

# Especificación de Requerimientos

## Descripción del Diseño

### Terra

Apellidos, Nombres	Correo electrónico	Rol
<b>Alejandra Montealegre</b>	<a href="mailto:mariamontealegreb@gmail.com">mariamontealegreb@gmail.com</a>	Back end
<b>Miguel Fernandes</b>	<a href="mailto:miguelangelantoniofernandez@gmail.com">miguelangelantoniofernandez@gmail.com</a>	Front end
<b>Andres Camilo Rojas</b>	<a href="mailto:camilin.andres90@gmail.com">camilin.andres90@gmail.com</a>	Tester
<b>Omar Castañeda</b>	<a href="mailto:omarricardocc@gmail.com">omarricardocc@gmail.com</a>	Master scrum
<b>Cristian Rodriguez</b>		Bases de datos

Fecha de presentación: 26/11/2022



## Contenido

<b>1</b>	<b>INTRODUCCIÓN.....</b>	<b>3</b>
1.1	PROPÓSITO .....	3
1.2	ALCANCE O ÁMBITO DEL SISTEMA .....	3
1.3	DEFINICIONES, ACRÓNIMOS Y ABREVIATURAS .....	3
1.3.1	<i>Referencias</i> .....	3
1.4	PERSPECTIVA GENERAL DEL DOCUMENTO.....	4
<b>2</b>	<b>DESCRIPCIÓN GENERAL DE LA APLICACIÓN .....</b>	<b>4</b>
2.1	PERSPECTIVA DE LA APLICACIÓN.....	4
2.2	FUNCIONES DE LA APLICACIÓN .....	4
2.3	CARACTERÍSTICAS DE LOS USUARIOS .....	4
2.4	RESTRICCIONES.....	4
2.5	SUPOSICIONES Y DEPENDENCIAS .....	4
2.6	REQUERIMIENTOS DIFERIDOS .....	5
<b>3</b>	<b>REQUERIMIENTOS ESPECÍFICOS.....</b>	<b>5</b>
3.1	REQUERIMIENTOS .....	5
3.1.1	<i>Product Backlog</i> .....	5
3.1.2	<i>Ciclo de Sprints del proyecto</i> .....	6
3.1.3	<i>Sprint Backlog</i> .....	6
3.1.4	<i>Historias de usuario (Tareas y Subtareas)</i> .....	7
3.1.5	<i>Mecánica de organización del grupo. (Reuniones, evidencias/artefactos)</i> .....	7
3.2	MODELO DE REQUERIMIENTOS .....	7
3.2.1	<i>Modelo de Casos de Uso</i> .....	7
<b>4</b>	<b>DESCRIPCIÓN DEL DISEÑO. ....</b>	<b>9</b>
4.1	INTERFAZ GRÁFICA (MOCKUPS).....	9
<b>5</b>	<b>GESTIÓN DE LA CONFIGURACIÓN .....</b>	<b>12</b>
<b>6</b>	<b>PRUEBAS .....</b>	<b>17</b>
6.1	DESCRIPCIÓN DE PRUEBAS UNITARIAS .....	17
6.2	DESCRIPCIÓN DE PRUEBAS DE ACEPTACIÓN .....	¡ERROR! MARCADOR NO DEFINIDO.
<b>7</b>	<b>GLOSARIO .....</b>	¡ERROR! MARCADOR NO DEFINIDO.
<b>8</b>	<b>ANEXO(S) .....</b>	¡ERROR! MARCADOR NO DEFINIDO.

# 1 INTRODUCCIÓN

El presente documento tiene como finalidad presentar la documentación, descripción, explicación y detalles sobre el uso configuración y código empleado para realizar una aplicación amigable para el usuario y operador en la interminable tarea de almacenar, vender prestar o remover de un inventario un libro.

## 1.1 Propósito

El documento tiene como fin presentar al cliente un resumen ejecutivo de las principales características del programa del cual se va a hacerse acreedor, para llevar la tarea de inventarios y venta de una tienda de manera más amigable y fácil.

## 1.2 Alcance o Ámbito del Sistema

Terra es un programa que integra los diferentes beneficios a la hora de almacenar, vender, eliminar o agregar un insumo a su inventario, permitiendo a los dueños de tiendas o e – comercios la fácil implementación de una plataforma de venta y manejo de catalogo.

## 1.3 Definiciones, Acrónimos y Abreviaturas

PHP: Es un lenguaje de programación enfocado en el desarrollo web, que permite a los usuarios cambiar modificar y agregar opciones al entorno de trabajo visual del programa.

CSS: Es un lenguaje de programación muy usado para establecer el diseño visual de los documentos web, e interfaces de usuario escritas en HTML o XHTML.

Mongo BD: Es una base de datos de documentos que ofrece una gran escalabilidad y flexibilidad, y un modelo de consultas e indexación avanzado.

Express JS: Proporciona una delgada capa de características de aplicación web básicas

React JS: Es una biblioteca Javascript de código abierto diseñada para crear interfaces de usuario con el objetivo de facilitar el desarrollo de aplicaciones en una sola página. Es mantenido por Facebook y la comunidad de software libre

Node JS: Es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor basado en el lenguaje de programación JavaScript, asíncrono, con E/S de datos en una arquitectura orientada a eventos.

### 1.3.1 Referencias

- 1) IEEE Std-830-1998.
- 2) ISO-IEC-IEEE-298148
- 3) IEEE Std-1016-2009
- 4) ISO/IEC/IEEE 29148:2011
- 5) OMG Unified Modeling Language
- 6) Schwinger, W.; Koch, N. "Modeling Web Applications", Chapter 3 en: Kappel, G.; Pröll,

7) Koch, N.; Knapp, A.; Zhang, G.; Baumeister, H. "UML-Based Web Engineering. An Approach Based on Standards", Chapter 7 en: Rossi, G.; Pastor.

## **1.4 Perspectiva General del Documento**

El documento está dirigido para el que operador y el usuario tengan una idea general de cómo funciona Terra y que ventajas sobre otros programas de almacenamiento interactivo de insumos en cuanto a tiempo y requerimientos a partir de una interfaz minimalista.

## **2 DESCRIPCIÓN GENERAL DE LA APLICACIÓN**

### **2.1 Perspectiva de la Aplicación**

Terra es una aplicación que permite el almacenamiento por medio de bases de datos de inventarios ya sea de insumos como de clientes, además de estar enfocando en la venta de productos, lo que permitiera al usuario llevar de manera mas ágil las cuentas y de esta forma determinar sus veneficios diarios y mensuales.

### **2.2 Funciones de la Aplicación**

Terra cuenta con las siguientes funciones:

- 1) Venta de productos y actualización de precios.
- 2) Listados de usuarios y personal.

### **2.3 Características de los Usuarios**

Todas aquellas personas que tengan un inventario, negocio o su principal nicho de mercado sea el comercio.

### **2.4 Restricciones**

Las siguientes son las restricciones con las cuales Terra cuenta.

- No se realizará manejo de inventario.
- No se realizará manejo de información referente a domicilios.
- No se realizará manejo de información referente a facturación.
- No se realizará manejo de información referente a proveedores.
- No se realizará manejo de información referente al estado contable del negocio.

### **2.5 Suposiciones y Dependencias**

Las siguientes son las suposiciones de Terra para asegurar su optimo funcionamiento.

- Se deberá contar con la información el número de clientes.
- Contar con la información de sus empleados en caso de tenerlos.

- Fotografías de los productos.
- Precios de los productos

## 2.6 Requerimientos Diferidos

Actualmente Terra tiene como propósito implementar en versiones futuras la conexión a diferentes bases de datos almacenadas en la nube, que no requieran un servidor físico ni de un tipado de base de datos específico, además se busca la interacción con diferente host que permitan la estimar cuales generan menor latencia y mayor rendimiento en la generación de resultados.

## 3 REQUERIMIENTOS ESPECÍFICOS

### 3.1 Requerimientos

Los requerimientos específicos para Terra actualmente son los siguientes:

- Interacción con las diferentes bases de datos usuarios, inventario, usuarios y empleados etc.

#### 3.1.1 Product Backlog

- 1) Creación de Repositorio de GitHub
- 2) Enlace de repositorios
- 3) Creación de Trello
- 4) Documento de gestión de configuración
- 5) Historias de usuario en Trello
- 6) Acta de entrega y reunión
- 7) búsqueda y creación de plantilla
- 8) Montaje html y css de la página principal y secundarias
- 9) Montaje js de las paginas
- 10) Conexión backend
- 11) Entrega para test
- 12) Corrección errores informados por las pruebas
- 13) Configuración y alistamiento de ambiente (validando si se realiza en atlas
- 14) Creación de la base de datos
- 15) Creación de la colección en DB mongo
- 16) Insertar documento de prueba
- 17) Configuración del entorno
- 18) Conexión con el localhost
- 19) Conexión BD Mongo
- 20) Creación de los esquemas
- 21) Creación de los controladores
- 22) Creación de los Reuters
- 23) Conexión con el Front
- 24) Instalación de herramientas indicadas para pruebas
- 25) Generación de Script de Pruebas

- 26) Pruebas Unitarias Back
- 27) Pruebas Unitarias Front
- 28) Pruebas de Acceso
- 29) Pruebas de Integración Back-BD
- 30) Pruebas Funcionales API

### 3.1.2 Ciclo de Sprints del proyecto

A continuación, se relacionarán los sprints y sus fechas para evaluación.

- Sprint 1 23/10/2022
- Sprint 2 06/11/2022
- Sprint 3 20/11/2022
- Sprint 4 27/11/2022

### 3.1.3 Sprint Backlog

#### Sprint 1

- Creación de Repositorio de Github
- Enlace de repositorios
- Creación de Trello
- Documento de gestión de configuración (avance)
- Historias de usuario en Trello
- Historia de usuario
- Plantilla Front End

#### Sprint 2

- Interfaz Grafica
- Documento de gestión de configuración (avance)
- búsqueda y creación de plantilla
- Montaje html y css de la página principal y secundarias
- Montaje js de las paginas
- Conexión backend
- Configuración del entorno
- Conexión con el localhost
- Conexión BD Mongo
- Creación de los esquemas
- Entrega para test Historias de usuario en Trello (Actualizadas)
- Historia de usuario (Actualizadas)
- Acta de entrega y reunión (Actualizadas)

#### Sprint 3

- Corrección errores informados por las pruebas
- Configuración y alistamiento de ambiente (validando si se realiza en atlas)
- Creación de la base de datos
- Creación de la colección en DB mongo
- Insertar documento de prueba
- Configuración del entorno
- Conexión con el localhost
- Conexión BD Mongo
- Creación de los esquemas

- Creación de los controladores
- Creación de los Reuters
- Conexión con el Front
- Instalación de herramientas indicadas para pruebas
- Historias de usuario en Trello (Actualizadas)
- Historia de usuario (Actualizadas)
- Acta de entrega y reunión (Actualizadas)

#### Sprint 4

- Generación de Script de Pruebas
- Pruebas Unitarias Back
- Pruebas Unitarias Front
- Pruebas de Acceso
- Pruebas de Integración Back-BD
- Pruebas Funcionales API
- Historias de usuario en Trello (Actualizadas)
- Historia de usuario (Actualizadas)
- Acta de entrega y reunión (Actualizadas)

### 3.1.4 Historias de usuario (Tareas y Subtareas)

### 3.1.5 Mecánica de organización del grupo. (Reuniones, evidencias/artefactos)

Para la evaluación de Terra se llevarán a cabo reunión semanales y reuniones para acta de entrega de los diferentes product backlog y de esta forma llamar un alto grado de control en el desarrollo de la aplicación.

## 3.2 Modelo de Requerimientos

A continuación, se validarán los métodos de caso que serán usados en los diferentes métodos a evaluar o información de entrada.

### 3.2.1 Modelo de Casos de Uso

#### CU-01: "Item del catálogo"

Iniciador	Usuario
Otros actores	Item
Precondiciones	Compra
Flujo básico	
Actor	Sistema
1 Usuario	Selección de artículo
botón selección de articulo	2 conexión Front end Back end
Ventana del articulo	3 conexión back end base de datos
4 precio del articulo	
Flujo alternativo 1	NA



## CU-02: "Búsqueda dentro del catálogo"

Iniciador	Usuario
Otros actores	Boton búsqueda
Precondiciones	Buscar
Flujo básico	
Actor	Sistema
1 usuario	Buscar articulo
botón selección búsqueda	2 conexión Front end Back end
Ventana del articulo	3 conexión back end base de datos
4 artículo disponible o no disponible	
Flujo alternativo 1	NA

## CU-03: "Nuevo artículo"

Iniciador	Usuario
Otros actores	botón búsqueda
Precondiciones	Buscar
Flujo básico	
Actor	Sistema
1 Usuario	Implementar un nuevo artículo
botón implementar nuevo artículo (solo administrador)	2 conexión Front end Back end
Ventana del articulo	3 conexión back end base de datos
4 artículo disponible y en línea para los clientes	
Flujo alternativo 1	NA

## CU-04: "Botón catalogo"

Iniciador	Usuario
Otros actores	botón búsqueda
Precondiciones	Buscar
Flujo básico	
Actor	Sistema
1 Usuario	Botón catalogo
botón catalogo	2 conexión Front end Back end
Listado de artículos	3 conexión back end base de datos
4 Pagina con todos los artículos organizados y con posibilidad de filtro	
Flujo alternativo 1	NA

## CU-05: "Pagina administración"

Iniciador	Usuario
Otros actores	botón búsqueda
Precondiciones	Buscar
Flujo básico	
Actor	Sistema
1 administrador	Panel de control
Empleados	2 conexión Front end Back end
Listado de empleados	3 conexión back end base de datos
4 Pagina con todos los empleados	

Flujo alternativo 1	NA
---------------------	----

## CU-06: "Carro de compras"

Iniciador	Usuario
Otros actores	botón de compra
Precondiciones	Buscar
Flujo básico	
Actor	Sistema
1 Usuario	Inicio, Inventario, Ventas
Administrador	2 conexión Front end Back end
Lista de valor total de los productos	3 conexión back end base de datos
4 Pagina de carrito	
Flujo alternativo 1	NA

## CU-06: "Redes Sociales"

Iniciador	Usuario
Otros actores	Botón a redes sociales
Precondiciones	Buscar
Flujo básico	
Actor	Sistema
1 Usuario	Inicio, Inventario, Ventas
Administrador	2 conexión Front end Back end
Enlace de páginas con redes sociales	3 conexión back end base de datos
4 Pagina de interés.	
Flujo alternativo 1	NA

## 4 DESCRIPCIÓN DEL DISEÑO.

### 4.1 Interfaz gráfica (Mockups).

Una vez avanzado en la realizada los bosquejos iniciales de al interfaz se realizó la implementación en HTML puro tal y como se muestra continuación.

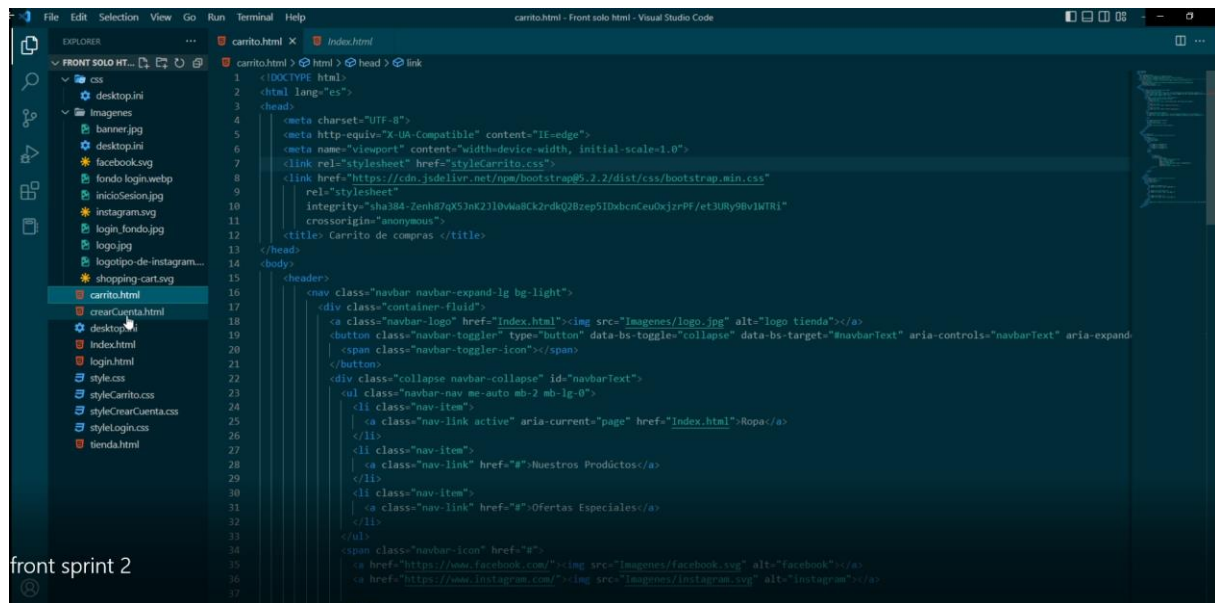


Figura 1 Código HTML Front



Figura 2 Interfaz de inicio para la tienda



Figura 3 Pagina de logging



Figura 4 Pagina de creación de cuenta

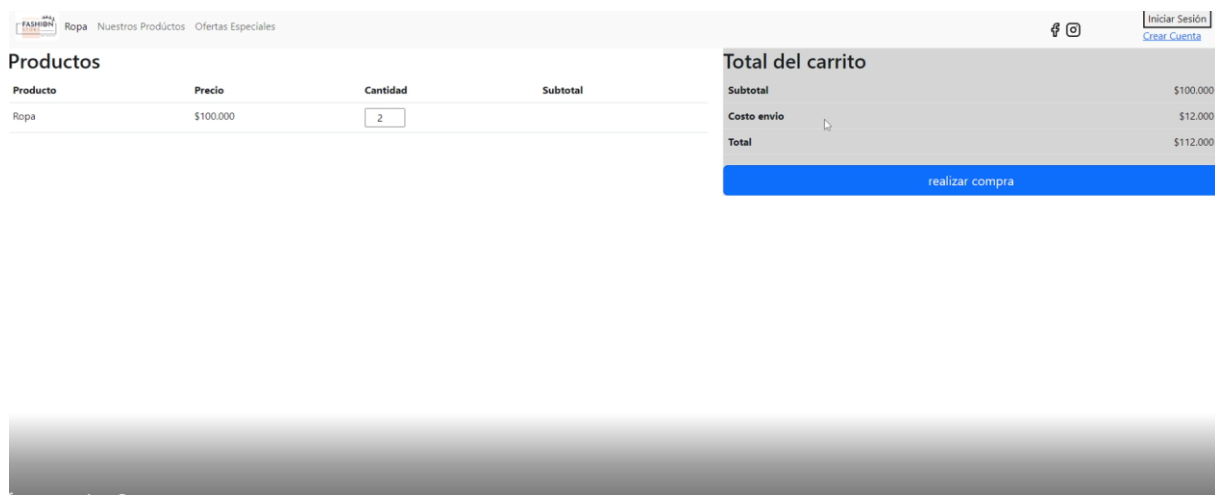


Figura 5 Pagina de compras

Posterior a esto se procede a la migración a React, tomando parte del código nativo.

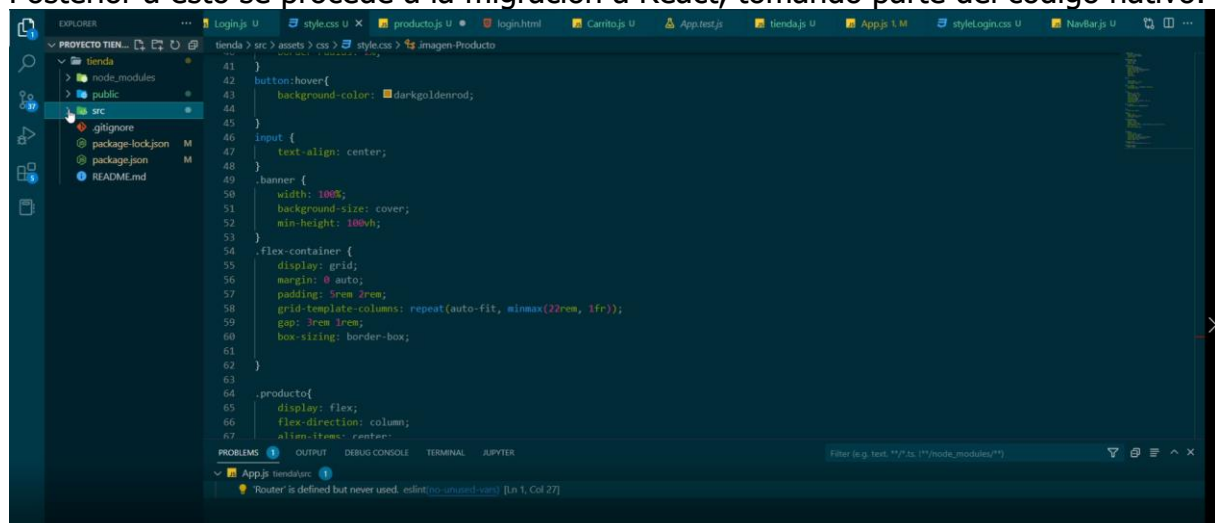


Figura 6 Migración a React

Además de esto se genera una página nueva para el catalogo de productos y su conexión al carro de compras.



Figura 7 Catálogo de productos

## 5 GESTIÓN DE LA CONFIGURACIÓN

### Back end.

Para el desarrollo del e-commerce desde la parte de back end se ha implementado las tecnologías anteriormente descritas, además de tener ya una conexión funcional con la base de datos, empleados productos y usuarios tal y como se muestra continuación.

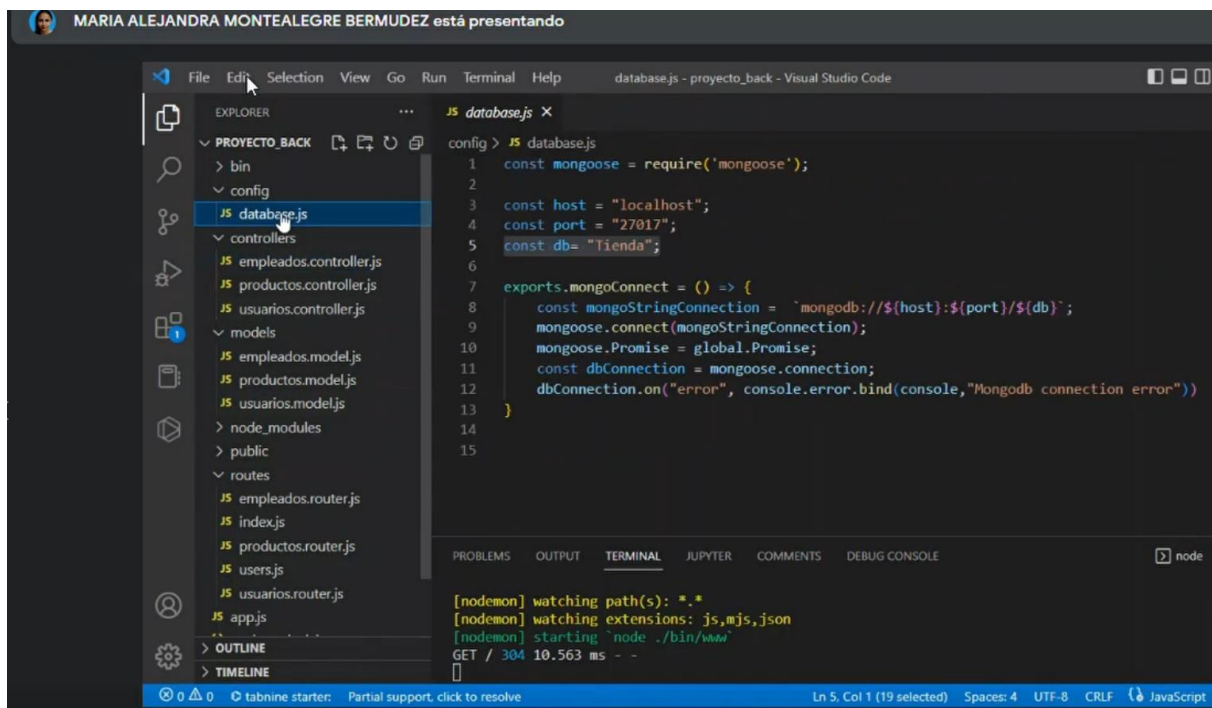


Figura 8 Back end

Se generaron a partir de los controladores se generaron las funciones necesarias para realizar un cruce y realizar su posterior verificación en postman.



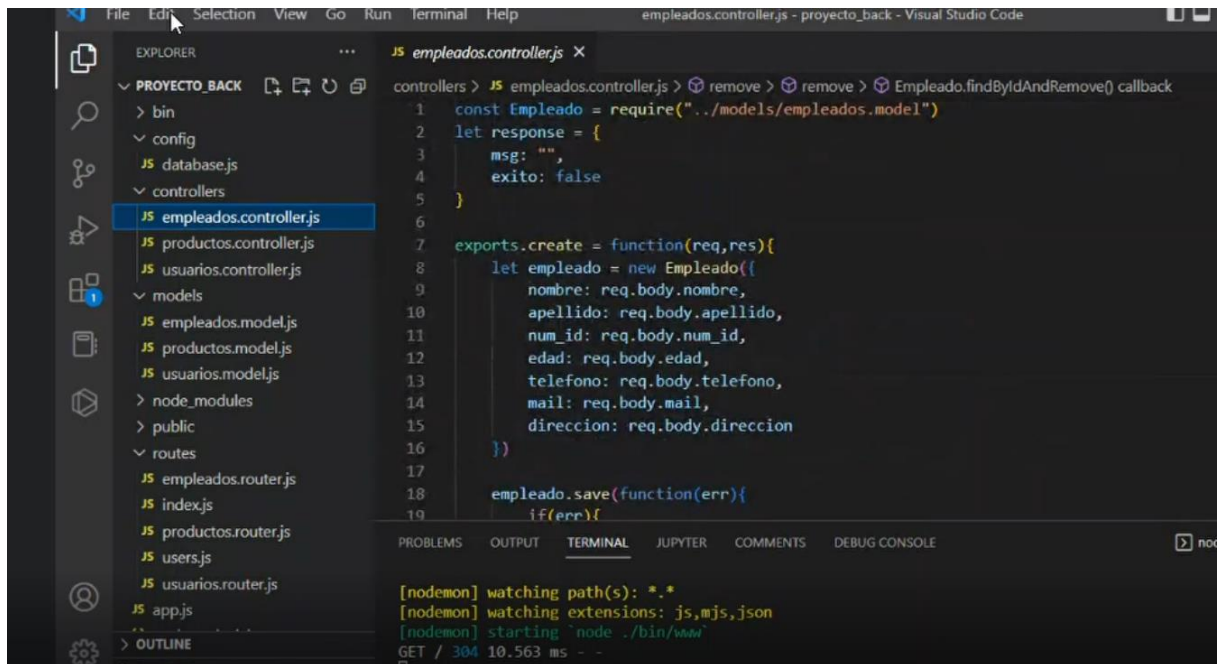


Figura 9 Implementación de las funciones y conexiones.

Se tienen hasta el momento las siguientes funciones.

- Crear un nuevo artículo.
- Guardar un nuevo artículo.
- Consultar un nuevo artículo.
- Actualizaciones de registros.
- Consultas de registros.
- Filtros en las consultas de los filtros por medio de un ID.
- Función de eliminar.

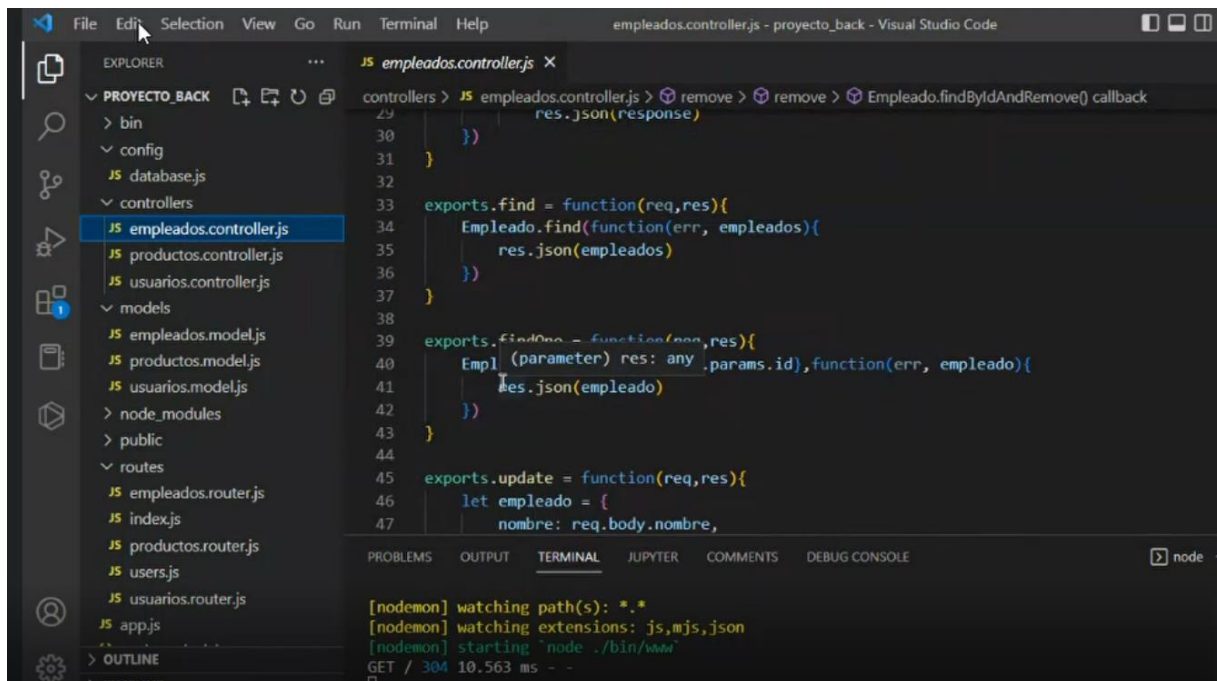
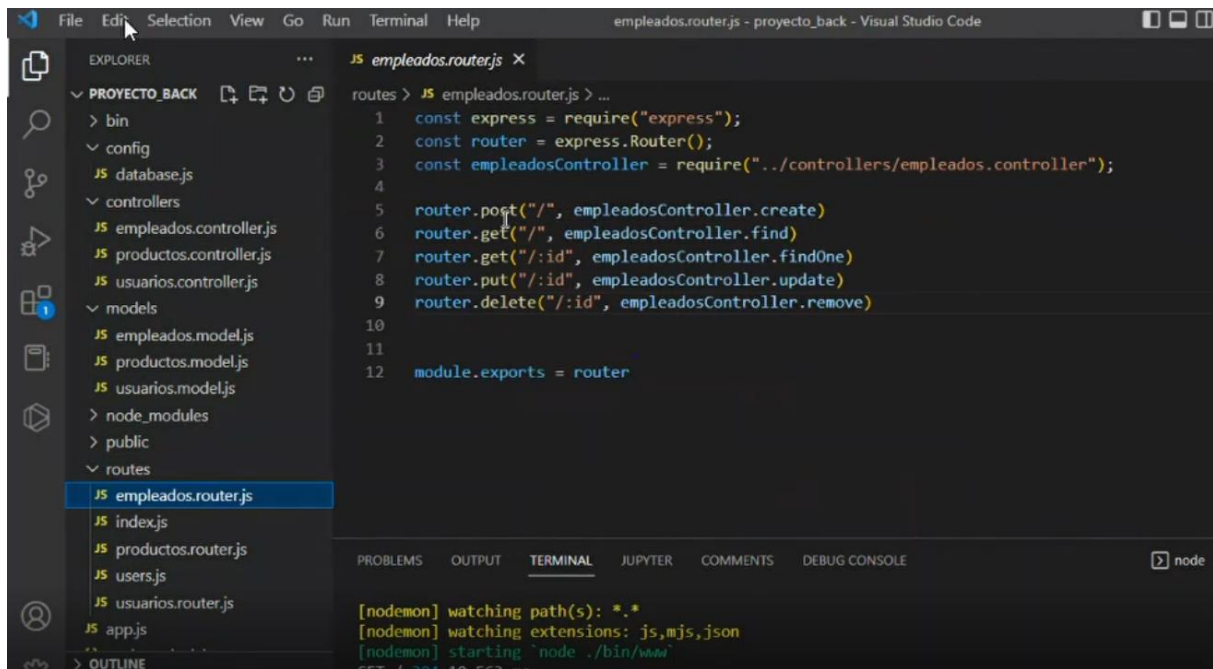


Figura 10 Funciones

También se generaron los archivos de tipo Router que nos permiten identificar las funciones declaradas.



```
File Edit Selection View Go Run Terminal Help empleados.router.js - proyecto_back - Visual Studio Code

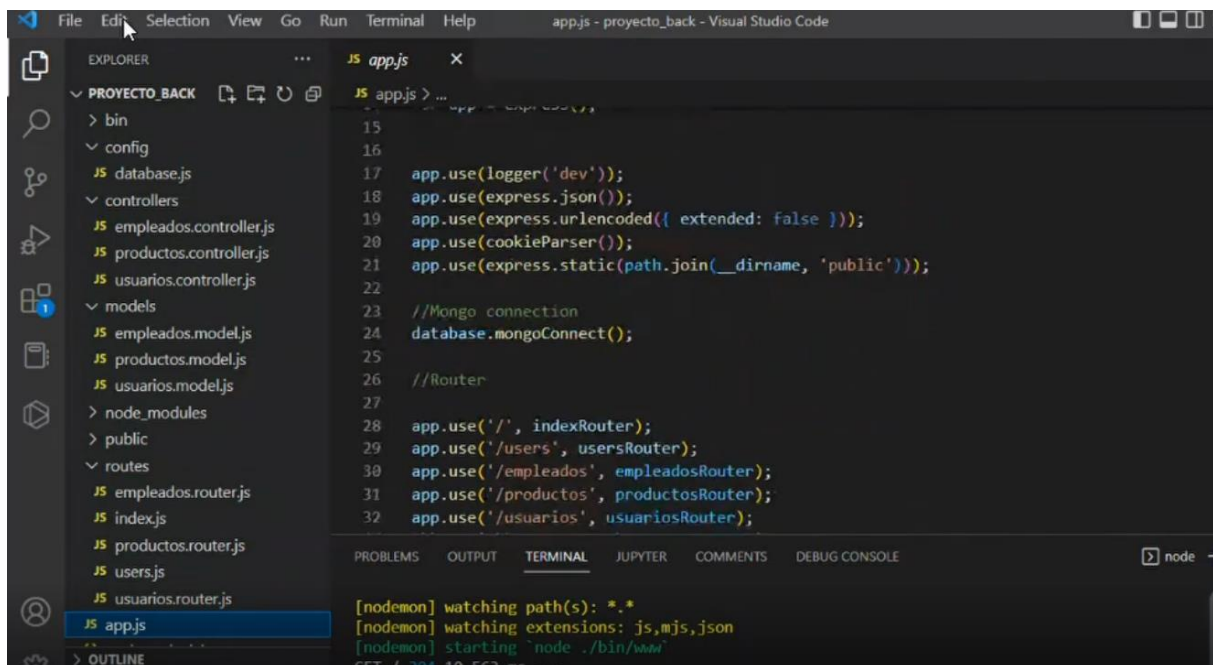
EXPLORER
PROYECTO_BACK
  bin
  config
  database.js
  controllers
    empleados.controller.js
    productos.controller.js
    usuarios.controller.js
  models
    empleados.model.js
    productos.model.js
    usuarios.model.js
  node_modules
  public
  routes
    empleados.router.js
    index.js
    productos.router.js
    users.js
    usuarios.router.js
  app.js
  OUTLINE

routes > JS empleados.router.js
1 const express = require("express");
2 const router = express.Router();
3 const empleadosController = require("../controllers/empleados.controller");
4
5 router.post("/", empleadosController.create)
6 router.get("/", empleadosController.find)
7 router.get("/:id", empleadosController.findOne)
8 router.put("/:id", empleadosController.update)
9 router.delete("/:id", empleadosController.remove)
10
11
12 module.exports = router

PROBLEMS OUTPUT TERMINAL JUPYTER COMMENTS DEBUG CONSOLE
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting 'node ./bin/www'
GET / 304 10.563 ms
```

Figura 11 Funciones declaradas dentro de archivo Router

Esto con el fin de que postman pueda identificar cada uno de los envíos de datos. Además, se agregó a nuestro archivo APP.JS los routers de cada uno de nuestros esquemas.



```
File Edit Selection View Go Run Terminal Help app.js - proyecto_back - Visual Studio Code

EXPLORER
PROYECTO_BACK
  bin
  config
  database.js
  controllers
    empleados.controller.js
    productos.controller.js
    usuarios.controller.js
  models
    empleados.model.js
    productos.model.js
    usuarios.model.js
  node_modules
  public
  routes
    empleados.router.js
    index.js
    productos.router.js
    users.js
    usuarios.router.js
  app.js
  OUTLINE

JS app.js
15 const express = require("express");
16
17 app.use(logger('dev'));
18 app.use(express.json());
19 app.use(express.urlencoded({ extended: false }));
20 app.use(cookieParser());
21 app.use(express.static(path.join(__dirname, 'public')));
22
23 //Mongo connection
24 database.mongooseConnect();
25
26 //Router
27
28 app.use('/', indexRouter);
29 app.use('/users', usersRouter);
30 app.use('/empleados', empleadosRouter);
31 app.use('/productos', productosRouter);
32 app.use('/usuarios', usuariosRouter);

PROBLEMS OUTPUT TERMINAL JUPYTER COMMENTS DEBUG CONSOLE
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting 'node ./bin/www'
GET / 304 10.563 ms
```

Figura 12 Archivo APP.JS

## Base de datos.

Una vez definidos todos los modelos de tablas o variables a usar se procede a crear la tabla de datos.

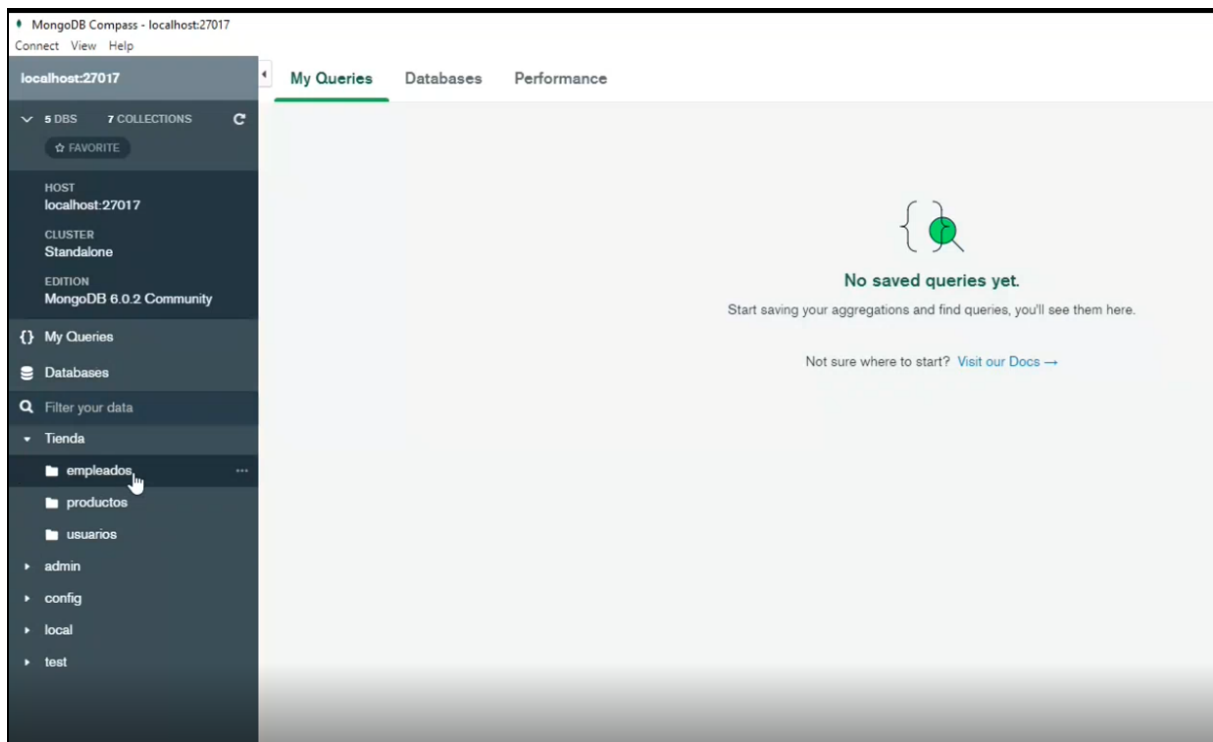


Figura 13 Bases de datos referencias

Para este caso contamos con tres tipos como lo son los empleados, los productos y los usuarios, con sus respectivas variables

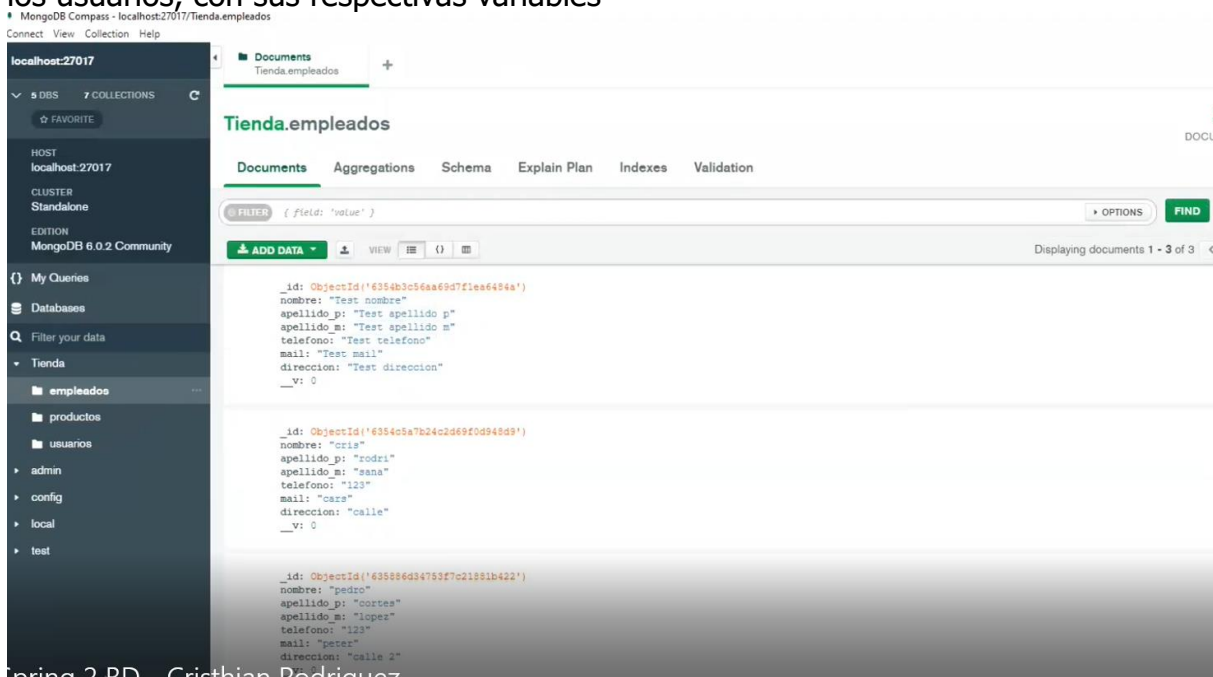


Figura 14 Base de datos usuarios

Además de realizaron un conjunto de pruebas de validación para asegurar el correcto funcionamiento de las bases de datos no relacionales, donde para los empleados se definieron el nombre, el apellido, el numero id, el teléfono, el mail y la dirección. Para los productos se definieron el id, el género, el motivo, la talla, el color, la decrecían y el precio.



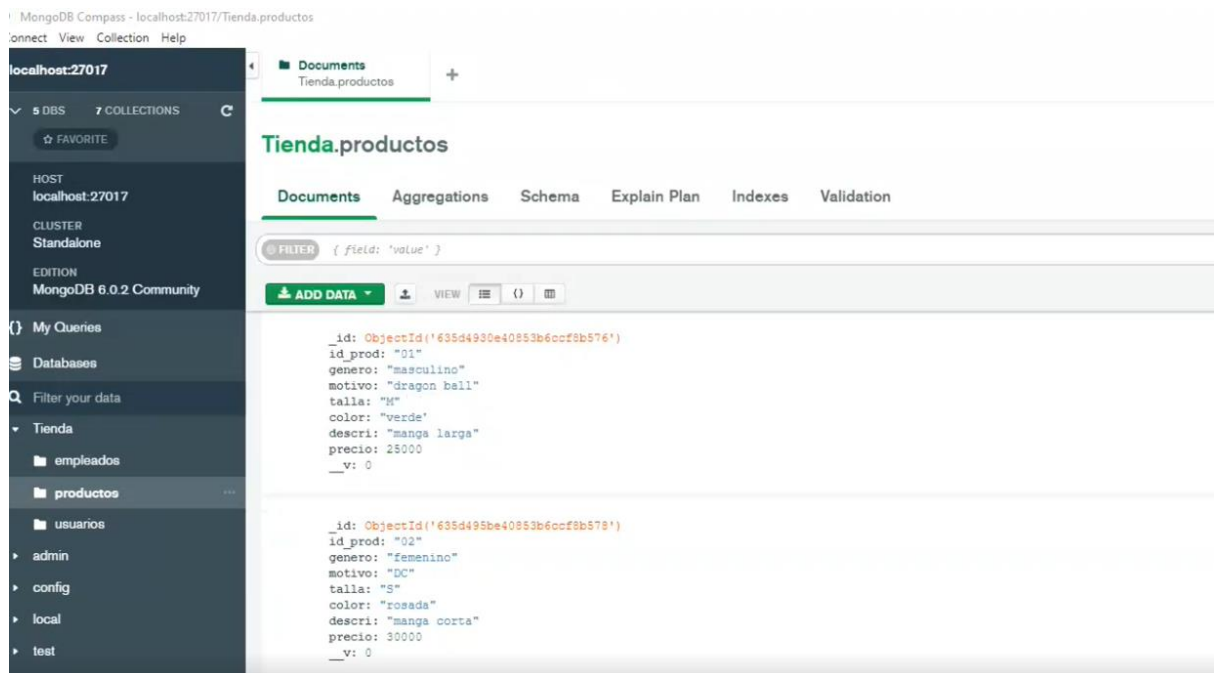


Figura 15 Base de datos catálogo.

Para los usuarios se definieron el nombre, los apellidos, el mail y el género.

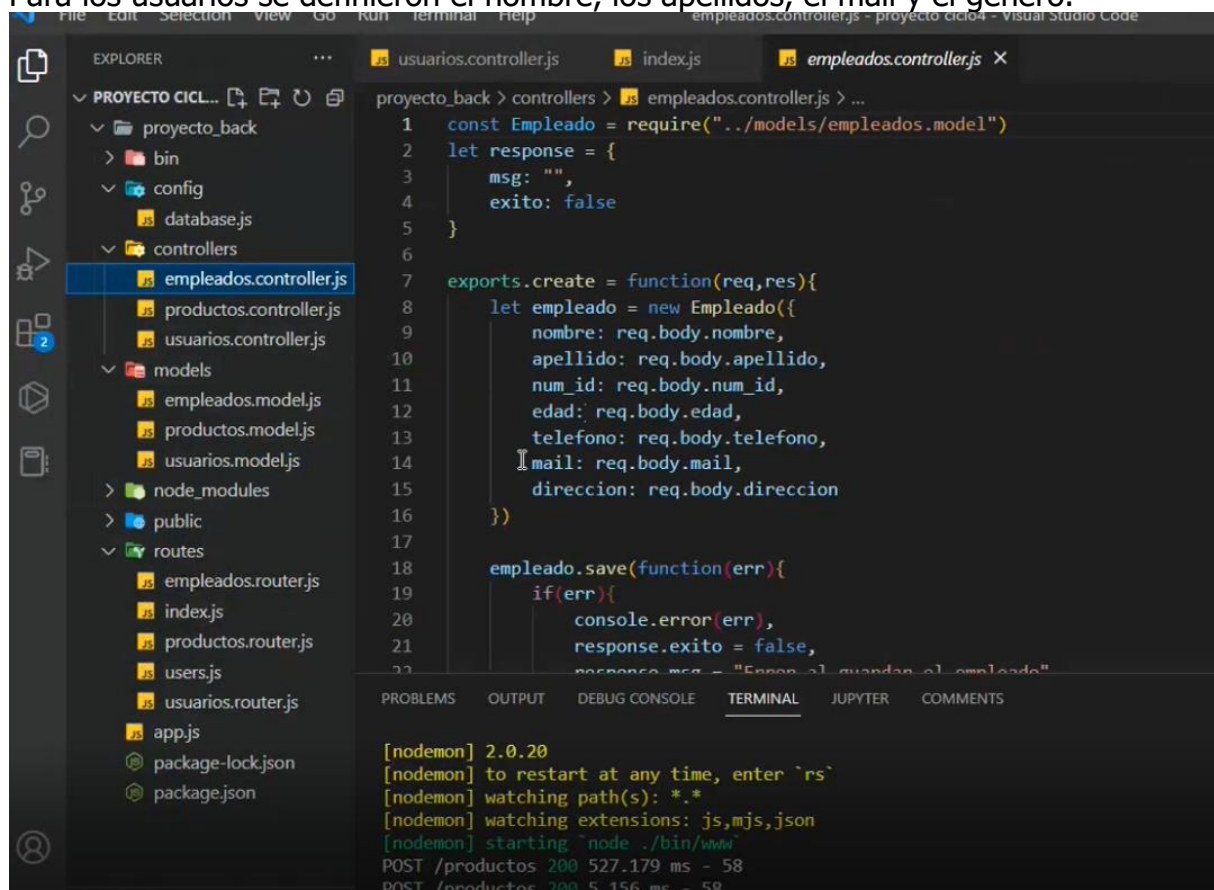


Figura 16 Base de datos conexión back end

## 6 PRUEBAS

### 6.1 Descripción de pruebas unitarias

Para la entrega actual se realizaron las pruebas del cru de las tablas generadas desde la base de datos con resultados favorables tal y como se puede apreciar en la siguiente figura.

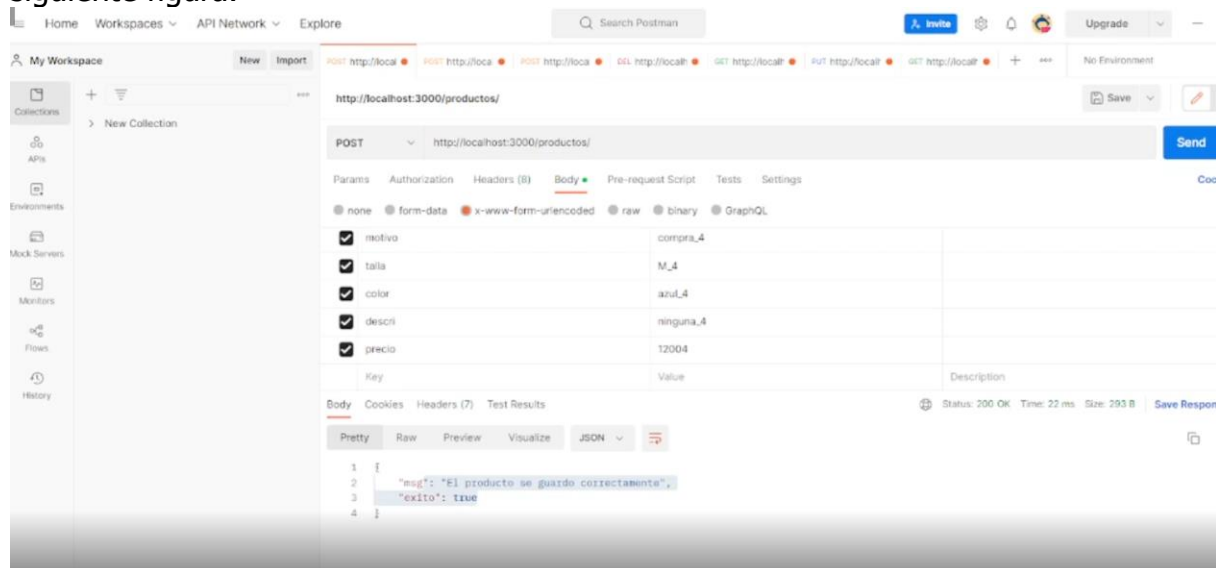


Figura 17 Testeo pruebas unitarias.

Resultados, los productos se puede actualizar y eliminar junto con los empleados, todas las partes del cru están funcionado correctamente, esto realizado a partir del programa postman.

## 7 CONCLUSIONES Y LECCIONES APRENDIDAS.

A lo largo de estos ciclos hemos aprendió la importancia del trabajo en equipo, ya que si bien alguna de las partes del equipo cumple a cabalidad con los insumos entregados y utilizando como metáfora del proyecto una gran cadena, si alguno de estos eslabones es débil en caso de presentar algún problema la cadena se romperá por este, entonces intentar solventar las debilidades y fortalezas de todos esos eslabones es lo que nos permite integrar en este caso un se realice un proyecto fuerte y robusto en donde todas las partes participen de manera fuerte, estratégica y en equipo.

Una lección aprendida a lo largo de este ciclo fue el estudio autónomo como clave del desarrollo de actividades específicas del proyecto, el uso de nuevas tecnologías, además de diferentes alternativas de solución para una problemática a partir de la autonomía en la búsqueda de respuestas y conocimiento.

El uso de herramientas para la coordinación y gerencia de proyecto fue algo clave en el desarrollo del proyecto ya que si bien la comunicación entre los participantes del proyecto por medio de redes sociales (whatsapp, Facebook, etc) el uso de

herramientas como trello y metodologías de tipo scrum permitió una sana distribución de tareas y con control minucioso y clave para la elaboración de propuestas.

Limitar el alcance para poder cumplir con las metas establecidas fue algo necesario ya que, si bien se tenía una idea clara acerca de que se iba realizar, ampliar las alternativas formas y portafolio de servicio de la página incurría en tiempo mayores y por ende al fracaso en la entrega de insumos.

La lluvia de ideas para la solvencia de problemas fue uno de los ejes principales del proyecto ya que nos permitió como equipo proponer diferentes alternitas de solución y someterlas a un diagnóstico de alternativas que permitiera escoger la mejor en términos de costo – beneficio.

En conclusión el proyecto pudo terminarse de acuerdo a los lineamientos estipulados por la Universidad Tecnológica de Pereira, ya que se conto con el apoyo de excelentes profesor, profesionales y una comunicación continua con la institución ayudando de una u otra manera a evitar una mayor deserción.