PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

Unidad Académica de Formación Técnica y Tecnológica – PUCE TEC

Sistema de automatización de pedidos e inventario para el restaurante Pollos a la Brasa del Valle

Proyecto de titulación previo a la obtención del título de: Tecnología en Desarrollo de Software

Autores:

Josue Joel Garcia del Valle

Julio César Guaña Viana

Tutor:

Carlos Miguel Cardenas Riofrio

Quito, Ecuador

2025

Tabla de contenidos

DECLARACIÓN y AUTORIZACIÓN

Yo, Julio César Guaña Viana **con C.I** 1724215460 autor(a) del trabajo de titulación intitulado: “Sistema de automatización de pedidos e inventario para el restaurante Pollos a la Brasa del Valle”, previa a la obtención del título de Tecnología en desarrollo de software en la Unidad Académica de Formación Técnica y Tecnológica PUCE TEC:

1.- Declaro tener pleno conocimiento de la obligación que tiene la Pontificia Universidad Católica del Ecuador, de conformidad con el artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de graduación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la Pontificia Universidad Católica del Ecuador a difundir a través de sitio web de la Biblioteca de la PUCE el referido trabajo de titulación, respetando las políticas de propiedad intelectual de Universidad.

Quito, 15 de febrero de 2025

Julio César Guaña Viana

C.I. 1724215460

DECLARACIÓN y AUTORIZACIÓN

Yo, Josue Joel Garcia Del Valle **con C.I** 1729222784 autor(a) del trabajo de titulación intitulado: **"Sistema de automatización de pedidos e inventario para el restaurante Pollos a la Brasa del Valle"**, previa a la obtención del título de Tecnólogo Superior en Desarrollo de Software en la Unidad Académica de Formación Técnica y Tecnológica PUCE TEC:

1.- Declaro tener pleno conocimiento de la obligación que tiene la Pontificia Universidad Católica del Ecuador, de conformidad con el artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de graduación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la Pontificia Universidad Católica del Ecuador a difundir a través de sitio web de la Biblioteca de la PUCE el referido trabajo de titulación, respetando las políticas de propiedad intelectual de Universidad.

Quito, 15 de febrero del 2025

Josue Joel Garcia Del Valle

C.I. 1729222784

Agradecimientos

Agradecimiento Julio César Guaña Viana

Quiero expresar mi más sincero agradecimiento en especial a mi familia que me ha apoyado en todo este proceso de la carrera de la Tecnología en Desarrollo de Software, a mis padres que hacían el esfuerzo de irme a ver a la universidad con la finalidad de llegar tranquilo a la casa, a mi hermana que me da la motivación de seguir mejorando como persona, a mi abuela que me brindo todo su apoyo en esto y sobre todo a mi abuelo que es la pieza fundamental en mi vida.

A mis amigos de la universidad, personas dedicadas que me han ayudado a mejorar en todo este proceso, con sus consejos, con su ayuda en temas que se me complicaban, agradecer a cada uno de ellos que se han convertido en una parte fundamental de este proceso.

A mis amigos de toda la vida, personas que desde el colegio han estado para mí, personas valiosas que me han demostrado su apoyo durante todo este proceso con sus consejos, y ayuda.

A Dios todos mis logros, por ser la base de todo esto, quien me ha ayudado en los momentos malos, en momentos de incertidumbre y de miedo, por formar parte de esto y de todos los logros futuros en mi vida profesional.

Agradecimientos Josue

Quiero expresar mi más sincero agradecimiento a mi mamá y a mi papá, por su amor, apoyo incondicional y por brindarme siempre la fortaleza necesaria para superar cualquier obstáculo. Gracias por estar siempre a mi lado, creyendo en mí y motivándome a seguir adelante.

A mi querido amigo Isaac, quien me brindó su apoyo en un momento clave de mi vida, ayudándome en un trabajo importante. Gracias por tu generosidad, por estar allí cuando más te necesité y por tu amistad, que ha sido una gran fuente de motivación.

A mi pareja, por su amor y apoyo constante. Gracias por estar a mi lado, por tu paciencia y por comprender los desafíos que este proceso implicó. Tu apoyo ha sido invaluable.

A mis amigos de la universidad, quienes me acompañaron durante todo este proceso, brindándome su apoyo y compartiendo experiencias que hicieron más llevadero este camino. Su amistad ha sido un pilar fundamental en esta etapa.

Introducción

La manera en que los negocios de hoy en día logran salir a flote es mediante la adopción de recursos tecnológicos como lo son los sistemas de automatización que faciliten procesos dentro del negocio, sin embargo, para los negocios de menor escala, estos recursos suelen ser difíciles de manejar.

Por lo cual esta tesis tiene como objetivo central demostrar que un sistema de automatización de pedidos e inventario en un negocio popular como en este caso un asadero de pollos puede llegar a ser fácil de entender y manejar, todo esto aplicando herramientas y nuevas tecnologías para garantizar la eficacia y adaptabilidad del programa para diferentes este y otros tipos de negocios, con funcionalidades como la creación de perfiles, la asignación de roles y el manejo de información estadística para llevar la contabilidad de vetas.

Para desarrollar este proyecto se llevó a cabo el uso de tecnologías modernas como lo son Express.js y Node.js dentro del backend, React en todo lo que es el frontend, esto es de suma importancia para una interfaz de usuario dinámica. Finalmente, PostgreSQL se usó para la generación de la base de datos, junto con librerías como Sweetalert2, Chart.js y React-Bootstrap, que ayudarán a generar una mejor experiencia al usuario y facilitarán la presentación de datos y alertas. Todas estas tecnologías serán explicadas en detalle en el capítulo 2.

La Tesis se estructura por tres capítulos principales, en los que se abordan varios temas que se enlazan entre sí. Capítulo 1 aborda el planteamiento principal del proyecto, así como los objetivos, requerimientos y diseño de este, Capítulo 2 habla de la

construcción del sistema, en donde se explica más a detalle del proceso y las tecnologías utilizadas para el desarrollo del proyecto, finalmente el Capítulo 3 especifica acerca de las pruebas que se hacen al sistema para conocer a detalle sobre el nivel de solicitudes que éste puede manejar junto con otras pruebas que ha sido sometido el sistema.

Capítulo I

Levantamiento de Requisitos y Diseño del Sistema

Objetivo del capítulo:

Documentar y establecer la estructura de los objetivos funcionales y no funcionales del sistema de automatización del restaurante Pollos a la Brasa del Valle, a su vez explicar la metodología aplicada en el desarrollo de este proyecto y la manera en la que se organizó el equipo para llevar a cabo el sistema.

Requisitos funcionales

- **Gestión de pedidos**

RF01: El sistema permite al usuario crear pedidos mediante la selección de los productos del menú.

RF02: El sistema registra la fecha de manera automática en el momento en el que se realiza el pedido.

RF03: El sistema reduce la cantidad de productos de manera automática al momento de hacer un pedido.

RF04: El sistema permite la verificación de los pedidos mediante un apartado donde se nos muestre la información de estos.

- **Visualización de menú y pedidos**

RF05: El sistema mostrara por separado el apartado del menú con los pedidos, así como el aparatado de comidas y bebidas dentro de menú.

RF06: El sistema mostrara la información de los pedidos en el frontend con todos los pedidos realizados.

RF07: La información de los pedidos que se mostrarán en pantalla serán id del pedido, fecha del pedido, nombre del producto, cantidad, precio por unidad, y total del producto.

- **Roles y permisos**

RF08: El sistema contendrá dos tipos de roles que se diferenciaran por su jerarquía, así como a su vez de sus permisos dentro del mismo.

RF09: El usuario administrador será capaz de acceder a todas las funcionalidades del sistema.

RF10: El usuario empleado únicamente será capaz de tomar pedidos.

Requisitos no funcionales

- **Rendimiento**

RNF01: El sistema responderá a las solicitudes de pedidos en un aproximado de 3 a 4 segundos.

RNF02: El apartado de backend deberá ser capaz de manejar la cantidad de pedidos que se manejen en el negocio durante un día.

- **Escalabilidad**

RNF03: La arquitectura del sistema nos deberá permitir incorporar nuevos módulos en un futuro, para mejorar las características de este sin necesidad de volverlo a estructurar.

- **Mantenimiento**

RNF04: El código debe estar documentado para futuras actualizaciones del sistema.

- **Usabilidad**

RNF05: La interfaz del sistema debe ser fácil e intuitiva para el usuario que opere el sistema.

Diseño del sistema:

1. **Arquitectura del sistema**

La arquitectura presentada en el diseño del sistema de automatización de pedidos e inventario para el restaurante Pollos a la Brasa del Valle es cliente-servidor, que básicamente consta de dos partes el proveedor y el consumidor, que interactúan entre sí para dar funcionamiento al sistema, compuesto por tres capas Backend, Frontend y Base de Datos.

Primera capa: Base de datos, el sistema de almacenamiento de datos utilizado para este proyecto fue PostgreSQL, que brinda el almacenamiento y gestión de datos.

Segunda capa: Backend, el framework ocupado para el backend es Express.js ya que facilita el manejo de rutas, controladores y la lógica para el procesamiento de los datos.

Tercera capa: Frontend, El framework utilizado para el frontend es **React.js**, el cual facilitó la gestión de las interfaces de usuario, la interacción con el sistema y el consumo de APIs..

2. Tecnologías utilizadas

Frontend

- **React.js**
- **Axios para consumo de APIs**
- **Bootstrap/Css**

Backend

- **Node.js**

DataBase

- **PostgresSQL**
- **PgAdmin**

3. Modelo de datos (Base de datos)

Tabla usuarios

- **Descripción**: Registra a los usuarios del sistema.
- **Campos**:
    - id: Identificador único (clave primaria).

- cedula: Número de cédula, único por usuario.
- password: Contraseña del usuario.
- full_name: Nombre del usuario.
- role: Rol del usuario, valores posibles (user, admin, etc.).
- active: Indica si el usuario está activo.

- **Relaciones**: Ninguna explícita.

Tabla categorias

- **Descripción**: Categorías de productos.
- **Campos**:
  - id: Identificador único (clave primaria).
  - nombre: Nombre de la categoría.
- **Relaciones**:
  - Relacionada con productos.

- **Tabla productos**

- **Descripción**: Lista de productos disponibles en el sistema.
- **Campos**:
  - id: Identificador único (clave primaria).
  - nombre: Nombre del producto.
  - descripcion: Descripción del producto.
  - precio: Precio del producto.
  - cantidad_disponible: Cantidad disponible en stock.
  - categoria_id: Clave foránea a categorias(id).
  - fecha_creacion: Fecha de creación del producto.

- **Relaciones**:
    - Clave foránea categoria_id que referencia categorias.

Tabla pedidos

- **Descripción**: Registra los pedidos realizados por los clientes.
- **Campos**:
    - id: Identificador único -> clave primaria.
    - fecha: Fecha en que se realizó el pedido.
    - usuario: Nombre del usuario que realizó el pedido.
    - total_compra: Total del pedido.
- **Relaciones**:
    - Relacionada con detalle_pedido.

Tabla detalle_pedido

- **Descripción**: Detalle de cada pedido.
- **Campos**:
    - id: Identificador único -> clave primaria.
    - pedido_id: Clave foránea que referencia pedidos(id).
    - producto_id: Clave foránea que referencia productos(id).
    - cantidad: Cantidad de productos en el pedido.
    - precio: Precio de cada producto en el pedido.
- **Relaciones**:
    - Clave foránea -> pedido_id que referencia pedidos.

- Clave foránea -> producto_id que referencia productos.

Tabla inventarios

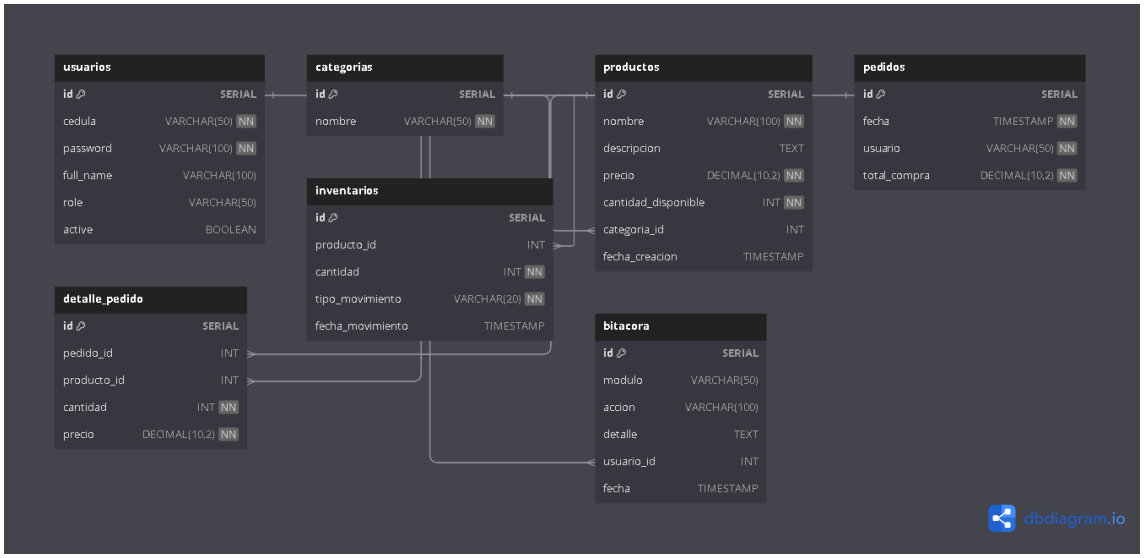
- **Descripción**: Registra movimientos de inventario (entradas y salidas).
- **Campos**:
  - id: Identificador único (clave primaria).
  - producto_id: Clave foránea que referencia productos(id).
  - cantidad: Cantidad involucrada en el movimiento.
  - tipo_movimiento: Indica si es una entrada o salida (Entrada, Salida).
  - fecha_movimiento: Fecha del movimiento.

Tabla bitacora

- **Descripción**: Registra movimientos de usuarios (entradas, salidas, ventas).
- **Campos**:
  - id: Identificador único (clave primaria).
  - modulo: Clave foránea que referencia productos(id).
  - accion: Acción realizada por el usuario.
  - detalle: Detalla la acción.
  - usuario_id: Clave foránea que referencia al usuario.
  - Fecha: Fecha de la acción.

Imagen 1

Modelo de datos – bddiagram.io

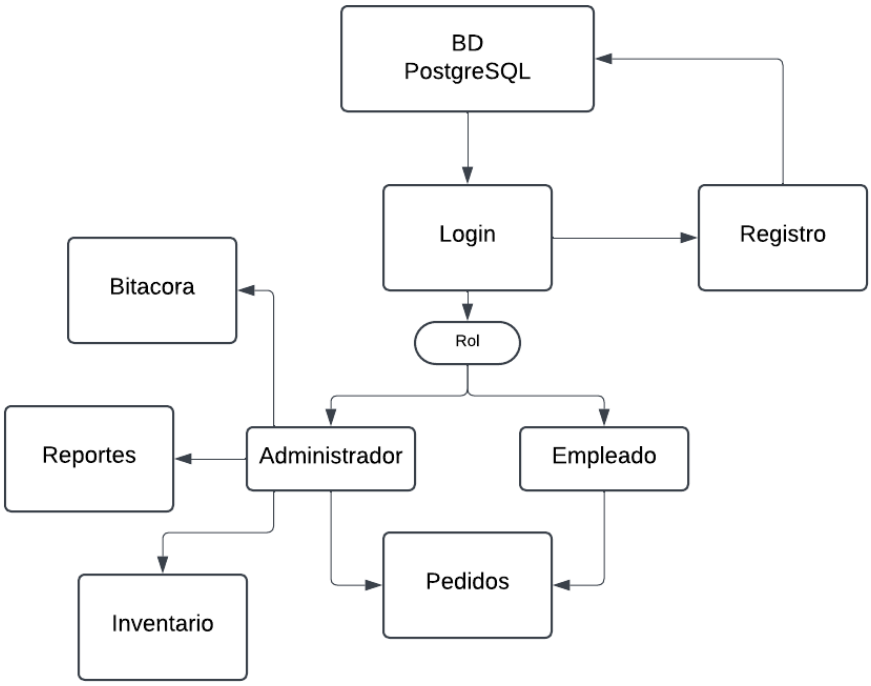


Nota: En esta imagen se muestra la estructura de la base de datos y las relaciones entre tablas.

4. Diagrama de componentes

Imagen 2

Diagrama de componentes - lucidchart



Nota: En esta imagen se detalla las interacciones que existen entre los módulos.

5. Metodología

La metodología utilizada en este proyecto para su organización y desarrollo fue Scrum, debido a sus ventajas como una de las metodologías ágiles más utilizadas. Dado que se trató de un trabajo de larga duración, Scrum permitió estructurar el proceso mediante Sprints, que son períodos cortos en los que se establecen objetivos específicos. En este caso, cada Sprint se organizó por mes, considerando la disponibilidad de los miembros del equipo. A continuación, se presenta una tabla con el resumen de las actividades realizadas en cada etapa.

Tabla 1

Organizador Scrum

Mes	**Semanas**	**Avances**
Mes 1	Septiembre	• Levantamiento de requisitos del sistema
Mes 2	Octubre	• Creación y levantamiento del repositorio en GitHub. • Creación y conexión a la Base de Datos
Mes 3	Noviembre	• Backend terminado • Conexión Backend y frontend
Mes 4	Diciembre	• Mejoras en el frontend • Inicio de la documentación
Mes 5	Enero	• Capitulo 1 y 2 terminados • Adelanto del frontend

Mes 6	Febrero	• Finalización de la documentación (Capítulos 1, 2, 3)

Nota: En esta tabla se describe las acciones realizadas en cada uno de los Sprints.

Capítulo II

Construcción del Sistema

Objetivo del capítulo:

Realizar el análisis de requerimientos y llevarlo hasta la implementación final y operativa del sistema. Además, se detallará el uso de las tecnologías empleadas en el proyecto, así como su diseño y funcionalidad.

Analisis de requerimientos

Parte inicial y fundamental para la organización y futuro desarrollo del proyecto, como punto de inicio se tomó en cuenta la problemática principal del cliente, la cual fue la mala organización y confusión en la toma de órdenes como a su vez la falta de organización en el almacenamiento de sus productos, con esto se derivó a la optimización y automatización en dichos procesos y se planteó el sistema de automatización de pedidos e inventarios, para lo cual desglosamos estos problemas en requisitos funcionales y no funcionales.

Requisitos funcionales

- Gestión de pedidos
- Visualización de menú y pedidos
- Roles y permisos

Requisitos no funcionales

- Rendimiento
- Escalabilidad

- Mantenimiento
- Usabilidad

Cada uno de estos requerimientos son explicados con más detalle en el Capítulo 1 en el apartado de levantamiento de requisitos.

Herramientas y tecnologías usadas:

Iniciaremos con la presentación de las tecnologías y herramientas empleadas, explicando su impacto en el desarrollo del sistema.

Lenguajes de programación.

JavaScript. -

- Lenguaje utilizado tanto en el apartado de backend como el de frontend.
- Dentro del backend se lo implemento para manejar la logica de los servidores y rutas
- Dentro del frontend se lo implemento para crear la interfaz interactiva que vera el usuario

SQL. -

- Lenguaje utilizado para la generación de la base de datos dentro de lo que es postgres junto a la gestión de los datos dentro de esta.

Frameworks y librerías:

Node.js. -

- Nos facilita el entorno para la ejecución de JavaScript en el servidor.
- Ayuda a administrar la conexión del cliente al servidor.

Express.js

- Es el framework de Node.js que nos ayuda con la creacion de rutas y middleware en el backend.
- Este maneja las solicitudes y las respuestas que hay entre el cliente y el servidor.

React.js

- Es la biblioteca utilizada en el frontend para la construcción de la interfaz de usuario y a su vez ayudo a que esta sea dinámica y modular.
- Ayudo a el manejo de componentes de manera reutilizables.

Axios

- Nos ayudó con las solicitudes HTTP desde el frontend al backend
- Va a facilitar la comunicación entre el cliente y el servidor mediante lo que es el API REST.

React – Bootstrap

- Nos proporcionó los componentes base para el diseño siendo estos compatibles con bootstrap.
- Facilitó la creación de las interfaces sin necesidad de saturar el código con CSS.

React-router-dom

- Ayudó con la gestión de la navegación del sistema sin necesidad de recargar la página.

Sweetalert2

- Integración de notificaciones al sistema para facilitar el manejo de esta.

Chart.js

- Genero los gráficos de las estadísticas de ventas y tiene un amplio catálogo como son gráficos de barras, líneas, circulares, etc...

React-toastify

- Maneja las alertas de confirmaciones ayudando a la administración de pedidos.

Bases de datos:

Postgres SQL. -

- Es el sistema de gestión para la base de datos, en donde se almacena y gestiona la información del sistema
- Interactua mediante el backend utilizando consultas en formato SQL ya sea de inserción, de actualización o búsqueda de datos.

Entornos de desarrollo:

Durante el proceso de elaboración de este sistema implementamos herramientas digitales que nos permitieron integrar cada una de las herramientas a su vez de que nos facilitaron el trabajo en conjunto a continuación se detallaran dichas herramientas.

1. **Git/GitHub:** Herramientas para la gestión de versiones del proyecto fundamental en la integración de los miembros del equipo dentro del proyecto, ayudaron a la organización de este.
2. **Visual Studio Code:** Editor de código básico empleado para lo que fue la programación del backend y del frontend.
3. **Postman:** Encargado de las pruebas de las rutas de las APIs y que estas solicitudes HTTP funcionen y se manden de manera correcta.

Capítulo III

Pruebas y Estabilización

Objetivo del capítulo:

Analizar y documentar los resultados de las pruebas sometidas al Backend, asegurando que cumplan los estándares necesarios para el óptimo funcionamiento del sistema dentro del negocio.

Es fundamental considerar que la etapa de pruebas y estabilización desempeña un papel clave en el desarrollo y entrega de un producto de software. En este capítulo, se abordarán dos aspectos principales: el proceso de ejecución de las pruebas y los resultados obtenidos. La metodología aplicada se basa en tres elementos clave: clientes, personal y productos del restaurante, con el objetivo de simular situaciones reales y obtener resultados representativos de su operación diaria

Las pruebas realizadas son de 3 tipos:

- Pruebas funcionales
- Pruebas de rendimiento
- Pruebas de usuario

Pruebas funcionales. -

Destinadas a verificar la funcionalidad de los módulos del sistema, tomando en cuenta como módulos a cada apartado de este como lo son la toma de pedidos, el registro de usuarios, el inventario y demás funciones que cuenta el sistema.

A continuación, se presenta la tabla de casos de pruebas en donde se entra más a detalle acerca de estas funcionalidades.

Tabla 2

Casos de prueba – pruebas funcionales

Caso de prueba	Descripción	Entrada	Resultado Esperado	Resultado Obtenido	Estado
CP-01	Verificar la creación de usuarios	Cédula, nombre completo y contraseña	El sistema genera un usuario en la base de datos con la información proporcionada	El usuario se registra correctamente en la base de datos	Aprobado
CP-02	Verificar el inicio de sesión	Cédula y contraseña	El sistema redirige al usuario dentro del sistema	Se redirige correctamente al sistema	Aprobado
CP-03	Diferenciar entre usuario empleado y usuario admin	Tipo de usuario	El sistema debe validar que el tipo de usuario y mostrar en el sistema las funciones que	Diferencia entre cada usuario y muestra sus respectivas funciones	Aprobado

			tiene cada uno		
CP-04	Verificar la toma de pedidos	Productos y cantidad	El sistema detecta los productos seleccionados y la cantidad de estos	Detecta tanto la cantidad como los productos y muestra el precio total	Aprobado
CP-05	Verificar los reportes de ventas	Ventas/ordenes	El sistema muestra la reportería de ventas mediante tablas y listado	Muestra las ventas realizadas y un apartado con las gráficas de estas	Aprobado
CP-06	Exporta las ventas en formato excel	Ventas/ordenes	El sistema exporta las ventas en un documento excel en el dispositivo local	Exporta el documento con las estadísticas de ventas	Aprobado
CP-07	Búsqueda de ventas por filtros	Dato de venta	El sistema mostrara las o la venta buscada	Muestra la venta requerida según el	Aprobado

			mediante el filtro	filtro colocado	
CP-08	Administración de usuarios (Usuarios admin)	Usuarios	El sistema debe permitir a los usuarios de rol admin modificar a los otros usuarios	Permite la modificar la información y eliminar a los usuarios registrados	Aprobado
CP-09	Búsqueda de usuarios por filtros	Datos del usuario	El sistema debe buscar a los usuarios mediante el filtro de búsqueda	Muestra a los usuarios mediante el filtro aplicado	Aprobado
CP-10	Administración de inventario	Productos	El sistema debe permitir modificar a los productos del inventario	Permite modificar y/o eliminar el producto	Aprobado
CP-11	Búsqueda de productos por filtros	Datos del producto	El sistema debe buscar los productos mediante el filtro aplicado	Muestra el producto según el filtro	Aprobado

Nota: En esta tabla se describe las pruebas funcionales del sistema.

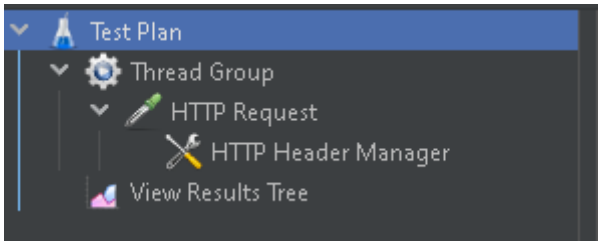
Pruebas de rendimiento. -

Destinadas a medir el rendimiento y capacidad del sistema mediante condiciones específicas, en donde se evalúan parámetros como el tiempo de respuestas, consumo de recursos y más. Para estas pruebas se utilizó la herramienta de Apache JMeter que nos ayudó con la generación de carga y la medición de los tiempos de respuesta, para lo que son pedidos, usuarios y productos.

Para el funcionamiento de Apache Jmeter se realizó una estructura básica que se compone de un Test Plan que contiene un Thread Group, un HTTP Request, un HTTP Header Manager y un View Results Tree.

Imagen 3

Estructura para pruebas en Apache Jmeter

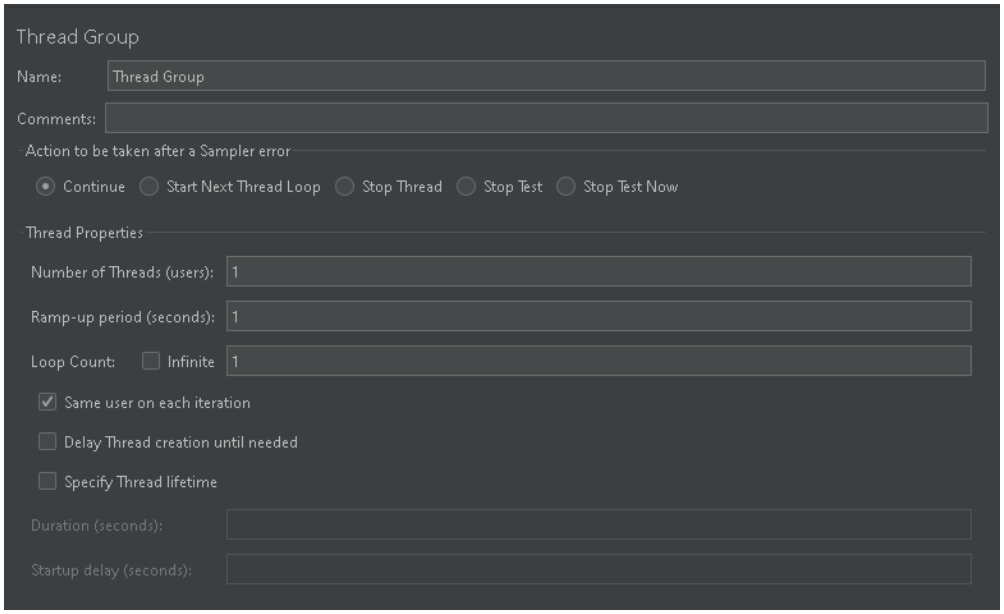


Nota: En esta imagen se muestra la estructura utilizada para las pruebas en Jmeter.

- **Thread Group:** Contenedor principal de Jmeter donde se define como se ejecutará la prueba de carga.

  Imagen 4

  Panel de configuración de Thread Group

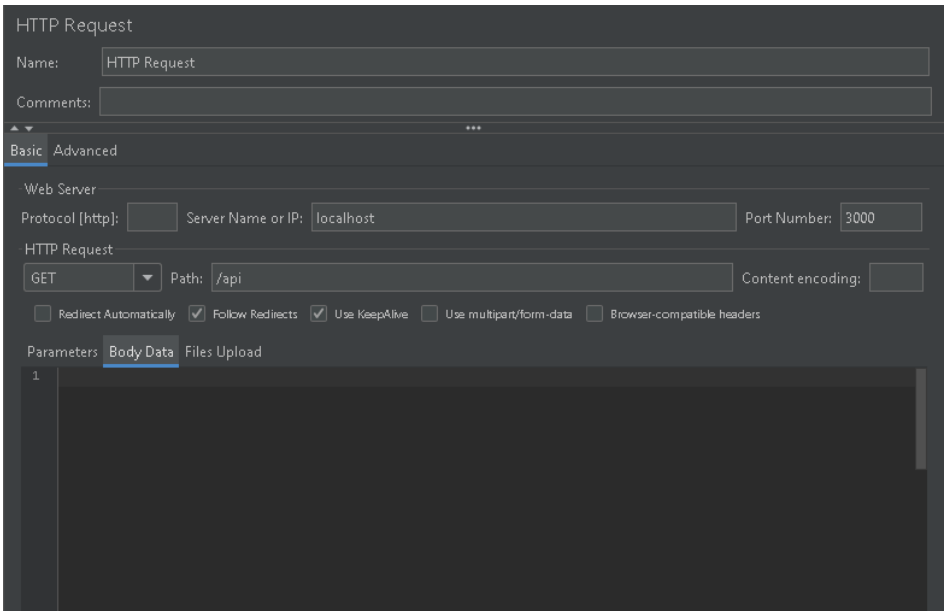


Nota: En esta imagen se detalla la configuración utilizada en Thread Group.

- **HTTP Request:** Es el sampler donde se envían las solicitudes HTTP al Backend, básicamente representa las acciones que se hacen dentro del sistema.

Imagen 5

Panel de configuración HTTP Request

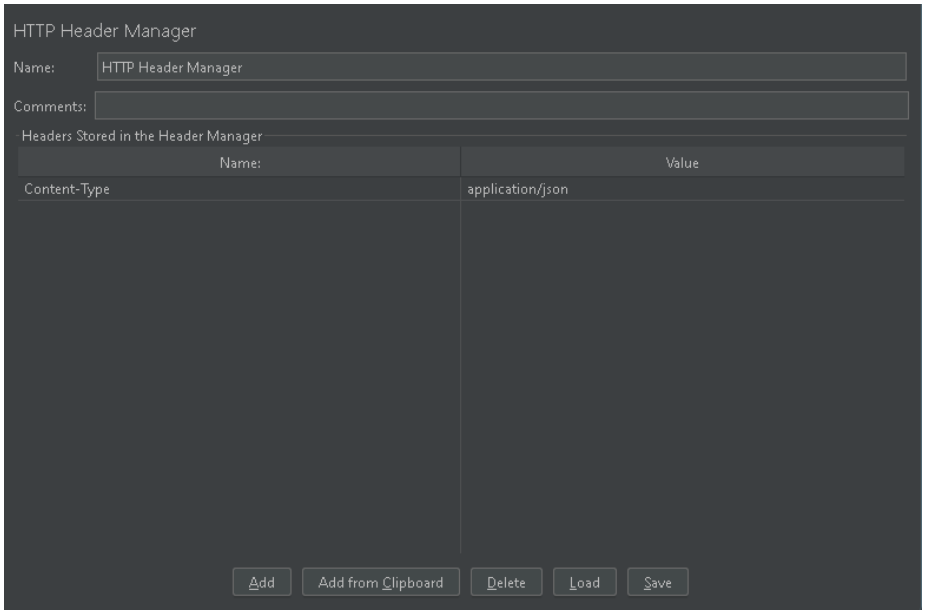


Nota: En esta imagen se detalla la configuración en HTTP Request.

- **HTTP Header Manager**

  Imagen 6

  Panel de configuración HTTP Header Manager

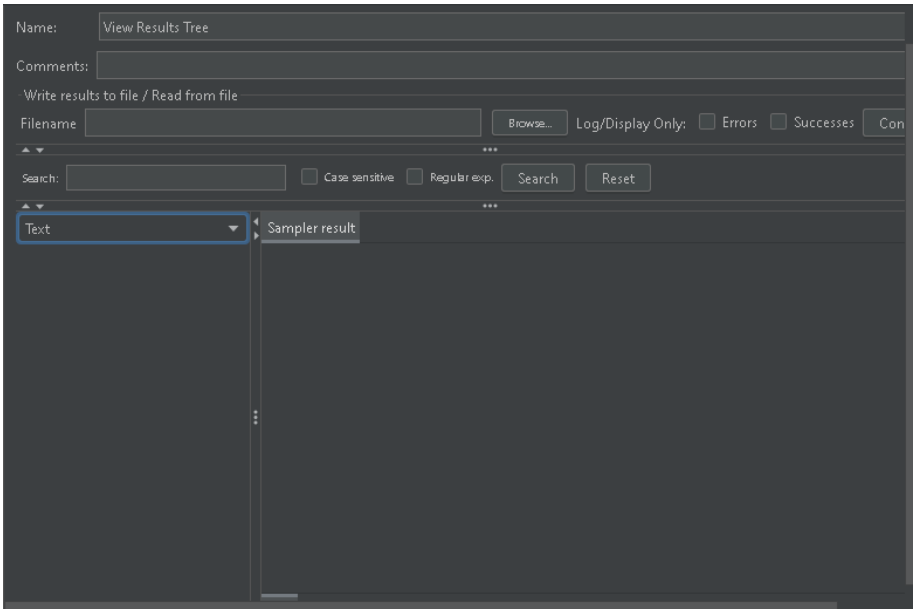


  Nota: En esta imagen se describe la configuración en HTTP Header Manager.

- **View Results Tree**

  Imagen 7

  Panel de configuración View Result Tree



Nota: En esta imagen se describe la configuración en View Result Tree.

Cada una de las tablas presentadas a continuación han sido realizadas mediante esta configuración básica en Apache Jmeter.

Pedidos

Tabla 3

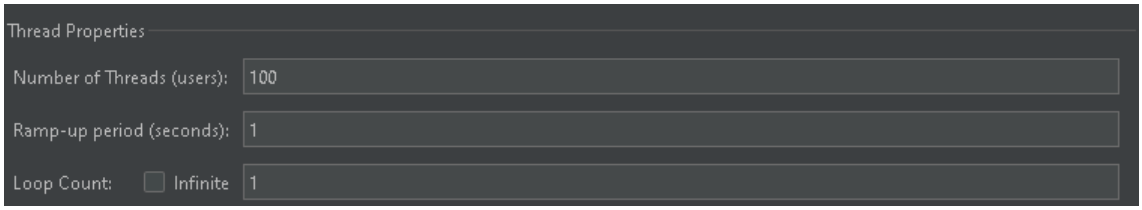
Pruebas de rendimiento GET/POST para pedidos

Método	Descripción	Escenario	Criterio de éxito	Resultado	Estado
POST	**Tiempo de respuesta en hacer pedidos**	**100 pedidos**	**Tiempo < 3 seg**	**2 seg**	**Aprobado**
GET PEDIDOS	**Tiempo de respuesta en mostrar pedidos**	**100 pedidos**	**Tiempo < 7 seg**	**6 seg**	**Aprobado**
GET DETALLES PEDIDOS	**Tiempo de respuesta en mostrar los detalles de los pedidos**	**100 pedidos**	**Tiempo < 10 seg**	**7 seg**	**Aprobado**

Nota: En esta tabla se detalla los resultados de las pruebas GET/POST realizadas con los pedidos.

Imagen 8

Configuración GET/POST para 100 pedidos



Nota: En esta imagen se detalla la configuración para la prueba de 100 pedidos.

Usuarios

Tabla 4

Prueba de rendimiento GET/POST para usuarios

Método	Descripción	Escenario	Criterio de éxito	Resultado	Estado
POST	**Tiempo de respuesta en registrar usuarios**	**10 usuarios**	**Tiempo < 3 seg**	**1 seg**	**Aprobado**
GET	**Tiempo de respuesta en mostrar usuarios**	**10 usuarios**	**Tiempo < 5 seg**	**4 seg**	**Aprobado**

Nota: En esta tabla se detalla los resultados de las pruebas GET/POST realizada para los usuarios.

Imagen 9

Configuración GET/POST para 10 usuarios

Nota: En esta imagen se detalla la configuración para las pruebas GET/POST de 10 usuarios.

Productos

Tabla 5

Pruebas de rendimiento GET/POST para productos

Método	Descripción	Escenario	Criterio de éxito	Resultado	Estado
POST	Tiempo de respuesta en registrar un producto	10 productos	Tiempo < 4 seg	2 seg	Aprobado
GET	Tiempo de respuesta en obtener un producto	10 productos	Tiempo < 5 seg	2 seg	Aprobado

Nota: En esta tabla se detallan los resultados de las pruebas de GET/POST para los productos.

Imagen 10

Configuración GET/POST para 10 productos.

Nota: En esta imagen se muestra la configuración para la prueba de 10 productos.

Pruebas de usuario:

Destinadas a familiarizar a los usuarios con el sistema, para que se pueda evaluar el manejo del funcionamiento de este a través de la experiencia de los usuarios finales que manejaran este, para esta prueba se realizaron diferentes actividades básicas del sistema tanto para el usuario Administrador como para el no administrador.

A continuación, se darán a conocer las actividades realizadas con los usuarios con los manejos de tiempos y observaciones realizadas.

Usuario Empleados

Tabla 6

Prueba de usuario a usuarios empleados

Actividad	Tiempo	Observaciones
Registro	50 segundos	Sencillo, logro llenar de manera correcta las casillas sin dificultades
Login	20 segundos	Sencillo, lleno los campos con la información

		proporcionada en el registro
Toma de pedidos	1 minuto 30 segundo	Intermedio, logro entender la funcionalidad, sin embargo, al momento de añadir cantidad de productos tuvo un poco de confusión

Nota: En esta tabla se detalla los resultados de las pruebas de usuario a los usuarios empleados.

Usuario Administrador

Tabla 7

Pruebas de usuario a usuarios administradores

Actividad	**Tiempo**	**Observaciones**
Registro	40 segundos	Sencillo, logro llenar de manera correcta las casillas sin dificultades
Login	15 segundos	Sencillo, lleno los campos con la información proporcionada en el registro
Toma de pedidos	1 minuto	Intermedio, logro entender la funcionalidad

Administración de usuarios	1 minuto 30 segundos	Intermedio, comprendido las funcionalidades básicas (editar/eliminar) y la búsqueda de empleados
Administración de ventas	2 minutos	Intermedio, entendí las funcionalidades, tuvo complejidad en entender las gráficas, le gusto la funcionalidad de exportar excel.

Nota: En esta tabla se detalla los resultados de las pruebas de usuario a los usuarios administradores.

Conclusiones

- El desarrollo del sistema de automatización de pedidos e inventario mejoró significativamente el servicio del restaurante **"Pollos a la Brasa del Valle"**, cumpliendo con el objetivo principal de optimizar y agilizar la gestión operativa del negocio.

- Las pruebas funcionales demostraron que la implementación del sistema redujo el tiempo entre la toma del pedido y el inicio de su preparación, pasando de un intervalo de hasta **3 minutos** a **1-1:30 minutos**, gracias a su facilidad de uso.

- La plataforma optimiza la gestión administrativa, permitiendo un control más eficiente de empleados y usuarios, además de fortalecer la seguridad mediante un mejor manejo de accesos y perfiles.

- Las pruebas de rendimiento del backend confirmaron que el sistema opera de manera óptima dentro del flujo de pedidos diarios promedio del negocio, garantizando estabilidad y eficiencia.

- La automatización del inventario y las ventas ha optimizado significativamente la administración del restaurante, reduciendo los tiempos

de gestión, los cuales anteriormente se realizaban de forma manual, mejorando así la eficiencia operativa y la toma de decisiones.

Recomendaciones

- Se recomienda proporcionar una capacitación inicial a todo el personal, tanto administrativo como operativo, para garantizar un uso adecuado del sistema, maximizar su aprovechamiento y reducir errores en la operación.

- Es fundamental establecer un plan de mantenimiento periódico que permita supervisar el correcto funcionamiento del sistema, identificar posibles fallos y aplicar las correcciones necesarias a tiempo.

- Para futuras actualizaciones, se sugiere optimizar la gestión de la base de datos mediante la implementación de índices y técnicas de caching, con el fin de mejorar el rendimiento y la velocidad en el procesamiento de datos.

- En cuanto al desarrollo del sistema, se recomienda aplicar buenas prácticas de programación, priorizando la limpieza y estructuración del código para facilitar su comprensión, mantenimiento y escalabilidad en futuras mejoras.

Referencias Bibliográficas

- *Apache JMeter - Apache JMeterTM*. (s. f.). https://jmeter.apache.org/
- *Node.js — Introduction to Node.js*. (s. f.). https://nodejs.org/en/learn/getting-started/introduction-to-nodejs
- *Express - Node.js web application framework*. (s. f.). https://expressjs.com/
- *PostgreSQL*. (2025, 27 enero). PostgreSQL. https://www.postgresql.org/
- *Empezando | Axios Docs*. (s. f.). https://axios-http.com/es/docs/intro
- *Quick start – react*. (s. f.). https://react.dev/learn
- *React Bootstrap | React Bootstrap*. (s. f.). https://react-bootstrap.netlify.app/
- *React Router home*. (s. f.). React Router. https://reactrouter.com/home
- *SweetAlert2*. (s. f.). A Beautiful, Responsive, Customizable And Accessible (WAI-ARIA) Replacement For JavaScript's Popup Boxes. https://sweetalert2.github.io/#usage
- *Chart.js | chart.js*. (s. f.). https://www.chartjs.org/docs/latest/
- *npm: react-toastify*. (s. f.). Npm. https://www.npmjs.com/package/react-toastify
- Atlassian. (s. f.). *Los distintos tipos de pruebas en software | Atlassian*. https://www.atlassian.com/es/continuous-delivery/software-testing/types-of-software-testing

Anexos

1. **Manual de usuario**