

UNIVERSIDAD MARIANO GALVEZ DE GUATEMALA
FACULTAD DE INGENIERÍA
INGENIERÍA EN SISTEMAS SEDE EL NARANJO
DESARROLLO WEB



ANA LAURA MERCEDES SANTIZO DOMÍNGUEZ **9490-20-12280**

ELVIS ESTUARDO DE LEÓN MORALES **9490-20-1768**

MANUAL TÉCNICO

SISTEMA DE COMERCIO ELECTRÓNICO

Desarrollado por:

ANA LAURA MERCEDES SANTIZO DOMÍNGUEZ

ELVIS ESTUARDO DE LEÓN MORALES

CONTENIDO

Introducción.....	4
Requisitos del Sistema.....	4
2.2. Requisitos de Software.....	5
Arquitectura del Sistema	6
3.1. Arquitectura de la Aplicación	6
Cliente (Frontend):.....	6
Servidor (Backend):	7
3.2. Estructura de Directorios.....	7
Tecnologías Utilizadas.....	8
Esquema Lógico.....	9
Módulo de Registro de Usuarios	9
Documentación Adicional	12

Introducción

Este manual es una guía exhaustiva que proporciona a los desarrolladores, ingenieros y equipos técnicos la información necesaria para entender, implementar, mantener y mejorar la plataforma de comercio electrónico.

El proyecto tiene como objetivo crear una tienda en línea altamente funcional, utilizando React, SASS/SCSS y TypeScript, lo que garantiza un desarrollo eficiente y de alta calidad. A lo largo de este manual, se detallan los diferentes módulos del sistema, desde el registro de usuarios hasta la gestión de productos y compras, cubriendo cada aspecto clave de la aplicación.

Este manual servirá como un recurso esencial para el equipo de desarrollo, permitiéndoles comprender la arquitectura del sistema, las tecnologías utilizadas, las restricciones de seguridad y las mejores prácticas de implementación. También se proporcionan ejemplos concretos, descripciones detalladas y pautas para abordar posibles desafíos durante el desarrollo y el mantenimiento.

La documentación técnica desempeña un papel fundamental en el éxito del proyecto, al permitir una colaboración efectiva, la resolución de problemas y la evolución continua del sistema.

Requisitos del Sistema

2.1. Requisitos de Hardware

- **Servidor de Aplicaciones:** Se recomienda un servidor de aplicaciones con al menos 4 GB de RAM y un procesador de 2 núcleos o superior. Esto asegurará un rendimiento óptimo, especialmente en situaciones de carga elevada.

- **Almacenamiento:** Se requiere suficiente espacio de almacenamiento para alojar la base de datos y los archivos estáticos, como imágenes de productos. Se recomienda un mínimo de 20 GB de almacenamiento.
- **Conexión a Internet:** Se necesita una conexión a Internet estable para garantizar que la plataforma esté disponible en línea y para permitir las transacciones seguras.
- **Sistema de Respaldo:** Es fundamental contar con un sistema de respaldo regular para garantizar la integridad de los datos y la capacidad de recuperación en caso de fallos del sistema.

2.2. Requisitos de Software

- **Servidor Web**
- **Node.js:** Node.js es necesario para ejecutar la aplicación React en el servidor. Asegúrese de tener la última versión de Node.js instalada.
- **Gestor de Paquetes:** Utilice npm (Node Package Manager) para administrar las dependencias de Node.js y las bibliotecas utilizadas en el proyecto.
- **Base de Datos:** Mongo DB
- **Entorno de Desarrollo:** Se recomienda un entorno de desarrollo que admita React, TypeScript y SASS/SCSS. Puede utilizar Visual Studio Code.
- **Control de Versiones:** Se recomienda el uso de un sistema de control de versiones como Git para rastrear y gestionar el código fuente del proyecto.
- **Herramientas de Despliegue:** Para implementar la aplicación en servicios en la nube, se necesitarán herramientas Google Cloud.

Arquitectura del Sistema

3.1. Arquitectura de la Aplicación

Cliente (Frontend):

- **React:** La parte frontal de la aplicación se desarrolla utilizando React, una biblioteca de JavaScript para construir interfaces de usuario interactivas. React proporciona un enfoque de componentes reutilizables que facilita la construcción de páginas web dinámicas y eficientes.
- **SASS/SCSS:** Para el estilo y la presentación de la interfaz de usuario, se utiliza SASS/SCSS. Estos preprocesadores CSS permiten una organización más efectiva del código CSS, lo que facilita la creación de interfaces atractivas y personalizables.
- **TypeScript:** Se utiliza TypeScript para mejorar la calidad del código y reducir errores. TypeScript agrega tipado estático a JavaScript, lo que hace que el desarrollo sea más seguro y comprensible.
- **Contextos de React:** Para administrar el estado de la aplicación, React Context es utilizado. Los contextos permiten compartir datos y funcionalidades en toda la aplicación sin necesidad de pasar propiedades manualmente entre componentes.
- **Enrutamiento:** React Router se utiliza para la navegación entre páginas y la gestión de rutas en la aplicación.

Servidor (Backend):

- Node.js: Node.js se utiliza como entorno de tiempo de ejecución del servidor. Es ideal para aplicaciones en tiempo real y permite la ejecución de JavaScript en el servidor.
- Express.js: Express.js es un marco web de Node.js que simplifica la creación de aplicaciones web y la gestión de rutas.
- Base de Datos: Mongo DB
- Autenticación y Autorización: Se implementa un sistema de autenticación y autorización para garantizar la seguridad de la aplicación. Esto puede incluir la generación y validación de tokens de acceso.
- API RESTful: La lógica de negocios se expone a través de una API RESTful que permite a la aplicación cliente interactuar con el servidor para realizar acciones como registro de usuarios, inicio de sesión, gestión de productos, carrito de compras y compras seguras.

3.2. Estructura de Directorios

Name	Last commit message
 .idea	added all
 src/app	finished all requirements for this project
 .eslintrc.json	added all
 .gitignore	added all
 README.md	Initial commit
 next.config.js	added all
 package-lock.json	added catalog, cart and dashboard
 package.json	added catalog, cart and dashboard
 postcss.config.js	added all
 tailwind.config.ts	added all
 tsconfig.json	added all

 AddProduct.js	finished all requirements for this project
 CartDialog.js	finished all requirements for this project
 Catalog.js	finished all requirements for this project
 Constants.js	finished all requirements for this project
 Login.js	finished all requirements for this project
 ProfileEditDialog.js	finished all requirements for this project
 RolesDialog.js	finished all requirements for this project
 dashboard.js	finished all requirements for this project

Tecnologías Utilizadas

4.1. React

4.2. SASS/SCSS

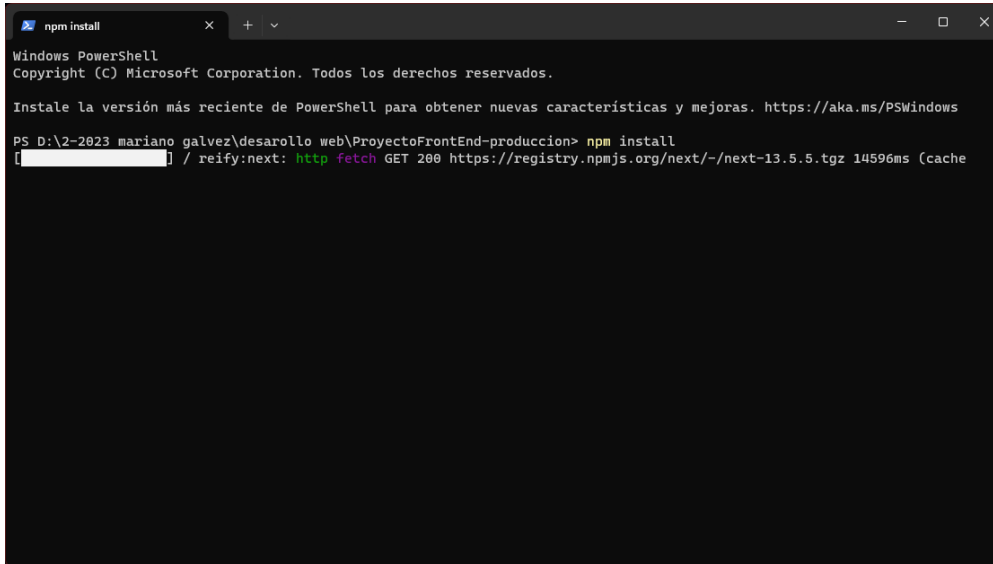
4.3. TypeScript

4.4. Otros

Ejecución de repositorio

Abrir una terminal dentro la carpeta donde se encuentre la aplicación y colocar

- Npm install



```
npm install
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS D:\2-2023 mariano galvez\desarollo web\ProyectoFrontEnd-produccion> npm install
[ ] / reify:next: http fetch GET 200 https://registry.npmjs.org/next/-/next-13.5.5.tgz 14596ms (cache
```

A partir de acá se realizó la instalación de las dependencias necesarias para su ejecución realizada en git lab.

Esquema Lógico

Módulo de Registro de Usuarios

- 5.1. Flujo de Registro
- 5.2. Validación de Correo Electrónico
- 5.3. Validación de NIT, Correo y DPI Únicos
- 5.4. Requisitos de Contraseña Segura
- 5.5. Validación de Campos del Formulario

Módulo de Login

- 6.1. Proceso de Autenticación
- 6.2. Uso de Contextos
- 6.3. Validación de Campos de Inicio de Sesión

Módulo de Gestión de Perfil

- 7.1. Validación del Token
- 7.2. Actualización de Campos de Perfil
- 7.3. Validación de Duplicación de Datos
- 7.4. Validación de Formato de Correo
- 7.5. Eliminación de Usuario e Invalidez del Token

Módulo Catálogo de Productos

- 8.1. Visualización Detallada de Productos
- 8.2. Añadir al Carrito de Compra
- 8.3. Validación de Token
- 8.4. Filtrado por Tipo de Categoría

Módulo de Gestión de Producto (Administrador)

- 9.1. Devolver Toda la Información del Producto
- 9.2. Validación del Token y Roles
- 9.3. Cambio de Estado al Eliminar
- 9.4. Validación de Campos Obligatorios

9.5. Múltiples Categorías por Producto

9.6. Gestión de Stock

Módulo de Carrito de Compra

10.1. Visualización del Carrito de Compra

10.2. Eliminación de Productos del Carrito

10.3. Cambio de Estado de Productos

10.4. Validación del Usuario

Módulo de Compra

11.1. Validación del Usuario

11.2. Descontar del Inventario

Despliegue del Proyecto

12.1. Preparación para el Despliegue

12.2. Implementación en Servicios en la Nube

Seguridad

13.1. Protección de Datos de Usuario

13.2. Seguridad en las Transacciones

Mantenimiento y Actualizaciones

14.1. Resolución de Problemas

14.2. Actualización de Dependencias

14.3. Copias de Seguridad

Documentación Adicional

Todo nuestro front necesario para su utilización lo puede bajar y acomodar a sus consumos de productos de API.

BackEnd

Todo está publicado en Google cloud y esta almacenado en gitlab con un enlace y esto manda la publicación del proyecto a Google cloud al momento de darle el npm install en gitlab.

FrontEnd

Esta almacenado en gitlab el repositorio y de ahí se configura el node y se hace el npm install y lo que construye se publica en Google cloud.

Enlace del repositorio de FrontEnd

<https://github.com/anniesantizo/ProyectoFrontEnd>

Proyecto completo

<https://gitlab-51661526-produccion-l7i4drwjxa-de.a.run.app/>