



ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
FACULTAD: INFORMÁTICA Y ELECTRÓNICA
ESCUELA DE INGENIERÍA EN SISTEMAS
CARRERA: SOFTWARE

GUÍA DE LABORATORIO DE APLICACIONES INFORMÁTICAS II

PRÁCTICA No. 2

1. DATOS GENERALES:

NOMBRE DEL ESTUDIANTE

Kevin Alexis Rodriguez Segura

CODIGO DEL

7295

FECHA DE REALIZACIÓN:

2026/01/29

FECHA DE ENTREGA:

2026/01/30

2. OBJETIVO(S):

2.1. GENERAL

Establecer un plan de mantenimiento de software que permita garantizar la continuidad operativa, calidad, seguridad y evolución del sistema, mediante la aplicación de procedimientos técnicos estandarizados y buenas prácticas de ingeniería de software.

2.2.ESPECÍFICOS

- Identificar los tipos de mantenimiento aplicables al sistema (correctivo, adaptativo, perfectivo y preventivo).
- Definir un proceso ordenado para la gestión de cambios y corrección de errores.
- Establecer criterios de priorización basados en impacto, severidad y urgencia.
- Documentar las herramientas y recursos necesarios para el mantenimiento.
- Reducir riesgos técnicos y operativos mediante planes de contingencia.

3. METODOLOGÍA

La metodología empleada se basa en un enfoque iterativo e incremental, alineado con buenas prácticas de mantenimiento de software. Se consideran fases de análisis, implementación, pruebas y documentación, asegurando trazabilidad de los cambios y mejora continua del sistema.

4. EQUIPOS Y MATERIALES:

- **Computador:** LAPTOP HP VICTUS 15-FB3019LA
- **Entorno integrado de desarrollo (IDE):** Visual Studio Code
- **Aula virtual**
- **Acceso a internet**
- **Bibliografía**

5. MARCO TEORICO:

Mantenimiento correctivo

El mantenimiento correctivo consiste en detectar y corregir errores, fallos o defectos que ya existen en el software y que afectan su funcionamiento correcto, como resultados incorrectos, pantallas vacías, campos que no validan, respuestas inesperadas en la API o vulnerabilidades de seguridad; su objetivo principal es restablecer el comportamiento esperado del sistema sin cambiar la funcionalidad ni agregar mejoras.

Mantenimiento adaptativo

El mantenimiento adaptativo comprende las modificaciones necesarias para que el software siga funcionando en un entorno cambiado, sin alterar su funcionalidad principal; se utiliza cuando hay nuevas versiones de sistemas operativos, navegadores, frameworks, librerías, motores, APIs externas o cambios en regulaciones (como privacidad, cookies o accesibilidad).

Mantenimiento perfecto

El mantenimiento perfecto incluye las mejoras que no son correcciones de errores ni adaptaciones obligatorias, sino modificaciones orientadas a aumentar el valor del sistema desde el punto de vista de los usuarios y del equipo técnico, como optimizar el rendimiento, mejorar la usabilidad, agregar funciones útiles o elevar la calidad del código.

Mantenimiento preventivo

El mantenimiento preventivo agrupa las actividades que buscan prevenir fallos y dificultades mediante la mejora continua de la estructura del software, sin atender a un error específico ni a un cambio externo inmediato; su objetivo es reducir la deuda técnica, aumentar la mantenibilidad y la estabilidad del sistema a largo plazo.

HTML5 (Lenguaje de marcado de hipertexto, versión 5)

HTML5 es el lenguaje de marcado estándar que se utiliza para definir la estructura y el contenido de una página web en el navegador. Consiste en un conjunto de etiquetas (como <html>, <head>, etc.) que indican el tipo de contenido que representan (títulos, párrafos, imágenes, enlaces, formularios, etc.) y cómo se organizan jerárquicamente. HTML5 introduce nuevas etiquetas semánticas que permiten describir mejor la intención del contenido y funcionalidades modernas como el manejo de multimedia (audio y video), gráficos en 2D/3D, almacenamiento local y APIs para geolocalización, arrastrar y soltar y canva, entre otras. (Alonso, 2023)

CSS (hojas de estilo en cascada)

CSS es un lenguaje de hojas de estilo que define cómo se presentan los elementos de un documento HTML (o XML) en diferentes dispositivos, especialmente en navegadores web. Permite controlar aspectos visuales como colores, tipografía, márgenes, rellenos, posicionamiento, animaciones, transiciones y diseño responsivo. CSS se aplica a través de selectores que identifican elementos HTML (por etiqueta, clase, id, atributo, etc.). Las versiones modernas (CSS3 y posteriores) incluyen módulos avanzados para transformaciones, animaciones, transiciones, flexbox, grid, variables y media queries, lo que permite crear interfaces ricas y adaptables a diversos dispositivos. (IngenioVirtual, 2015)

JSON (Notación de objetos JavaScript)

JSON es un formato de texto ligero y estándar para representar datos estructurados, basado en la sintaxis de objetos literales de JavaScript. Se utiliza principalmente para el intercambio de información entre cliente y servidor en aplicaciones web, como respuestas de APIs REST, configuraciones, almacenamiento de datos estructurados y mensajes asíncronos (AJAX). Su ventaja principal es que es fácil de leer para humanos, sencillo de procesar para las máquinas y ampliamente soportado por lenguajes modernos, lo que lo convierte en el estándar de facto para la comunicación de datos en aplicaciones web y móviles. (Arsys, 2023)

6. PROCEDIMIENTO:

1. Introducción

El presente documento describe de manera formal el proceso de mantenimiento del sistema de software, detallando las actividades realizadas para garantizar su correcto funcionamiento, estabilidad y evolución. Este informe sirve como referencia técnica para la gestión, mejora y control del sistema durante su ciclo de vida operativo.

1.1 Propósito del documento

El propósito de este documento es definir y documentar el proceso de mantenimiento del sistema de gestión de inventarios de TechZone store, estableciendo las actividades, criterios y procedimientos aplicados. Está dirigido a desarrolladores, administradores del sistema y responsables técnicos, con el fin de facilitar la comprensión de los cambios realizados y apoyar futuras intervenciones sobre el software.

1.2 Alcance del mantenimiento

El mantenimiento abarca los módulos funcionales principales del sistema correspondientes, incluyendo funcionalidades de gestión de productos, autenticación, control de stock y visualización para usuarios Cliente y Administrador. No se consideran dentro de este alcance ni los módulos que se encuentren en desuso o módulos de facturación.

1.3 Definiciones, acrónimos y abreviaturas

- IDE (Integrated Development Environment): Entorno Integrado de Desarrollo que proporciona herramientas para la programación, depuración y gestión del código fuente.

- API (Application Programming Interface): Interfaz que permite la comunicación entre el frontend y el backend del sistema.

1.4 Referencias

Para la elaboración de este documento se consideraron las siguientes referencias:

- Manuales técnicos y funcionales del sistema.
- Documentación interna del proyecto.
- Normas y buenas prácticas de desarrollo y mantenimiento de software.
- Acuerdos de nivel de servicio (SLAs), cuando aplique.

2. Descripción General del Sistema

2.1 Resumen del sistema

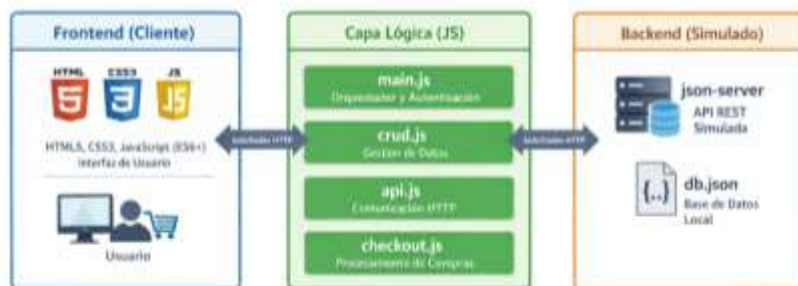
El sistema es una aplicación de gestión de ventas y productos tecnológicos llamado Tech Store, cuyo propósito principal es facilitar el registro, control y administración de productos, usuarios y transacciones de compra. La aplicación permite visualizar productos disponibles, gestionar un carrito de compras y registrar las ventas realizadas, asegurando la integridad de la información almacenada en la base de datos.

El objetivo del software es optimizar el proceso de venta, mejorar la experiencia del usuario y garantizar un control adecuado del inventario, contribuyendo a una gestión eficiente y confiable del negocio.

2.2 Arquitectura del sistema

La arquitectura del sistema se basa en un modelo cliente-servidor, donde la capa de presentación interactúa con el usuario, la capa de lógica de negocio procesa las operaciones y la capa de datos gestiona el almacenamiento de la información.

- Diagrama de arquitectura



La arquitectura cliente-servidor del sistema de inventario de una tienda tecnológica permite que los dispositivos del personal (clientes) se conecten a un servidor central que gestiona la base de datos y la lógica del negocio, procesando consultas y actualizaciones de productos de manera segura y eficiente.

- Componentes principales

Módulos de productos, ventas, usuarios e inventario.
Lógica de validación y control de transacciones.

- Dependencias internas y externas

Internas: Módulos JS (ES Modules).

Externas: json-server (Backend).

2.3 Entornos

Desarrollo

En el entorno de desarrollo, se implementa y modifica todo el código utilizando Visual Studio Code, trabajando con HTML5, CSS3 y JavaScript, mientras que json-server simula la API y db.json funciona como base de datos local.

Pruebas

En el entorno de pruebas, se verifica que todas las funcionalidades operen correctamente, realizando pruebas de integración, validación de la interfaz y flujo de transacciones para asegurar la calidad del sistema antes de su despliegue

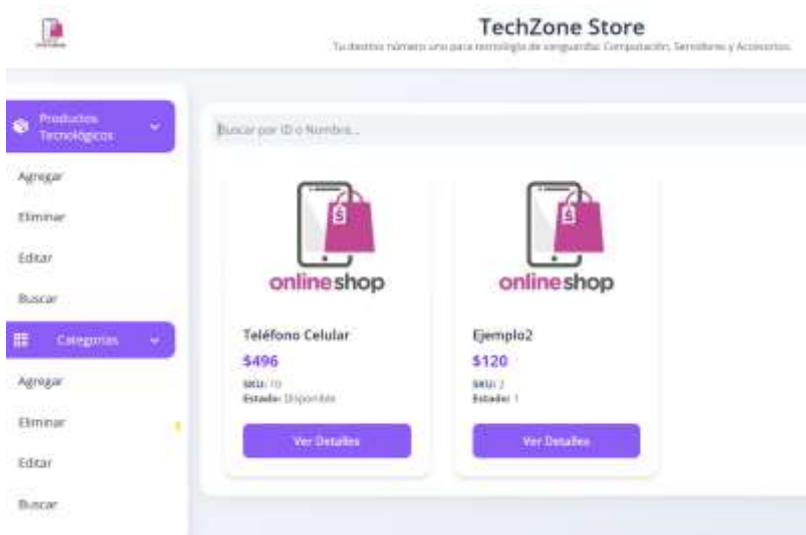
Producción.

En el entorno de producción, puse a disposición la aplicación de inventario para la tienda tecnológica, asegurando que funcionara de manera estable y eficiente para la gestión del inventario y las transacciones en tiempo real. Como único responsable del desarrollo, configuré el modo Cliente para que los usuarios puedan consultar productos, verificar precios y realizar compras de manera ágil y segura, con una interfaz intuitiva que facilita la navegación.

3. Tipos de Mantenimiento

3.1 Mantenimiento correctivo

- 1) El sistema presenta dificultades en las funcionalidades de editar, eliminar y buscar productos, ya que estas acciones requieren que el usuario conozca previamente el ID del producto. Los productos mostrados en pantalla no permiten ser gestionados directamente, lo que limita la interacción y genera confusión en el usuario, afectando la funcionalidad y la usabilidad del sistema.



- 2) Al iniciar la aplicación, no se visualiza un formulario de autenticación que permita el ingreso del nombre de usuario y la contraseña. La inexistencia de este módulo impide el acceso adecuado al sistema y compromete el control de usuarios y la seguridad de la información.



3.2 Mantenimiento adaptativo

- 1) La aplicación no cuenta con un mecanismo de gestión de stock, lo que impide controlar la disponibilidad real de los productos. Como consecuencia, los usuarios pueden adquirir productos sin considerar la cantidad existente, generando inconsistencias en el inventario.



- 2) El sistema no muestra automáticamente los productos al ingresar al modo Cliente, obligando al usuario a presionar el botón “Ver Catálogo” para cargarlos. Esto dificulta la experiencia de navegación, ralentiza la consulta del inventario y puede generar confusión al no visualizarse el catálogo de manera inmediata.



3.3 Mantenimiento perfectivo

- 1) Las opciones de editar, eliminar y buscar se encuentran separadas y mal organizadas dentro de la interfaz, con dos de ellas agrupadas y una aislada, lo que provoca una navegación poco intuitiva. Esta disposición dificulta el uso eficiente del sistema y reduce la claridad en la gestión de los productos.



Las pestañas se mantienen abierta los que hacer que se desplegué mucha información

3.4 Mantenimiento preventivo

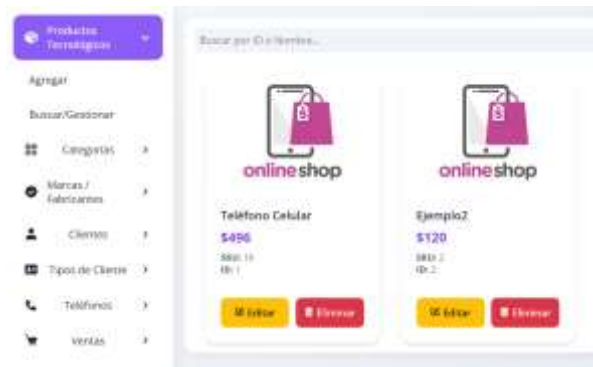
- 1) eliminación de código redundante y aplicación de buenas prácticas de programación.



4. Proceso de Mantenimiento

4.1 Registro de módulos / áreas a intervenir

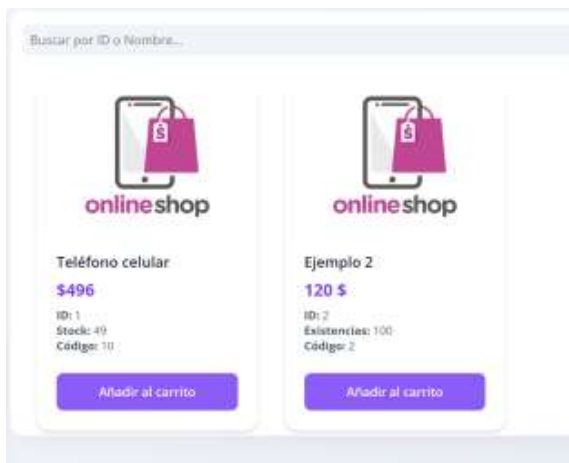
1) funcionalidades de editar, eliminar y buscar productos



2) formulario de autenticación



3) gestión de stock



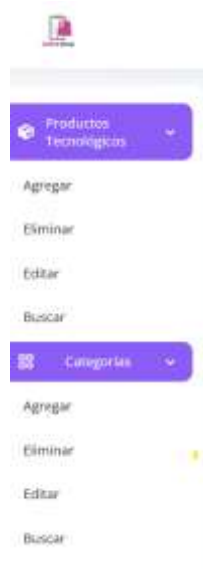
4) Visualización de productos cliente



5) mal organizadas dentro de la interfaz



6) eliminación de código redundante y aplicación de buenas prácticas de programación



4.2 Análisis y priorización

Problema identificado	Severidad	Impacto	Urgencia	Justificación
-----------------------	-----------	---------	----------	---------------

Falta de visualización automática de productos en modo Cliente	Media	Alta	Alta	Afecta directamente la experiencia del usuario, ya que impide visualizar el catálogo sin una acción adicional.
Funcionalidades de editar, eliminar y buscar productos	Alta	Alta	Alta	Son funciones críticas para la gestión del inventario; su mal funcionamiento compromete la operación del sistema.
Gestión de stock	Alta	Alta	Alta	Un error en el control de stock puede generar inconsistencias en el inventario y pérdidas operativas.
Formulario de autenticación	Alta	Media	Alta	Un fallo en la autenticación compromete el acceso seguro al sistema.
Mala organización de elementos en la interfaz	Media	Media	Media	Dificulta la usabilidad, aunque no impide el funcionamiento principal del sistema.
Presencia de código redundante y falta de buenas prácticas	Media	Media	Media	Afecta la mantenibilidad y escalabilidad del sistema a mediano y largo plazo.

Debido a eso se prioriza un mantenimiento programado de con la siguiente estructura.

1. **Gestión de stock**

Es la primera prioridad, debido a su alta severidad, impacto y urgencia. Un control inadecuado del stock puede generar inconsistencias en el inventario y afectar directamente la operación de la tienda.

2. **Funcionalidades de editar, eliminar y buscar productos**

Constituyen funciones esenciales para la administración del inventario. Su correcta implementación es crítica para garantizar la gestión eficiente de los productos.

3. **Formulario de autenticación**

Se prioriza en tercer lugar, ya que garantiza el acceso seguro al sistema y la correcta diferenciación entre los roles de Cliente y Administrador.

4. **Visualización automática de productos en modo Cliente**

Aunque no impide el funcionamiento del sistema, tiene un alto impacto en la experiencia del usuario, por lo que se aborda después de asegurar las funcionalidades críticas.

5. **Organización de los elementos en la interfaz**

Se atiende posteriormente, ya que afecta principalmente la usabilidad y claridad visual del sistema.

6. **Eliminación de código redundante y aplicación de buenas prácticas**

Se aborda como una mejora estructural final, orientada a optimizar el mantenimiento, la escalabilidad y la calidad del código a largo plazo.

4.3 Implementación de cambios

Durante la mejora del sistema de inventario, implementé diversas modificaciones y refactorizaciones para optimizar su funcionalidad y experiencia de usuario. Las principales áreas de intervención fueron:

1. Funcionalidades de productos se Incorporó y optimicé las acciones de editar, eliminar y buscar productos, garantizando que los usuarios puedan gestionar el inventario de manera eficiente.
2. Formulario de autenticación Ajustes el formulario para reforzar la validación de credenciales y asegurar un acceso seguro al sistema.
3. Gestión de stock se buscó mejorar la actualización y control del inventario, permitiendo registrar entradas y salidas de productos de manera confiable.
4. Visualización de productos en modo Cliente del catálogo para que los productos se muestren de manera más accesible, mejorando la experiencia de navegación.
5. Organización de la interfaz de elementos visuales que estaban mal distribuidos, logrando un diseño más intuitivo y uniforme.
6. Refactorización de código donde se eliminó código redundante y apliqué buenas prácticas de programación, como modularización, nomenclatura consistente y comentarios descriptivos, mejorando la legibilidad y mantenibilidad del proyecto.

4.4 Pruebas

- Funcionales: Verificación de que todas las funcionalidades (editar, eliminar, buscar, gestión de stock y autenticación) operen correctamente.
- De interfaz: Comprobación de que los elementos estén organizados correctamente y sean intuitivos para el usuario.
- De integración: Asegurando que la comunicación entre frontend y backend simulado (json-server) se realice sin errores.

4.5 Despliegue

El despliegue del sistema modificado se realizó en local, utilizando Visual Studio Code y json-server para simular la API. Se verificó que todos los cambios funcionaran correctamente, incluyendo la visualización de productos, gestión de stock y autenticación.

5. Cierre y documentación

Se encuneta en el anexo A

6. Herramientas de mantenimiento

Durante el proceso de mantenimiento del sistema utilicé diversas herramientas que facilitaron la implementación, verificación y control de los cambios realizados. **Visual Studio Code** fue empleado como entorno de desarrollo principal, permitiendo la edición, refactorización y depuración del código fuente. **json-server** se utilizó para simular el backend mediante una API REST, lo que permitió realizar pruebas controladas sin necesidad de un servidor real. Adicionalmente, se emplearon herramientas del navegador para pruebas de interfaz y validación de peticiones HTTP. El uso de estas herramientas se justifica por su ligereza, facilidad de uso y compatibilidad con proyectos web basados en JavaScript.

7. Riesgos y Planes de Contingencia

7.1 Identificación de riesgos

Durante el desarrollo y mantenimiento del sistema se identificaron los siguientes riesgos:

- **Riesgos técnicos:** errores en la comunicación entre el frontend y la API simulada, fallos en la gestión del stock o en la autenticación de usuarios.

- **Riesgos operativos:** pérdida de información por fallos en el archivo db.json o interrupciones durante el uso del sistema.
- **Riesgos humanos:** errores involuntarios en la manipulación del inventario por parte del administrador o uso incorrecto del sistema por parte de los usuarios.

7.2 Planes de mitigación

Para reducir el impacto de los riesgos identificados, implementé diversas medidas de contingencia. Se realizaron copias de seguridad periódicas del archivo db.json para evitar pérdida de información. Asimismo, se realizaron copias del sistema en el repositorio de git con la finalidad de no perder los cambios realizados.

8. Anexos

- Diagramas
- Plantillas

7. CONCLUSIONES Y RECOMENDACIONES:

Conclusión

Como resultado del proceso de mantenimiento y mejora del sistema de inventario, se logró optimizar su funcionalidad, organización interna y experiencia de usuario. La aplicación de criterios de priorización permitió atender primero los problemas críticos, garantizando la correcta gestión del stock, la seguridad en la autenticación y el adecuado manejo de productos. Además, la refactorización del código y la aplicación de buenas prácticas fortalecieron la mantenibilidad y escalabilidad del sistema, asegurando su correcto funcionamiento en el entorno de producción.

Recomendación

Se recomienda continuar con la mejora progresiva del sistema, incorporando un backend real y una base de datos robusta para entornos de producción más exigentes. Asimismo, se sugiere implementar pruebas automatizadas y mecanismos de monitoreo que permitan detectar fallos de manera temprana. Y por último actualizar la documentación esto facilitará futuras modificaciones y garantizará la sostenibilidad del sistema a largo plazo.

8. BIBLIOGRAFÍA:

Los 4 tipos de mantenimiento de software: ¿qué es el mantenimiento de software? (2025). Thales Cloud Security Products. <https://cpl.thalesgroup.com/es/software-monetization/four-types-of-software-maintenance>

Alonso, S. (2023, December 12). *Entendiendo el lenguaje HTML - El blog de dinahosting*. El Blog de Dinahosting. <https://dinahosting.com/blog/entendiendo-la-base-del-desarrollo-web-html-css-y-javascript/>

IngenioVirtual. (2015, April 21). *Conceptos básicos sobre tecnologías de desarrollo web - ingeniovirtual.com*. Ingeniovirtual.com. <https://www.ingeniovirtual.com/conceptos-basicos-sobre-tecnologias-de-desarrollo-web/>

Arsys. (2023, August 2). *Arsys*. Arsys. <https://www.arsys.es/blog/archivo-json-que-es-y-para-que-sirve>

Link del repositorio Git

<https://github.com/kevinalexsRod/Mantenimiento.git>

Anexo A