UNIVERSIDAD MARIANO GÁLVEZ DE GUATEMALA

FACULTAD DE INGENIERÍA EN SISTEMAS

DESARROLLO WEB

ING. JOSÉ MIGUEL VILLATORO HIDALGO

MANUAL TÉCNICO

PROYECTO 1

KELVIN JOSÉ GÓMEZ MORALES 9490-19-480

LESTER HAROLDO BLANCO MELENDRES 9490-19-5517

GUATEMALA, 11 DE SEPTIEMBRE DE 2023

Tabla de Contenidos

Introducción

Nuestro sistema de comercio electrónico abarca una gama de características fundamentales que son esenciales para brindar una experiencia de compra en línea efectiva y satisfactoria. Estas características incluyen un sistema de registro y autenticación de usuarios, una plataforma de inicio de sesión segura, una lista de productos completa y detallada, un carrito de compras intuitivo y un apartado de ventas que permite a los comerciantes administrar eficientemente las transacciones.

A lo largo de este manual, exploraremos en detalle cada uno de estos componentes, proporcionando información técnica esencial, esperamos que este manual sea de gran ayuda para entender, utilizar y evaluar el sistema creado.

Contenido

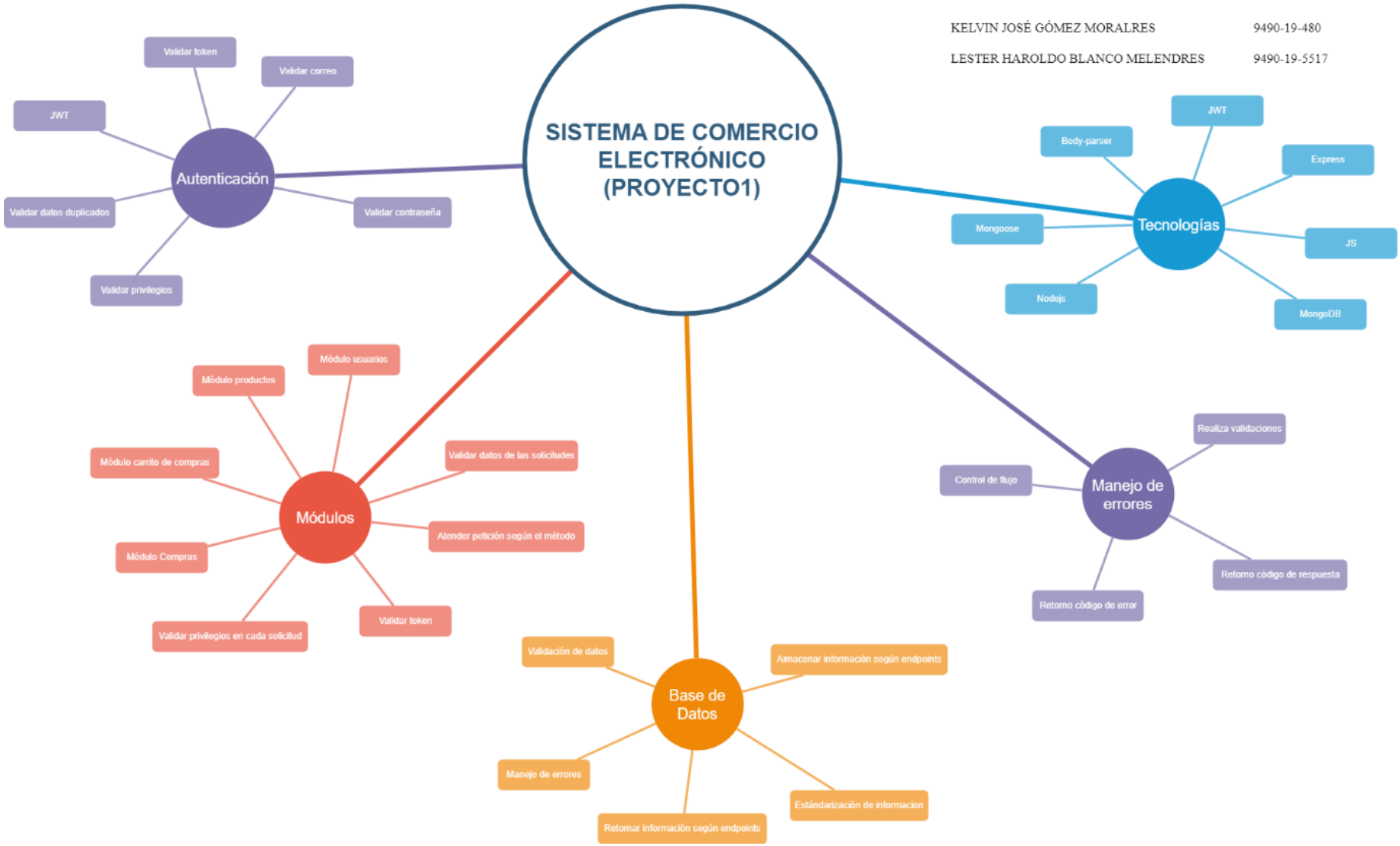
Esquemas

Esquema conceptual

En el siguiente esquema encontrará una representación visual y descriptiva que proporciona una vista general y estructurada de los aspectos fundamentales del proyecto, lo que facilita la comprensión de su arquitectura y funcionamiento sin entrar en detalles técnicos exhaustivos. En este esquema, se abordan cuatro aspectos clave: autenticación, módulos, manejo de errores y tecnologías del proyecto.

Este esquema conceptual proporciona una visión general esencial, está enfocado en la simplicidad y facilidad de entendimiento para cualquier persona. Permite comprender rápidamente los aspectos clave del proyecto, facilitando la colaboración y el mantenimiento a lo largo del ciclo de vida del proyecto.

En resumen, un esquema conceptual sirve como una guía visual que permite a todos los interesados en el proyecto tener una comprensión clara de los aspectos clave sin sumergirse en los detalles técnicos. Facilita la comunicación, la colaboración y el mantenimiento, lo que contribuye a la eficiencia y el éxito general del proyecto.



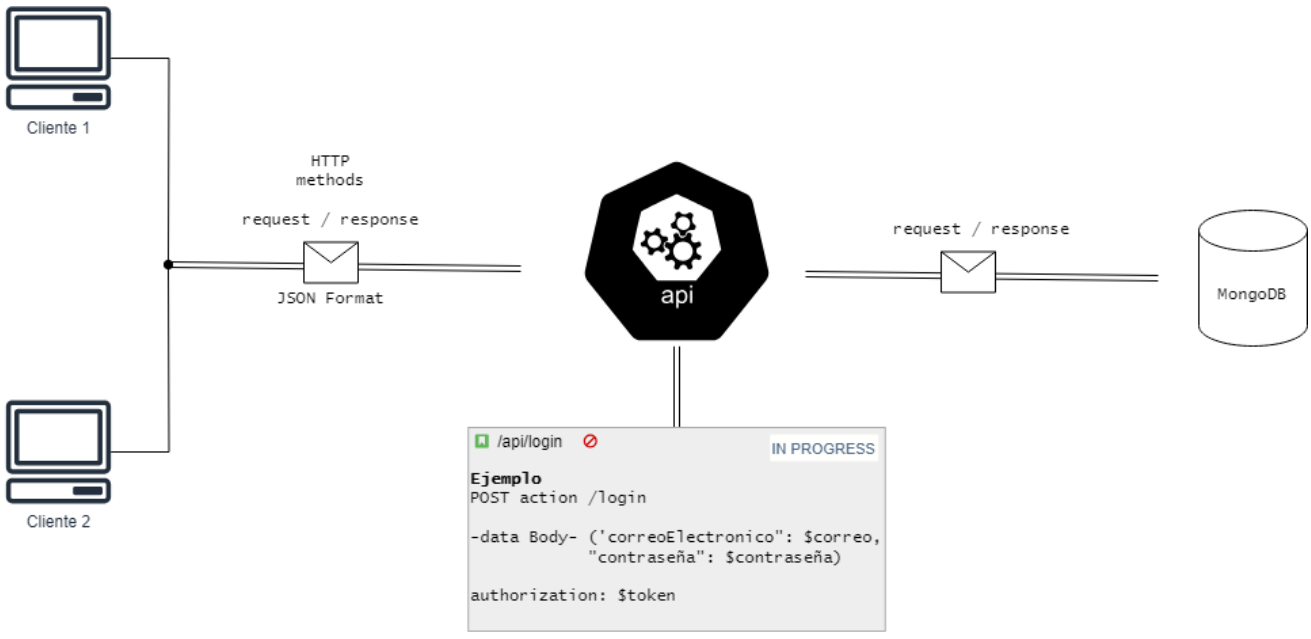
Creadores:

KELVIN JOSÉ GÓMEZ MORALRES	9490-19-480
LESTER HAROLDO BLANCO MELENDRES	9490-19-5517

Esquema lógico

En este esquema se busca representar de manera visual, el flujo de interacción entre dos clientes y una API que realiza procesos como la validación de credenciales, y se comunica con una base de datos MongoDB para validar estas credenciales, proporciona una visión general de alto nivel del proceso de autenticación en el sistema. A continuación, se presenta una descripción de este esquema

Aunque el alcance del proyecto es mucho mayor, un ejemplo sencillo como este queda bastante entendible y fácil de aprender.

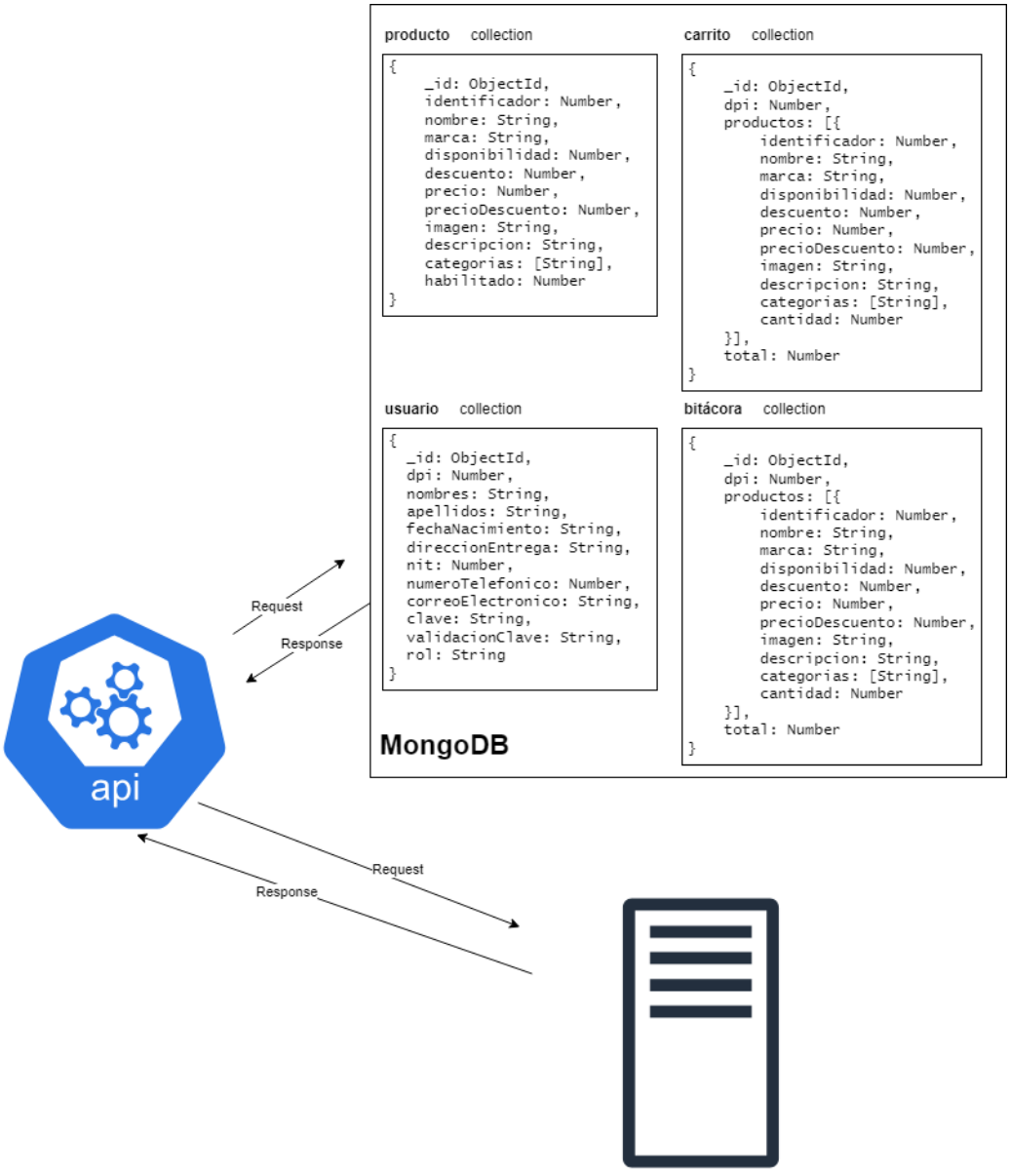


Creadores:

KELVIN JOSÉ GÓMEZ MORALRES	9490-19-480
LESTER HAROLDO BLANCO MELENDRES	9490-19-5517

Esquema físico

Este último esquema también representa de forma visual,la interacción entre una API, una base de datos MongoDB y un servidor muestra cómo estos componentes interactúan entre sí en un nivel técnico y concreto.



Creadores:

KELVIN JOSÉ GÓMEZ MORALRES 9490-19-480

LESTER HAROLDO BLANCO MELENDRES 9490-19-5517

Descripciones

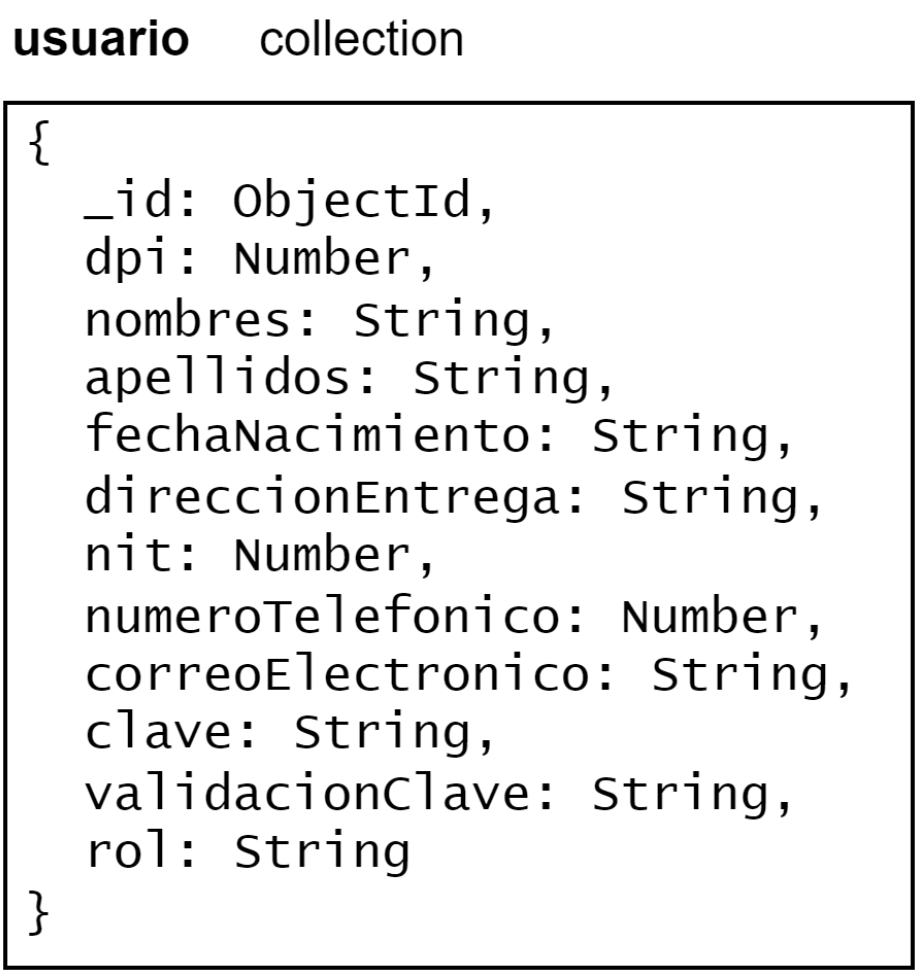
Descripción de tablas (colecciones)

Usuario

Esta tabla almacenará todos los datos de los usuarios que se registren.

Un campo muy importante para validar los privilegios y permisos que tendrá cierto usuario en concreto, será el de rol, el cuál contendrá “admin” o “usuario”.

Los campos que se requieren y sus tipos están descritos en la siguiente imagen.

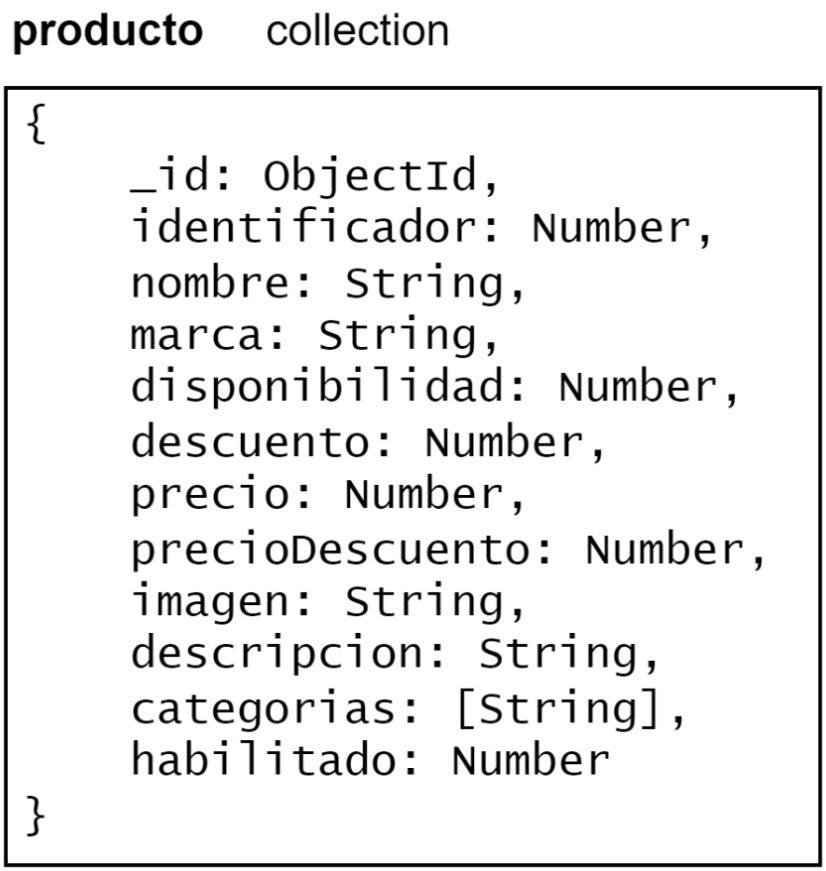


Producto

Los productos que los administradores puedan añadir se almacenarán en esta tabla. Cualquier persona con un token válido podrá visualizarlos y obviamente, solo los administradores podrán agregar, modificar y eliminar. Eliminar un producto lo hace de forma lógica es decir de 1 a 0 (habilitado = 1, deshabilitado = 0).

El campo **precioDescuento** se calculará a través de del precio y el descuento que se hará. El campo **categorías** contendrá un arreglo con las diferentes categorías a las que pertenece este producto.

Los campos que se requieren y sus tipos están descritos en la siguiente imagen.



Carrito

En esta tabla se irán agregando los diferentes productos que el usuario decida tener en su carrito de compras. Habrá un solo documento por cliente y acá es donde se podrá ver el carrito de compras que tiene actualmente y el total de este se irá actualizando en función si se agregan o eliminan productos.

El campo **productos**, almacenará todos los productos y sus detalles para luego si se realiza una compra, hacer el respectivo descuento en disponibilidad, cobro del total calculado.

Los campos que se requieren y sus tipos están descritos en la siguiente imagen.

carrito collection

```
{
    _id: ObjectId,
    dpi: Number,
    productos: [{
        identificador: Number,
        nombre: String,
        marca: String,
        disponibilidad: Number,
        descuento: Number,
        precio: Number,
        precioDescuento: Number,
        imagen: String,
        descripcion: String,
        categorias: [String],
        cantidad: Number
    }],
    total: Number
}
```

Bitácora

Esta tabla almacenará las compras que hayan realizado los usuarios, existe un documento por cada cliente para ir agregando las compras que haga a lo largo de su uso en el sistema, contiene el mismo esquema que el carrito de compras por lo qué funciona de la misma manear a nivel de privilegios y permisos.

Los campos que se requieren y sus tipos están descritos en la siguiente imagen.

bitácora collection

```
{
    _id: ObjectId,
    dpi: Number,
    productos: [{
        identificador: Number,
        nombre: String,
        marca: String,
        disponibilidad: Number,
        descuento: Number,
        precio: Number,
        precioDescuento: Number,
        imagen: String,
        descripcion: String,
        categorias: [String],
        cantidad: Number
    }],
    total: Number
}
```

Descripción de la API

La API creada para este sistema de comercio electrónico permite a las personas crear un nuevo usuario, ingresar al sistema con ese usuario creado, ver su perfil de usuario, actualizar sus datos de dicho perfil o eliminar su perfil por completo, también incluye a estos usuarios normales la posibilidad de ver los diferentes productos disponibles y todos sus detalles, elegir un producto y agregarle cierta cantidad para agregarlo a un carrito de compras y posteriormente confirmar dicha compra, esto habilitará la opción de que puedan consultar su historial de pedidos.

En la parte de los administradores, cuentan con la opción de crear nuevos productos, actualizarlos o eliminarlos (nivel lógico). Adicionalmente, cuenta con todos los permisos que tiene los usuarios normales, haciendo que puedan realizar pruebas y ajustes según sea necesario.

La ruta inicial y donde se puede acceder a la API es: http://localhost:3000/api

Descripción de Endpoints

Administradores

Crear nuevos productos

Ruta: http://localhost:3000/api/producto

Método: POST

Esta funcionalidad permite a los usuarios que tengan como rol administrador, crear nuevos productos y que aparezcan en el listado general de estos. El campo que no deben ingresar es *precioDescuento*: ya que este se calcula.

Este Endpoint valida si los campos ingresados **no son campos vacíos**.

Ejemplo de Body en formato JSON:


```
{
  "identificador": 2,
  "nombre": "Televisor",
  "marca": "Samsung",
  "disponibilidad": 10,
  "descuento": 500,
  "precio": 5000,
  "precioDescuento": 4500,
  "imagen": "./img/tv-samsung.png",
  "descripcion": "La brillante calidad de imagen 4K que completan 20 redes neuronales diferentes te garantiza la potencia de 4K.",
  "categorias": [
    "Televisores",
    "Hogar",
    "Electrónicos"
  ],
  "habilitado": 1
}
```


Actualizar productos

Ruta: http://localhost:3000/api/producto/:identificador

Método: PATCH

Esta funcionalidad permite a los usuarios que tengan como rol administrador, actualizar los datos de los productos existentes mediante el identificador de este y que aparezcan en el listado general de estos con los nuevos datos. Campos que no se pueden cambiar: *identificador*, *precioDescuento.*

Este Endpoint valida si los campos ingresados **no son campos vacíos**.

Ejemplo de Body en formato JSON:

Header

authorization: token

Body

```
{
   "marca": "Asus",
   "imagen": "./img/asus-motherboard.png",
   "habilitado": 0
}
```

Eliminar productos

Ruta: http://localhost:3000/api/producto/:identificador

Método: DELETE

Esta funcionalidad permite a los usuarios que tengan como rol administrador, eliminar un producto a nivel lógico mediante el identificador de este y que con este cambio ya no aparezca en el listado general de productos para ser seleccionado o comprado.

Ejemplo de cómo usar este Endpoint:

Header

authorization: token

Body

Vacío, no es requerido campos adicionales.

Estas fueron las funcionalidades que los administradores pueden hacer de forma única, por lo que ahora ese se describirán los de los usuarios normales.

Usuarios

Registrarse

Ruta: http://localhost:3000/api/registro/:dpi

Método: POST

Esta funcionalidad permite a las personas que no tiene un usuario, crearlo y poder acceder al sistema con sus credenciales. Solo administradores pueden crear otros administradores.

Este Endpoint valida si los campos ingresados **no son campos vacíos ni campos duplicados en base de datos y que el correo y contraseña sean válidos y cumplan con requisitos de seguridad.**

Ejemplo de solicitud en formato JSON:

Header: no se requiere token

Body:

```
{
  "nombres": "Kelvin José",
  "apellidos": "Gómez Morales",
  "fechaNacimiento": "11-03-2000",
  "direccionEntrega": "Casa",
  "nit": 12345,
  "numeroTelefonico": 54321,
  "correoElectronico": "kgomezm11@miumg.edu.gt",
  "clave": "admin",
  "validacionClave": "dw31Das1!”",
}
```

Login

Ruta: http://localhost:3000/api/login

Método: POST

Esta funcionalidad permite a los usuarios registrados anteriormente, poder ingresar al sistema y adquirir un token de autenticación que les servirá para poder acceder a los recursos según sus privilegios y permisos.

Este Endpoint valida si los campos ingresados **no son campos vacíos y cumplen con requisitos de seguridad**.

Ejemplo de solicitud en formato JSON:

Header

authorization: no requiere token

Body:

```
{
   "correoElectronico": " kgomezm11@miumg.edu.gt ",
   "clave": "admin"
}
```

En este caso en específico es donde se genera el token que debemos de enviar con las solicitudes siguientes para poder acceder a ciertos recursos proporcionados.

Ver perfil de usuario

Ruta: http://localhost:3000/api/perfil/:dpi

Método: GET

Esta funcionalidad permite a los usuarios ver sus datos asociados a su cuenta.

Este Endpoint no requiere que se le envíen datos, únicamente que el dpi sea válido y cuente con el token correspondiente.

Ejemplo de solicitud en formato JSON:

Header

authorization: token

Body:

Vacío, no es requerido campos adicionales.

Actualizar datos del perfil

Ruta: http://localhost:3000/api/perfil/:dpi

Método: PATCH

Esta funcionalidad permite a los usuarios registrados anteriormente, poder cambiar los datos que ellos requieran. Cabe mencionar que valida token y su usuario asociado, de tal forma que un usuario no puede acceder ni cambiar los datos de otro usuario.

Este Endpoint valida si los campos nuevos a ingresar **no son campos vacíos, duplicados y cumplen con requisitos de seguridad**.

Ejemplo de solicitud en formato JSON:

Header

authorization: token

Body:

```
{
   "numeroTelefonico": 66778899
}
```

Eliminar un perfil

Ruta: http://localhost:3000/api/perfil/:dpi

Método: DELETE

Esta funcionalidad permite a los usuarios registrados anteriormente, eliminar por completo su perfil, es decir todos sus datos asociados y la invalidación del token

Este Endpoint valida si el usuario ingresado existe en la base de datos y el que usuario según el token sea el correcto mediante el DPI.

Ejemplo de solicitud en formato JSON:

Header

authorization: token

Body:

Vacío, no es requerido campos adicionales.

Obtener lista de todos los productos

Ruta: http://localhost:3000/api/productos

Método: GET

Esta funcionalidad permite a los usuarios obtener un listado de todos los productos que estén habilitados en ese momento.

Ejemplo de solicitud en formato JSON:

Header

authorization: token

Body:

Vacío, no es requerido campos adicionales.

Respuesta:

Listado de todos los productos y sus detalles.

Obtener detalle de un producto en específico

Ruta: http://localhost:3000/api/producto/:identificador

Método: GET

Esta funcionalidad permite a los usuarios obtener el detalle de un producto en específico que esté habilitado en ese momento.

Ejemplo de solicitud en formato JSON:

Header

authorization: token

Body:

Vacío, no es requerido campos adicionales.
Respuesta:

Listado de detalles de dicho producto a solicitar

Listado productos del carrito

Ruta: http://localhost:3000/api/carrito/

Método: GET

Esta funcionalidad permite a los usuarios obtener el detalle de todos los productos que tengan en el carrito en ese momento.

Ejemplo de solicitud en formato JSON:

Header

authorization: token

Body:

Vacío, no es requerido campos adicionales.

Respuesta:

Listado de detalles de todos los productos añadidos al carrito

Agregar productos al carrito

Ruta: http://localhost:3000/api/carrito

Método: POST

Esta funcionalidad permite agregar productos existentes del listado general al carrito de compras, con esto el usuario podrá comprar cierto producto con dicha cantidad

Este Endpoint valida si los campos nuevos a ingresar **no son campos vacíos, duplicados, que exista el producto y que exista la disponibilidad necesaria.**

Ejemplo de solicitud en formato JSON:

Header

authorization: token

Body:

```
{
   "identificador": 1,
   "cantidad": 9
}
```

Quitar productos al carrito

Ruta: http://localhost:3000/api/carrito

Método: DELETE

Esta funcionalidad permite quitar productos existentes del carrito de compras, con esto el usuario podrá comprar quitar producto y se devolverá la cantidad deseada a la disponibilidad.

Este Endpoint valida si el identificador del producto a eliminar **no sea campo vacío, sea valido y que exista el producto en el carrito.**

Ejemplo de solicitud en formato JSON:

Header

authorization: token

Body:

```
{
   "identificador": 1
}
```

Compras realizadas por cliente

Ruta: http://localhost:3000/api/compra

Método: GET

Esta funcionalidad permite a los usuarios obtener el detalle de todos los productos que hayan comprado en el sistema.

Ejemplo de solicitud en formato JSON:

Header

authorization: token

Body:

Vacío, no es requerido campos adicionales.

Respuesta:

Listado de detalles de todos los productos comprados

Realizar una compra

Ruta: http://localhost:3000/api/compra

Método: POST

Esta funcionalidad permite a los usuarios comprar todos los productos que tengan en su carrito en ese momento.

Ejemplo de solicitud en formato JSON:

Header

authorization: token

Body:

Vacío, no es requerido campos adicionales.

Respuesta:

Confirmación de la compra.

Conclusión

En conclusión, este manual técnico ha proporcionado una visión detallada y completa de nuestro sistema de comercio electrónico, que incluye funciones esenciales como registro, inicio de sesión, listado de productos, carrito de compras y un apartado de ventas. Hemos explorado los aspectos técnicos de cada uno de estos componentes, aunque no muy a fondo cómo el código.

Esperamos que este manual sirva como ayuda para que las personas con conocimiento más técnico puedan entender, apoyar y evaluar el sistema creado.

Recursos utilizados

Drive a los diagramas realizados:

https://drive.google.com/drive/folders/1_MRWZvrtZS-EsPza6gWmg_vJd8TiZ_Lu?usp=sharing

Colección de Requests en POSTMAN (JSON)

https://drive.google.com/drive/folders/13H8A3evRKdbIR0h1G_v73tW55e1hRFkS?usp=sharing

Colecciones de MongoDB (JSON)

https://drive.google.com/drive/folders/1e9Zvd96k-PqTg9Gi8rB3lrimvBvsQftV?usp=sharing