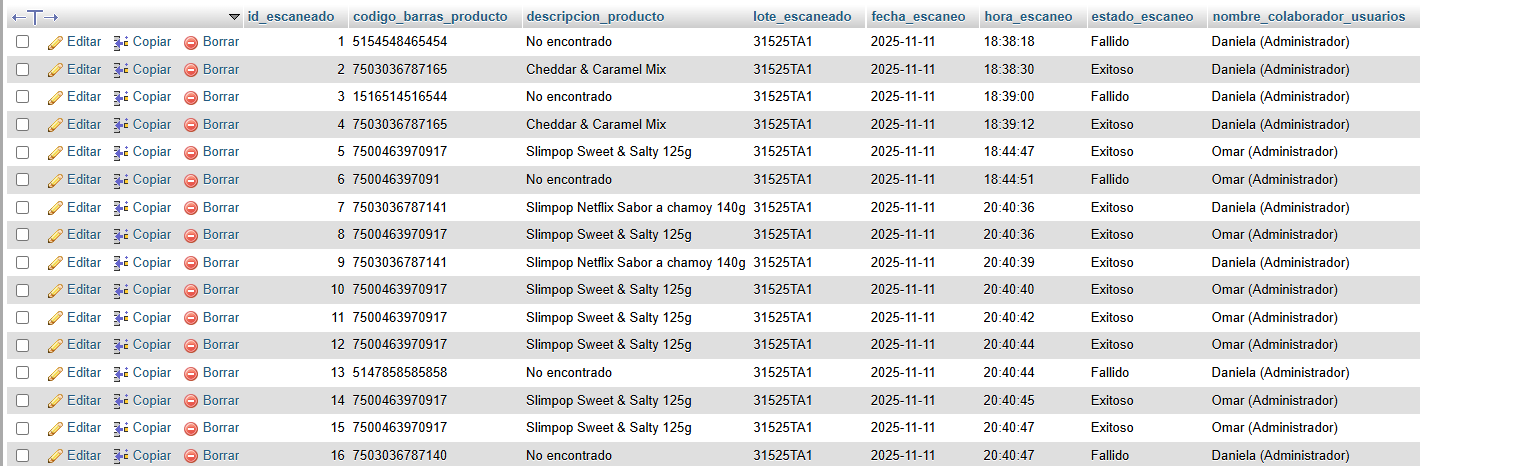
En el siguiente archivo se almacena el código que va a permitir guardar los datos en la tabla de la base de datos llamada registro\_escaneos mediante esta tabla se guardan los datos dependiendo del producto que se haya elegido marcándolo como no encontrado o el nombre del producto, así como también su estado si fue fallido o exitoso, comprobando así el producto escaneado mostrando el resultado visualmente.

Figura 61

*El código muestra la lógica para poder mandar a traerlos datos desde la base de datos.*

*Nota:* En la imagen se puede observar que el código lleva un proceso de sucesión para poder realizar cada paso y ejecutarlo cada vez que se indique.

Figura 62

*La tabla de la base de datos donde almacena los registros que se realizan.*

*Nota:* En la imagen se muestra cómo es que se están almacenando los datos una vez que se escanean.

Mediante otro archivo se pueden visualizar los productos que fueron escaneado y cuales fueron exitosos y cuales fueron fallidos, para eso solo se necesita utilizar una sentencia SQL para poder mostrar los datos.

Finamente esta es la sentencia que se utiliza para poder mandar a traer los productos escaneados: **SELECT id\_escaneado, lote\_escaneado, nombre\_colaborador\_usuarios, fecha\_escaneo, hora\_escaneo, codigo\_barras\_producto, descripcion\_producto, estado\_escaneo FROM registros\_escaneos ORDER BY fecha\_escaneo ASC, hora\_escaneo ASC;** este se ordenan de manera ascendente.

Y la otra sentencia es: **SELECT nombre\_colaborador FROM usuarios WHERE estatus='activo' ORDER BY nombre\_colaborador;** para mandar a traer los nombres de los operadores que realizan el escaneo.

Figura 63

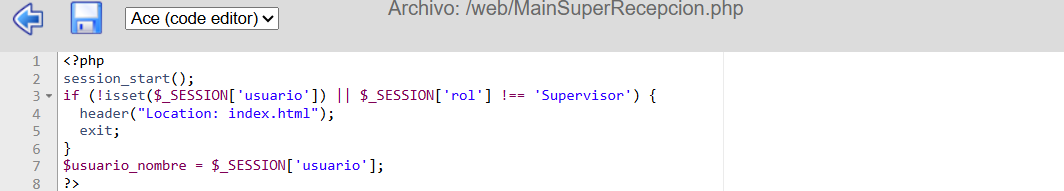
*El código muestra la conexión con la base de datos para ejecutar las sentencias SQL.*

*Nota:* En la imagen se muestra la lógica del código para extraer los datos.

**Backend para recibir los datos, validar y almacenar del módulo escanear de recepción.**

Para los usuarios con el área de recepción obtienen su direccionamiento como MainSuperRecepcion.php donde el código PHP valida que sea correcto para poder acceder a su panel correspondiente.

Figura 64

*El código valida los datos del usuario.*

*Nota:* El código realiza la verificación de rol en la que se está ingresando y también al área en la que está registrado.

El proceso de realizar el escaneo de un producto funciona de la misma manera en la que ya se había explicado anteriormente, se selecciona un producto que previamente ya se tenían guardadas en la base de datos para poder verificar su existencia.

Al inicio del código PHP se configura el inicio de sesión con las siguientes líneas de código:

**session\_start();**

**include 'conexion.php';**

**date\_default\_timezone\_set('America/Mexico\_City');**

En estas primeras líneas se realiza la configuración inicial del sistema donde la función session\_start() inicia o reanuda una sesión existente, lo cual permite guardar la información del usuario, sin esta función las variables de sesión no funcionan.

Posteriormente se obtiene el nombre del usuario que inició sesión almacenando esa información en la siguiente variable **$\_SESSION['usuario']** la cual se usa para recuperar ese dato.

La variable **$mensaje** se inicializa como una cadena vacía y se usará más adelante para mostrar mensajes de error o advertencia al usuario, por ejemplo, si no selecciona un producto.

**$usuario = $\_SESSION['usuario'] ?? 'desconocido';**

**$mensaje = "";**

Para obtener la lista de productos registrados en la base de datos se crea un arreglo vacío llamado **$productos**, donde se guardarán los nombres de los productos.

Para luego ejecutar una consulta SQL que selecciona la columna descripción de la tabla productos\_escanear, ordenando los resultados alfabéticamente (ORDER BY descripcion ASC) esto es necesario para mostrar los productos ordenados.

Sentencia: **SELECT descripcion FROM productos\_escanear ORDER BY descripcion ASC**

El while recorre cada fila del resultado usando fetch\_assoc(), que devuelve los datos como un arreglo, cada descripción obtenida se agrega al arreglo $productos, el cual luego puede usarse para mostrar los productos al usuario.

**$productos = [];**

**$resultado = $conn->query("SELECT descripcion FROM productos\_escanear ORDER BY descripcion ASC");**

**while ($row = $resultado->fetch\_assoc()) {**

**$productos[] = $row['descripcion'];**

**}**

Esta condición verifica si el formulario fue enviado usando el método POST, para verificar que el código que está dentro solo se ejecutará cuando el usuario presione el botón de enviar del formulario, y no cuando la página se carga por primera vez.

**if ($\_SERVER["REQUEST\_METHOD"] === "POST") {**

Para realizar la recuperación de los datos enviados desde el formulario se utiliza la siguiente línea de código **$\_POST['producto']** obteniendo así el producto seleccionado, y para obtener automáticamente el valor 'Recepción', lo que funciona como un valor por defecto para el sistema, el if ($producto !== '' && $lote !== '') { verifica que ningún campo este vacío.

**$producto = trim($\_POST['producto'] ?? '');**

**$lote = trim($\_POST['lote'] ?? 'Recepcion');**

**if ($producto !== '' && $lote !== '') {**

Una vez que se guardan los datos correctamente, el sistema redirige automáticamente al usuario a la página escaneo\_productoRecepcion.php, en caso de que el usuario no selecciona un producto, el sistema no avanza y en su lugar asigna un mensaje de advertencia a la variable **$mensaje**, este mensaje puede mostrarse en la interfaz para indicarle al usuario qué debe corregir antes de continuar.

**header("Location: escaneo\_productoRecepcion.php");**

**} else {**

**$mensaje = "⚠️ Por favor selecciona un producto antes de continuar.";**

**}**

Figura 65

*Se muestra las líneas de código*

*Nota:* En la imagen se muestra el procedimiento que se debe de realizar para que se manden a traer los datos que están almacenados en la base de datos.

Para llevar a cabo el escaneo del producto que se seleccionó anteriormente se necesita verificar que el usuario sea el correcto, y que los campos estén activos como el producto y el lote que en este caso le corresponde al mensaje de Recepción.

**$usuario = $\_SESSION['usuario'] ?? 'desconocido';**

**$rol = $\_SESSION['rol'] ?? 'sin rol';**

**$producto\_esperado = $\_SESSION['producto\_seleccionado'] ?? '';**

**$lote = $\_SESSION['lote\_generado'] ?? '';**

Estas variables controlan la respuesta visual y lógica del escaneo ya que almacenan información que es fundamental para mostrar al usuario como la variable **$descripcion** almacenará el nombre del producto encontrado en la base de datos y la variable **$imagen** tiene una imagen por defecto que se muestra mientras no se realice el escaneo, mientras que la variable **$estado** indicará si el escaneo fue exitoso o fallido ejecutando una alerta sonora que indica que no fue encontrado, el audio se almacena en la variable $alerta\_sonora.

**$descripcion = '';**

**$imagen = 'img/carga.webp';**

**$estado = '';**

**$alerta\_sonora = false;**

También se construye una cadena que identifica completamente al operador, incluyendo su rol, lo cual será útil para los registros históricos, posteriormente se prepara una consulta SQL usando sentencias preparadas para la seguridad.

Sentencia utilizada: **SELECT descripcion, imagen FROM productos\_escanear WHERE codigo\_barras = ?");**

El código de barras se pasa como parámetro y se ejecuta la consulta para buscar el producto correspondiente en la base de datos.

**$consulta = $conn->prepare("SELECT descripcion, imagen FROM productos\_escanear WHERE codigo\_barras = ?");**

**$consulta->bind\_param("s", $codigo);**

**$consulta->execute();**

**$resultado = $consulta->get\_result();**

En caso de que el código no se encuentra en la base de datos, el sistema automáticamente marcará el escaneo como fallido, mostrando una imagen de error y activará una alerta sonora, de esta forma, el sistema siempre tiene un comportamiento definido.

**$estado = "Fallido";**

**$descripcion = "No encontrado";**

**$imagen = "img/dialogerror.png";**

**$alerta\_sonora = true;**

Si la consulta devuelve resultados, significa que el código de barras sí existe, por lo que se extraen la descripción y el nombre de la imagen del producto en caso de no encontrar la imagen muestra una por defecto.

Después, se compara la descripción del producto escaneado con el producto esperado, si coinciden exactamente, el escaneo se marca como Exitoso y se desactiva la alerta sonora.

**if (!empty($imagenNombre) && file\_exists($directorioImagenes . $imagenNombre)) {**

**$imagen = $directorioImagenes . $imagenNombre;**

**} else {**

**$imagen = "img/placeholder.png";**

**}**

**if ($descripcion === $producto\_esperado) {**

**$estado = "Exitoso";**

**$alerta\_sonora = false;**

Para llevar un control correcto en la base de datos los registros que se realizan al día se deben de almacenar para tener un reporte de lo realizado esto es muy importante para auditorías y control de calidad.

Finalmente, se cierran las consultas preparadas para liberar recursos del servidor y mantener el sistema eficiente y ordenado.

**$registro = $conn->prepare("INSERT INTO registros\_escaneos ( codigo\_barras\_producto, descripcion\_producto, lote\_escaneado, fecha\_escaneo, hora\_escaneo, estado\_escaneo, nombre\_colaborador\_usuarios ) VALUES (?, ?, ?, CURDATE(), CURTIME(), ?, ?)");**

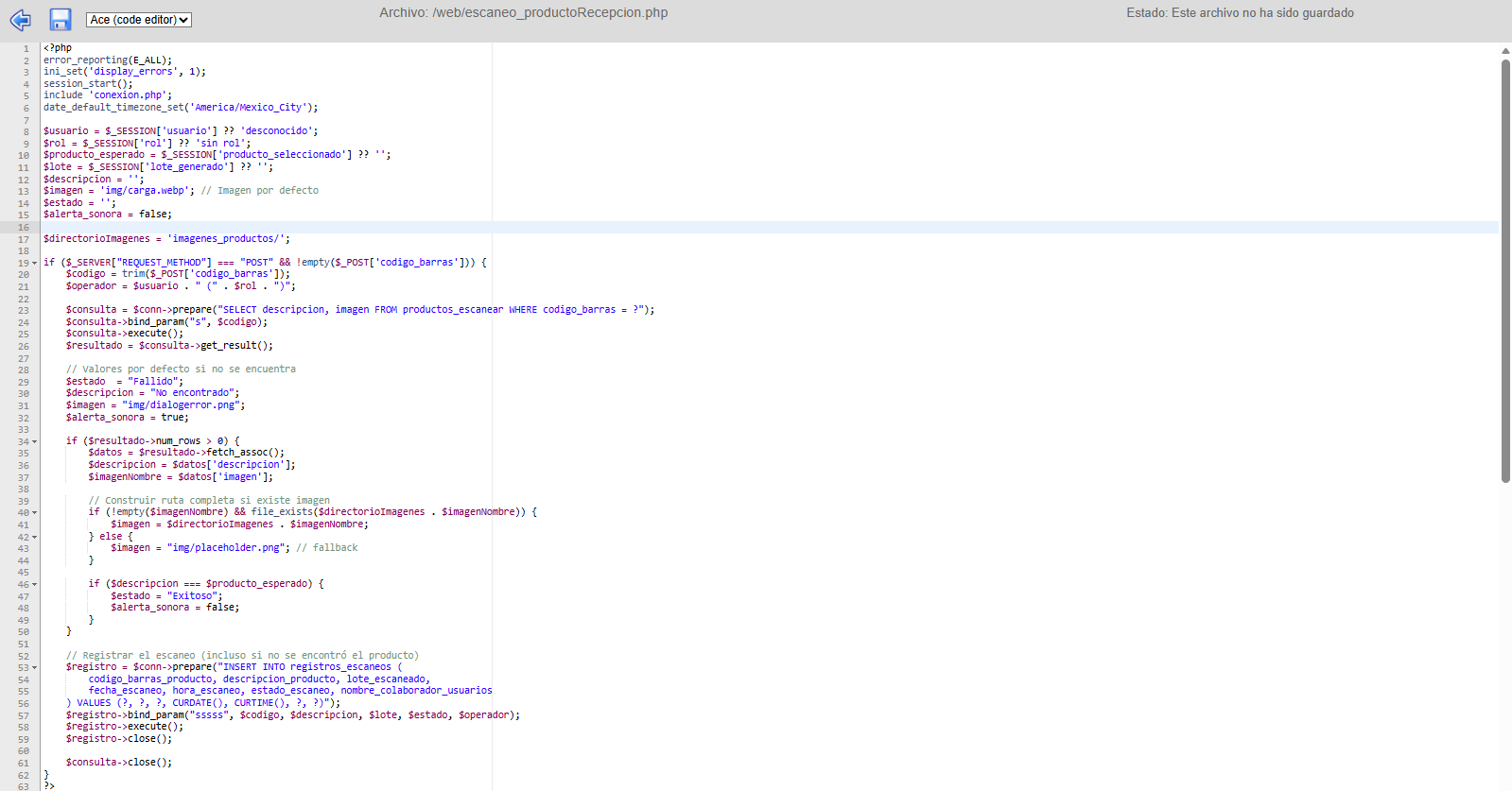
**$registro->bind\_param("sssss", $codigo, $descripcion, $lote, $estado, $operador);**

**$registro->execute();**

**$registro->close();**

**$consulta->close();**

Figura 66

*El código muestra el proceso para extraer la información de la base de datos.*

*Nota:* El código realiza las validaciones de los registros de la base de datos.

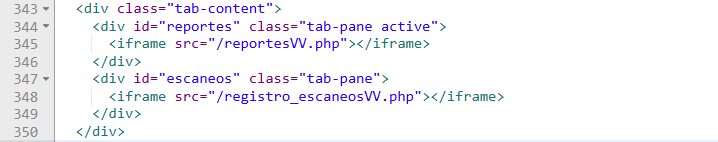
**Backend para recibir los datos, validar y almacenar del módulo inventarios del administrador.**

**Backend para recibir los datos, validar y almacenar del módulo inventarios del supervisor.**

**Backend para recibir los datos, validar y almacenar de generar reportes del administrador, supervisor.**

Para generar reportes de manera más rápida se crea una clase en el archivo reportes.html para extraer la información de los anteriores archivos para se utiliza la siguiente línea de código.

Figura 67

*En la imagen se muestras los iframe que se mandan a traer.*

*Nota:* Mediante el siguiente código se muestra la extracción de archivos para mostrar la información.

**Backend para recibir los datos, validar y almacenar para el registro de usuarios del administrador.**

En archivo de usuarios.php es el encargado de llevar los registros de los usuarios en la base de datos para eso se tiene una tabla que contiene los campos que se necesitan para que el usuario sea registrado donde este código inicia con session\_start(), lo cual permite acceder y manejar variables de sesión en PHP ya las sesiones se usan para mantener información del usuario entre distintas páginas, para saber si un usuario ya inició sesión y qué permisos tiene dentro del sistema.

Después, el bloque de verificación de sesión se encarga de comprobar si existe la variable $\_SESSION['usuario'] y si no está vacía, esto sirve para asegurar que solo usuarios autenticados puedan acceder a este archivo en caso de que la sesión no exista o está vacía, el usuario es redirigido automáticamente a index.html, la cual sería la página de inicio de sesión, y el script se detiene con exit() para evitar que el resto del código se ejecute.

Luego se guarda el nombre del usuario administrador en la variable $admin\_usuario, obteniéndolo directamente desde la sesión, esta variable se utilizará más adelante para validar acciones sensibles, como la confirmación de la contraseña del administrador.

En la sección de conexión a la base de datos, se definen las variables necesarias para conectarse a MySQL: el servidor (localhost), el usuario, la contraseña y el nombre de la base de datos, para crear un objeto mysqli, que es una clase nativa de PHP utilizada para manejar conexiones y consultas a bases de datos MySQL, en caso de que ocurra un error al conectarse, el programa se detiene mostrando el mensaje del error, además, se configura el conjunto de caracteres utf8mb4 para permitir el uso correcto de acentos, emojis y caracteres especiales este siendo opcional si es que se desea agregar caracteres.

El las acciones de actualización de estatus se ejecuta cuando se recibe una petición POST con la acción toggle\_estatus, este bloque sirve para activar o desactivar a un usuario recibiendo así el número de colaborador y el nuevo estatus, y se ejecuta una consulta UPDATE usando sentencias preparadas.

Las sentencias preparadas ayudan a prevenir inyecciones SQL por lo que una vez que se ejecuta la actualización, el servidor responde con un JSON indicando que la operación fue exitosa.

La sección de actualización de datos de usuario permite modificar la información de un colaborador, como su nombre, rol, área y contraseña, esta lógica se activa cuando la acción recibida es editar\_usuario.

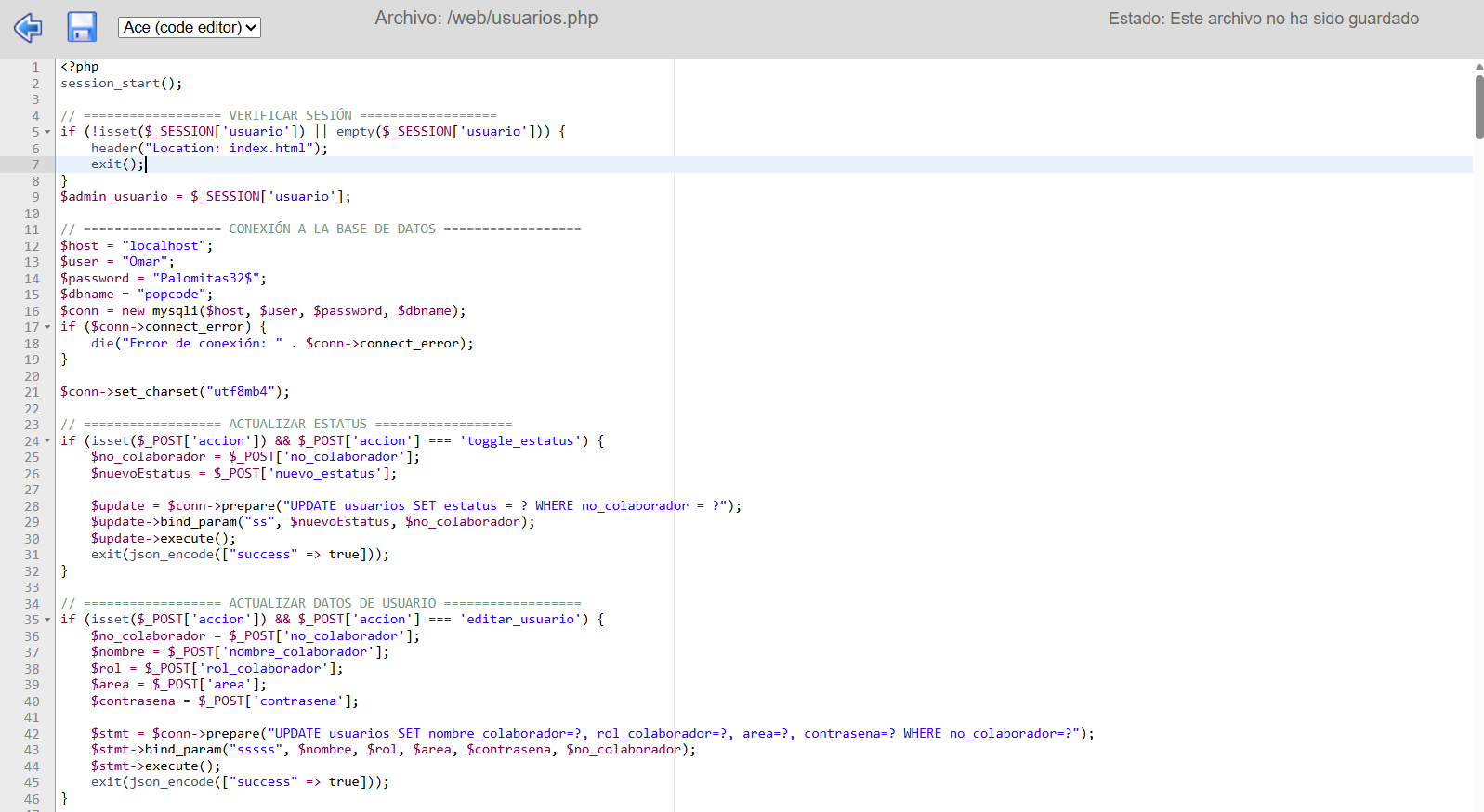
Por lo que se recopilan los datos enviados por el formulario, ejecutando así una consulta UPDATE preparada y se responde con un mensaje JSON de éxito, en este bloque es comúnmente usado en formularios de edición de usuarios dentro de paneles.

Para que se pueda visualizar la contraseña que tiene el usuario el sistema debe de verificar la solicitud si la contraseña ingresada coincide con la del administrador que tiene la sesión activa o inicializada, realizando una consulta a la tabla usuarios buscando al usuario con rol de “Administrador” en caso de que sea encontrada como correcta se devuelve un JSON indicando que la validación fue correcta de lo contrario, se indica que es inválida esto suele usarse como una capa extra de seguridad antes de realizar acciones críticas.

Para poder consultar de usuarios se de construir dinámicamente una consulta SQL con filtros opcionales inicializando las variables para almacenar condiciones y parámetros si se recibe una petición POST sin una acción específica, se toman en cuenta los filtros enviados, como el tipo de usuario o un texto de búsqueda, en estos filtros se agregan a la cláusula WHERE de la consulta de manera segura usando parámetros preparados.

Finalmente, se lleva a cabo una consulta SQL completa para obtener los usuarios de la base de datos, en caso de que existan parámetros se asignan dinámicamente usando bind\_param d etal manera que la consulta se ejecuta y se obtiene el resultado, el cual se guarda en $result, donde también se obtiene el número total de usuarios.

Figura 68

*El código que se muestra en la imagen es la lógica que se debe de seguir para obtener los datos de la base de datos.*

*Nota:* mediante estas líneas de código se obtienen los datos que están almacenados en la base de datos

Figura 69

*Continuación del código*



*Nota:* La imagen muestra la continuación del código.

**Backend para recibir los datos, validar y almacenar el registro de usuarios para el supervisor del área de calidad. AQUÍ ME QUEDE**

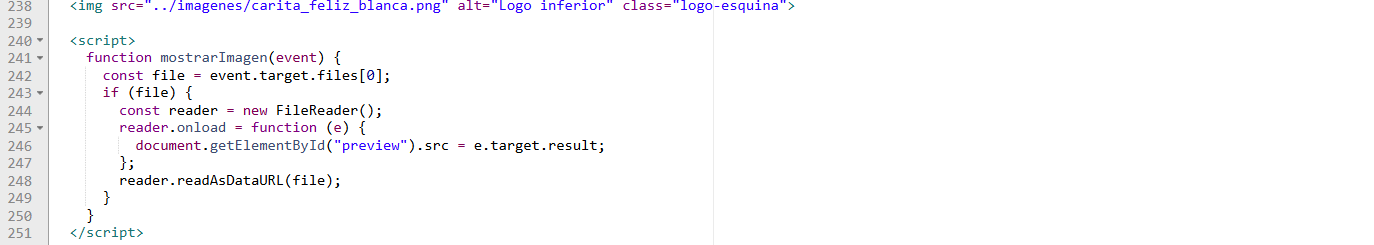
Esta área de calidad es la única con rol de supervisor que puede realizar los registros de usuarios que pertenezcan a calidad con el rol de operador

### Programar el frontend para mostrar información al administrador.

Añadir.php

Para el caso de la imagen se necesitó realizar un script para mostrar la imagen donde primero se obtiene el archivo que se haya seleccionado verificando así que exista, finalmente se ejecuta el código cuando el archivo termina de cargar para asignar la imagen cargada al elemento, posteriormente lee el archivo como una URL.

Figura

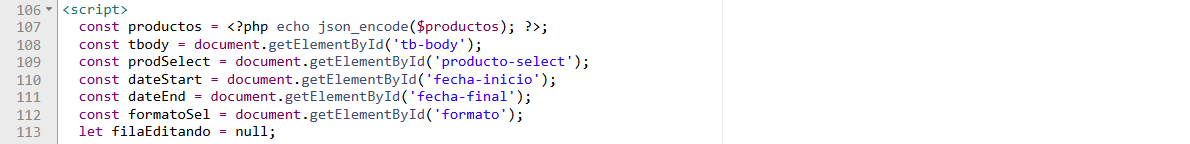
*Script ara visualizar la imagen en los archivos.*

*Nota:* En la imagen se muestra cómo es que se manda a traer el archivo para asignarlo como URL.

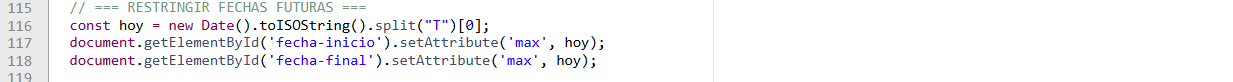
Reportes.php

El script que se llevó a cabo en este archivo gestiona un listado de productos que provienen de PHP, permitiendo mostrar, filtrar, editar, eliminar y generar reportes en PDF o Excel desde una tabla HTML. Además, controla modales, validaciones de fechas y la comunicación con el servidor mediante fetch.

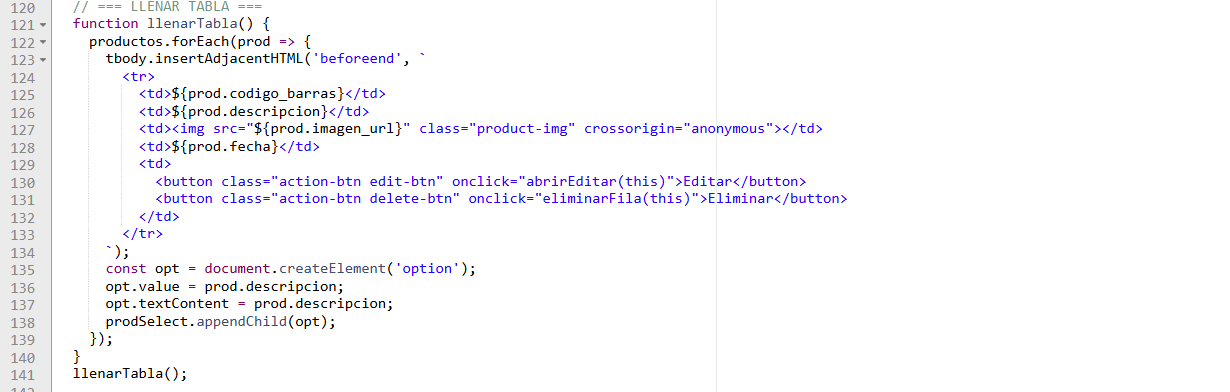
Al inicio del script se declaran constantes y variables globales las cuales van a ser utilizadas más adelante permitiendo acceder fácilmente a los elementos del DOM y mantener el estado de edición.



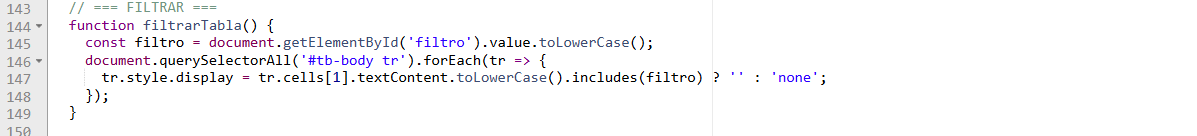
También se obtiene la fecha actual y se asigna como valor máximo (max) a los campos de fecha inicio y fin, esto evita que el usuario seleccione fechas futuras en los filtros del reporte.

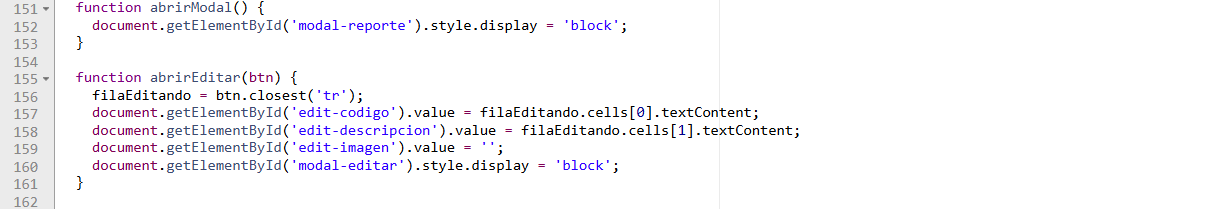


Existe una función llamada llenarTabla () mediante este permite insertar dinámicamente filas <ti> en la tabla con los campos siguientes: código de barras, descripción, imagen, fecha, y como opciones de realizar una opción de editar y eliminar, también llena el <Selecta> de productos para los reportes ejecutándolo automáticamente al cargar el script.

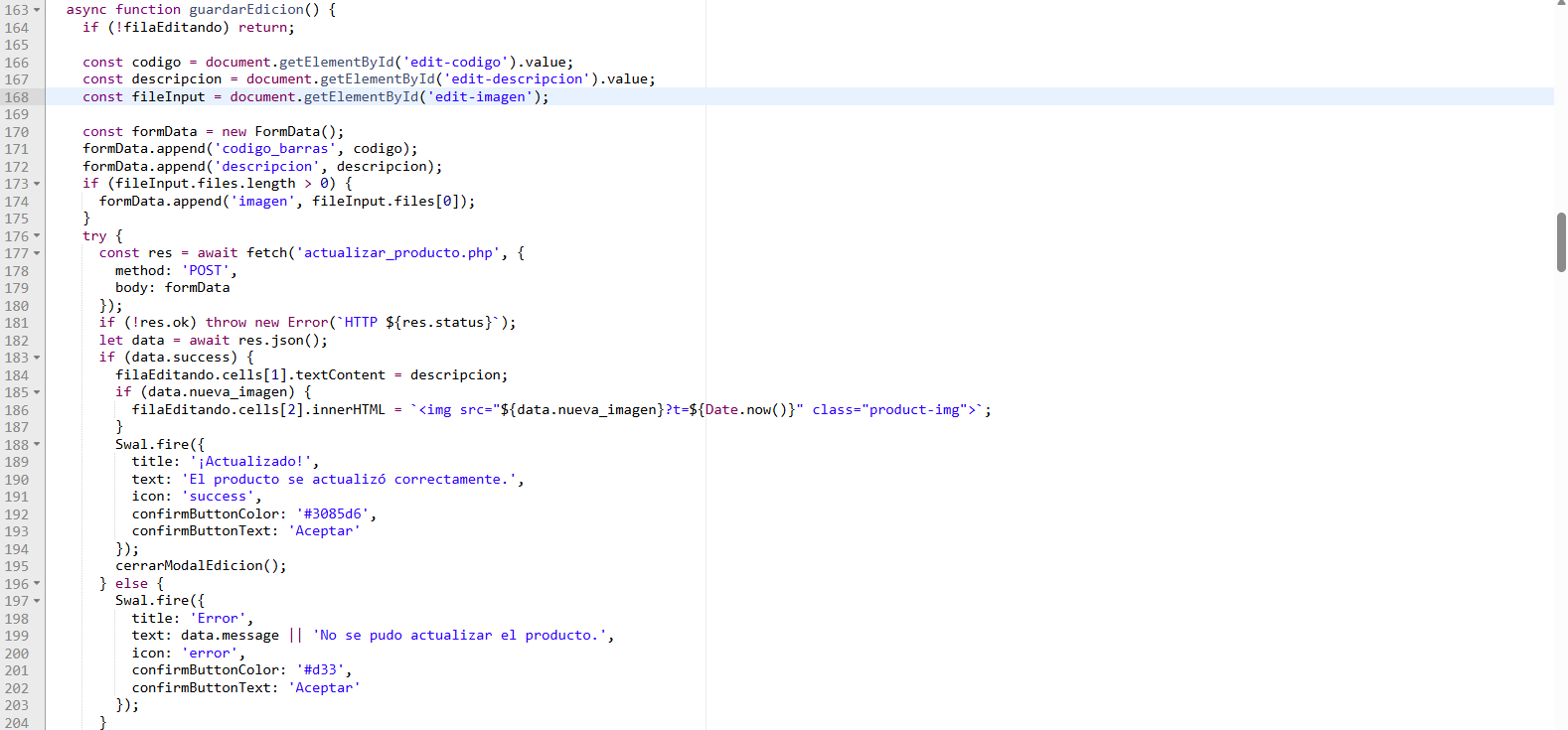


Para la función filtrarTabla () permite realizar un filtro de los productos por descripción, leyendo el texto ingresado en el campo de búsqueda, mostrando así los registros con coincidencias.

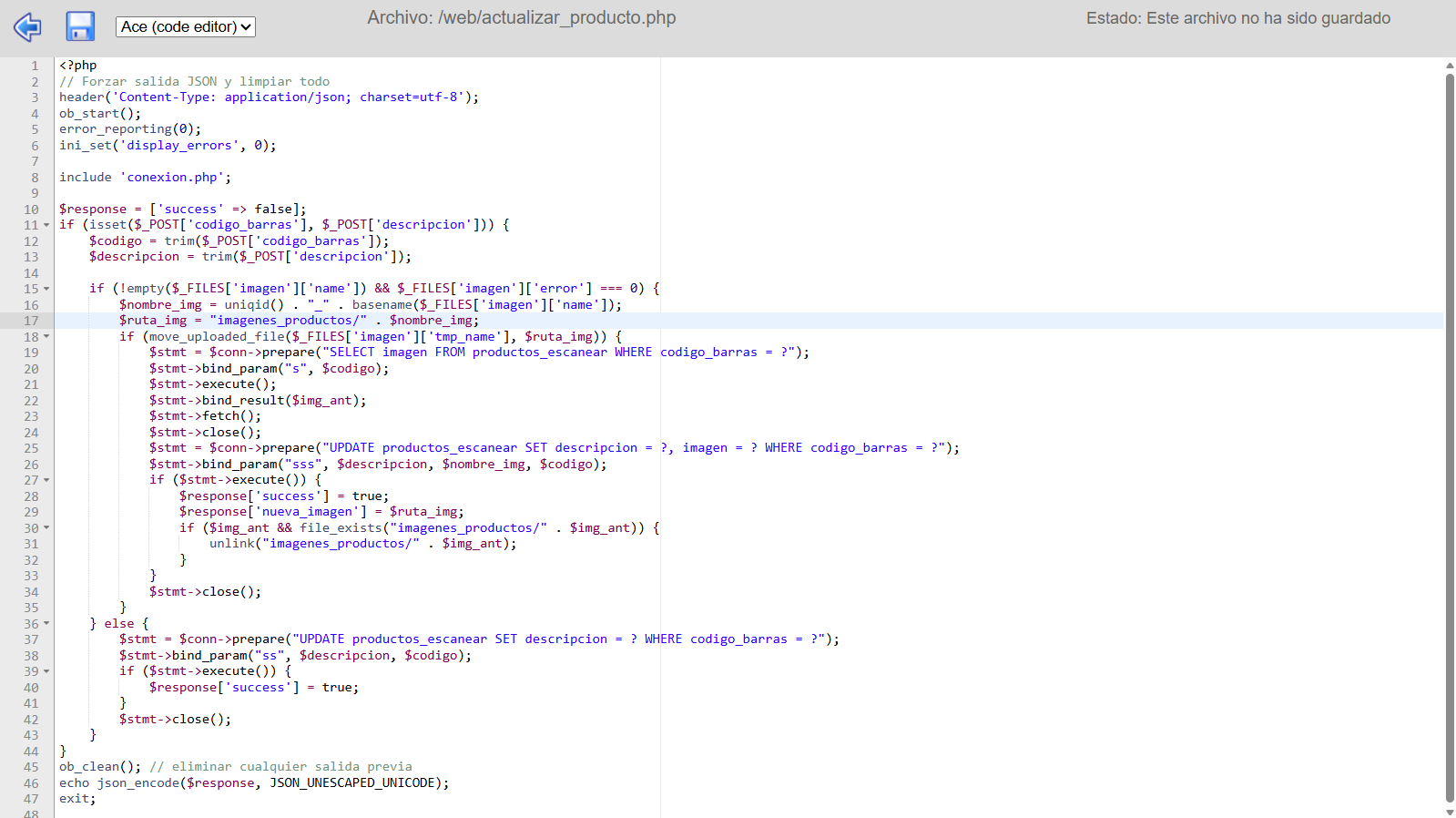


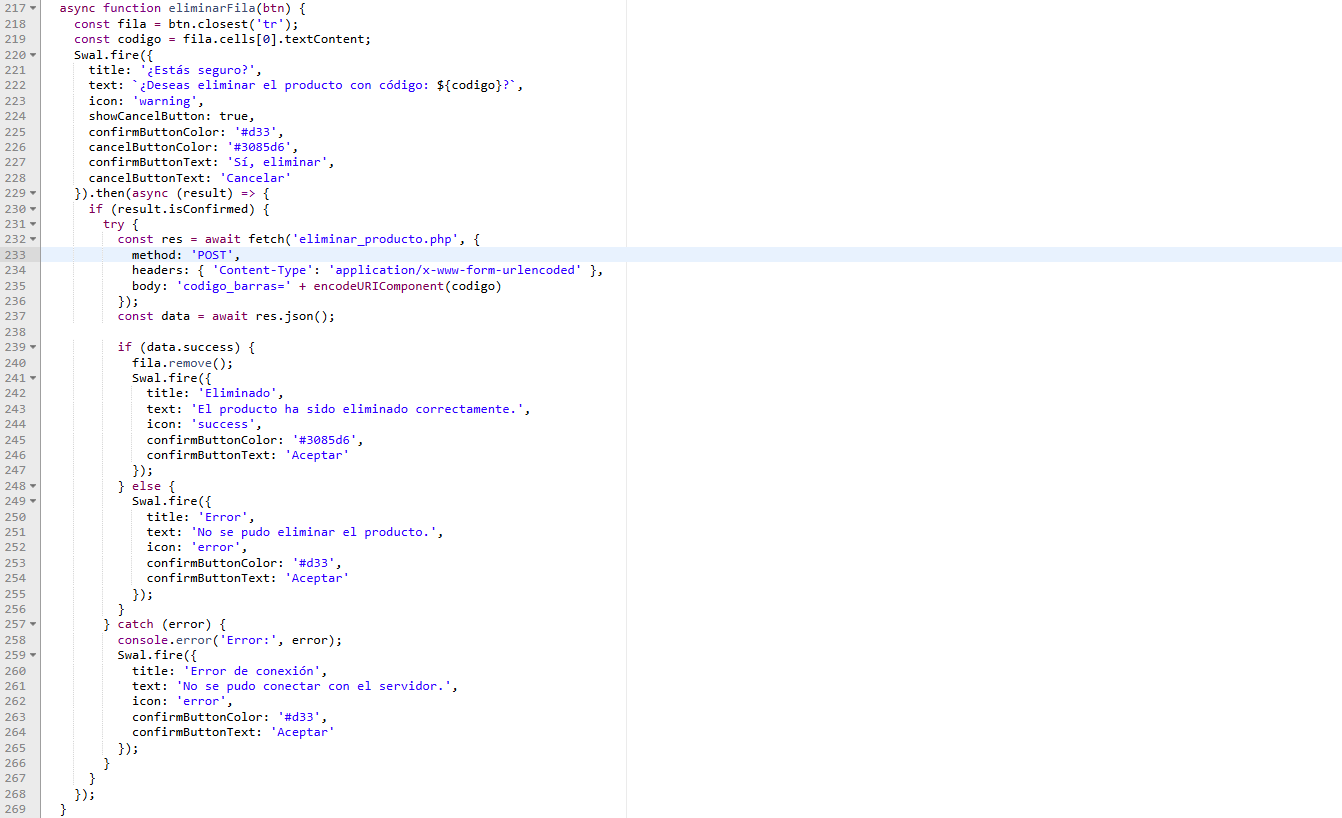
En las funciones modales de abrirModal (), abrirEditar(btn), cerrarModalEdicion (), se muestran el modal para poder generar reportes, abre el modo edición y carga de datos de la fila seleccionada y cierra el modal limpiando la referencia de la fila editada. 

La función que se tiene codificada también es la de guardarEdicion () ya que este se encarga de tomar los datos editados del formulario para crear un nuevo objeto FormData y enviar los datos al servidor con el archivo actualizar\_producto.php este utilizando un fetch, mostrando al final un mensaje de la actualización exitosa este por medio de SweetAlert.



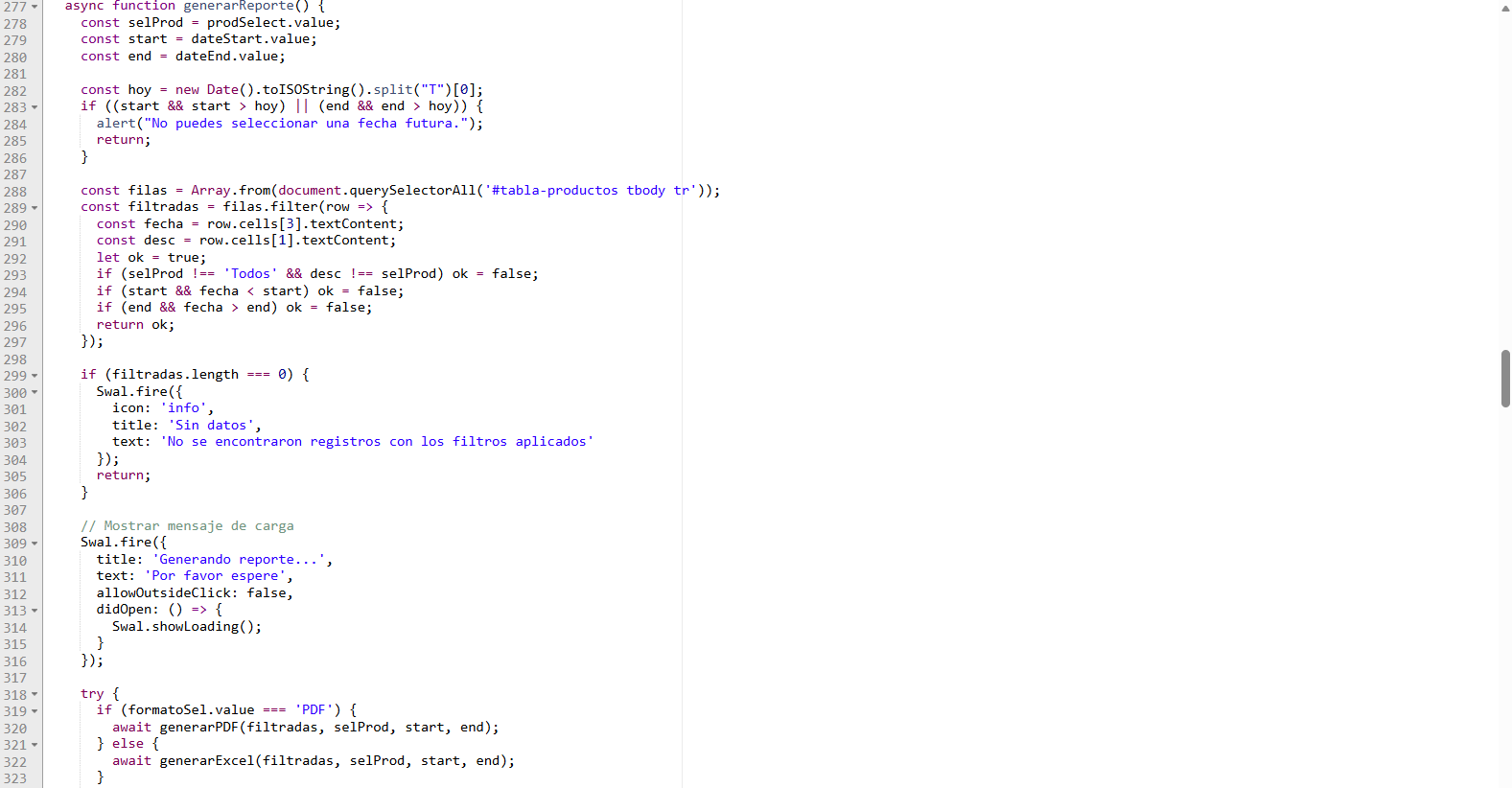
Este archivo actualizar\_producto.php se encarga de actualizar un producto en la base de datos, permitiendo modificar la descripción y opcionalmente la imagen.

El script responde exclusivamente en formato JSON, ya que está diseñado para ser consumido mediante una petición AJAX (fetch) desde el frontend. 

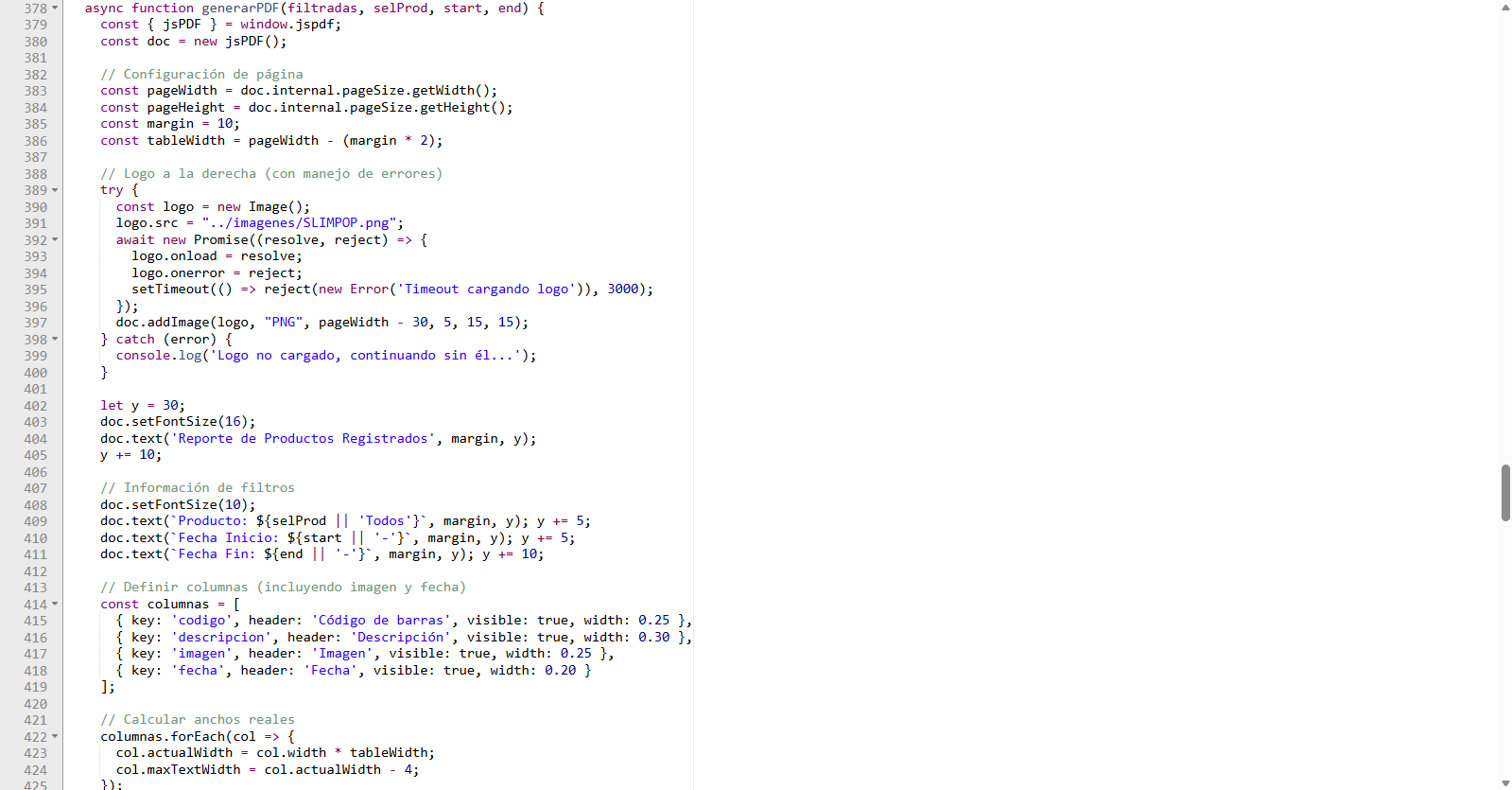
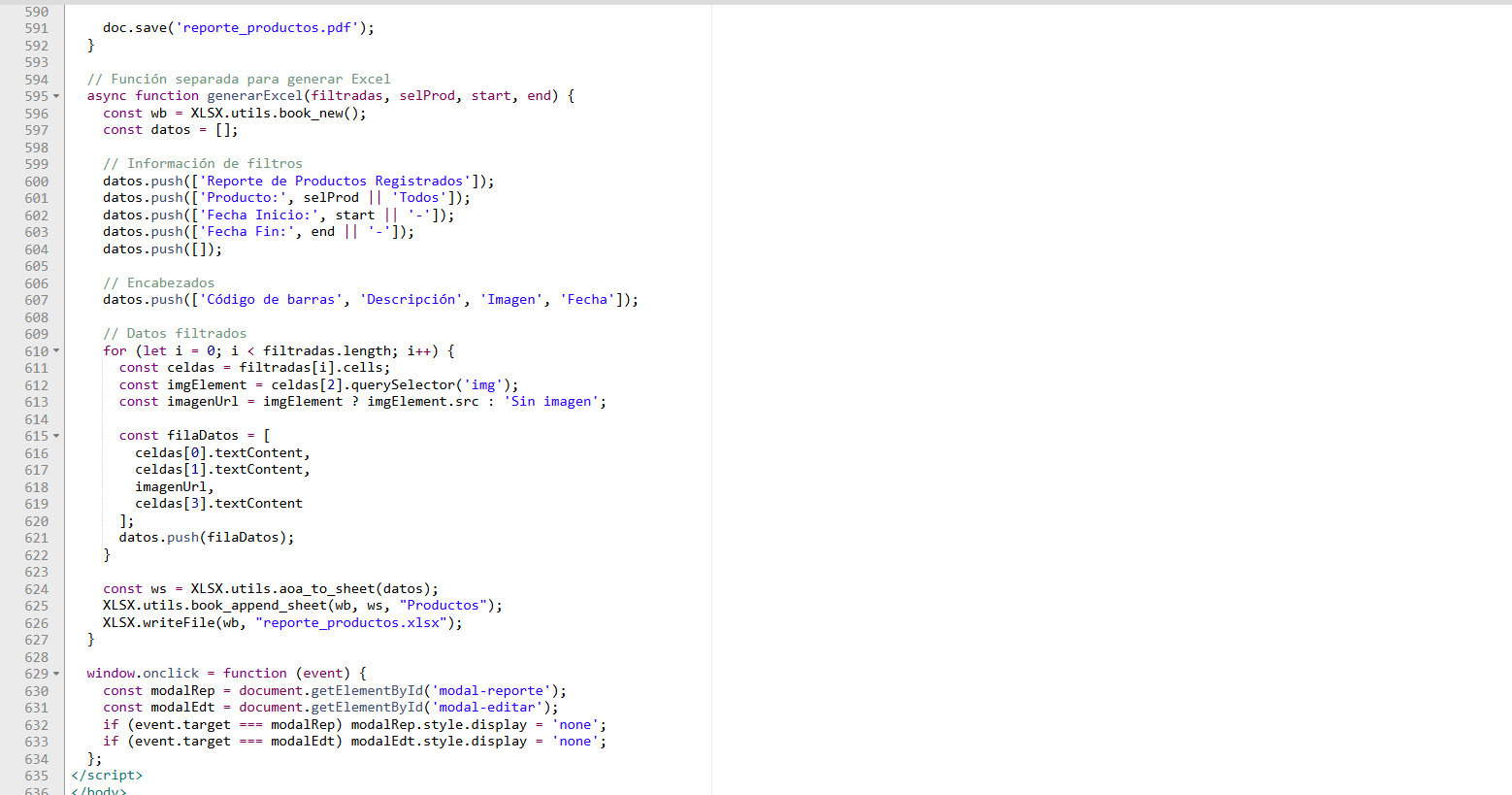
En la función eliminarFila muestra una confirmación antes de poder eliminarlo por completo, enviando el código del producto al servidor, archivo(eliminar\_producto.php), si el servidor responde correctamente elimina la fila de toda la tabla manejando una serie de éxito o errores de conexión y respuestas fallidas.

El archivo eliminar\_producto.php se encarga de eliminar un producto de la base de datos utilizando su código de barras. Además, si el producto tenía una imagen asociada, el script también elimina el archivo físico del servidor la respuesta se devuelve en formato JSON, ya que es consumida por una petición AJAX (fetch) desde JavaScript.

generarReporte () es otra de las funciones que se integraron en el script para controlar la generación de reportes, aplicando los filtros por producto y rango de fechas, así como validar que no se usen fechas futuras esto para generar un control de tiempos, genera los formatos seleccionados (PDF, Excel).



Por último, la función de generarPDF y generarExcel se utilizan para crear los reportes correspondientes, y para el caso del PDF se utilizó la librería jsPDF para mayor flexibilidad, y para el Excel se utilizó la librería de SheetJS(XLSX). De esta, manera se configuran las páginas las cuales va a tener el documento, así como ajustar las imágenes del logo, mostrando los filtros que se apliquen, controlando los saltos de páginas, inserta encabezados y por último guarda el archivo según e formato seleccionado.

Se utilizan algunas funciones auxiliares que permitir que al cargar la imagen funcione de forma asíncrona para usarlas en el pdf, y también para ajustar los textos se utilizó el dividirTexto(doc, texto, maxWdth).

### Programar el frontend para mostrar información al supervisor.

### Programar el frontend para escanear del operador.

### Configurar la base de datos en XAMPP.

### Integrar la lógica de activación de alarmas desde el lector.

## Fase 4

## Verificación

### Pruebas de la base de datos

### Pruebas unitarias de cada módulo (escaneo, validación, almacenamiento).

### Pruebas de integración del sistema completo

### Pruebas de rendimiento de la velocidad de respuesta, carga de datos.

### Pruebas de error

### Validación con usuarios

## Fase 5

## Mantenimiento

### Corrección de errores

### Hacer actualizaciones

### Manual de usuario

### Manual técnico