CREACIÓN DE UNA APLICACIÓN WEB PARA UNA TIENDA DE COMPONENTES DE COMPUTADORA

MANUAL TECNICO

JOSE CARLOS CASTRO CASTILLO

DAVID JOSE JADID HOYOS

INTRODUCCIÓN

Este manual técnico permite conocer la lógica detrás del proyecto de una aplicación web para una tienda de componentes de computadora, incluyendo los lenguajes de programación utilizados, los pasos a seguir para el desarrollo del proyecto y la descripción de todas las herramientas o programas que fueron implementados en la contracción de mismo.

1. ESPECIFICACIONES TÉCNICAS

En la tabla que se muestra a continuación se encuentran las especificaciones o requisitos técnicos para poder desarrollar la aplicación

CARACTERISTICA	DESCRIPCIÓN
Sistema Operativo	Windows 7 o superior, 32 - 64 bits.
Manejador de Base de Datos	MongoDB versión 4.4.4, Robo 3T 1.4.3.
Lenguajes de Programación	JavaScript, Node.js 14.17.0 con npm 6.14.13, React, HTML5, CSS con Framework Bootstrap 4.
Navegador Web	Cualquier navegador compatible con HTML5.
Desktop	Cualquier computador con navegador web compatible con HTML5.
Equipos portátiles	Cualquier equipo portátil con navegador compatible con HTML5.

2. DESARROLLO DE LA APLICACIÓN

La aplicación fue desarrollada utilizando React, la cual es una librería que está enfocada en el desarrollo de interfaces de usuario bajo el lenguaje de programación JavaScript, esta interfaz de usuario además de ser creada con React es crucial el uso de los lenguajes que se mencionaron anteriormente, como lo son HTML5, CSS y el Framework Bootstrap 4, en lado del servidor y la construcción de la lógica del BackEnd se implementa Node.js el cual es un entorno que trabaja en tiempo de ejecución y permite el soporte para APIS, a su vez presenta un gran rendimiento y alta escalabilidad, cabe resaltar que al implementar node también es necesario implementar un gestor de paquetes, este no es más que npm, el cual proporciona acceso a muchos paquetes reutilizables y permite un buen manejo en las dependencias a la hora de compilar nuestra aplicación.

Para el manejo de la base de datos que estará conectada con la aplicación web se hace uso de MongoDB, que es una base de datos de documentos que ofrece una buena escalabilidad y flexibilidad, así como un modelo de consultas e indexación avanzado, para una mayor facilidad en el manejo de la base datos se recurre Robo 3T, antes denominado Robomongo, la cual es una interfaz gráfica de usuario para MongoDB, nos permite crear documentos, consultas y visualizar los documentos que ya se encuentren en la base de datos.

3. HERRAMIENTAS UTILIZADAS

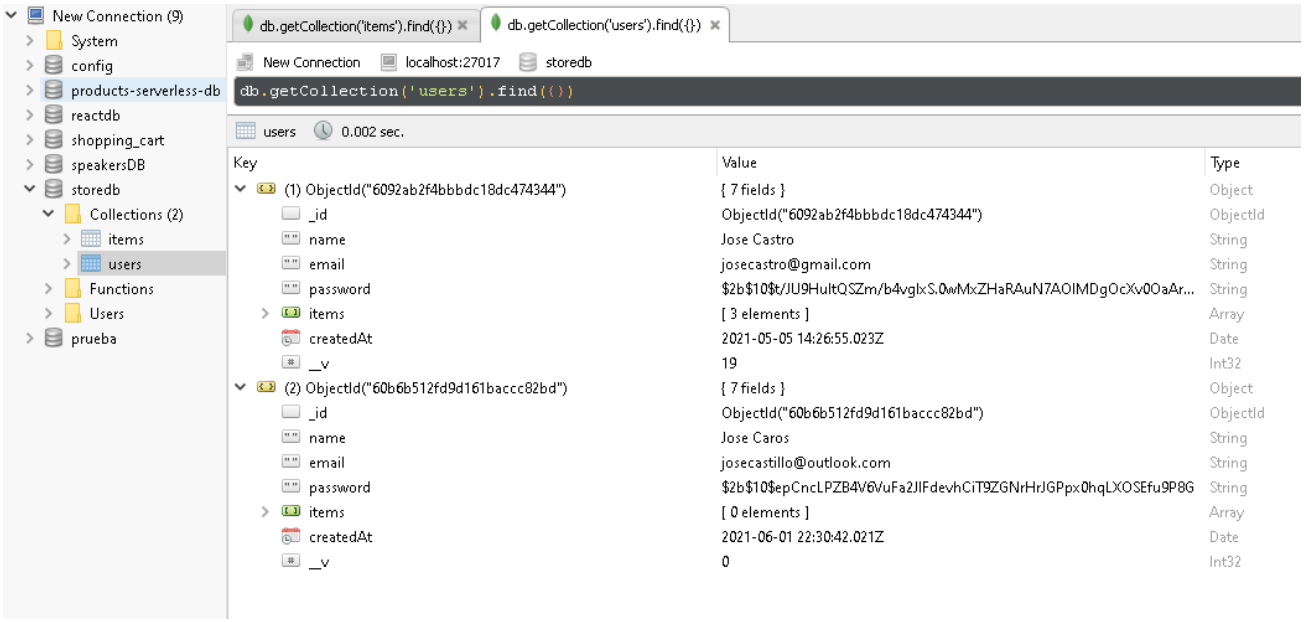
En esta parte se describen los entornos de desarrollo, lenguajes de programación, extensiones, plataformas y herramientas implementadas para el desarrollo de la aplicación:

- Node.js versión 14.17.0
- Npm versión 6.14.13
- MongoDB versión 4.4.4
- Mongodb versión 3.6.8
- Robo 3T versión 1.4.3
- Jsonwebtoken versión 8.5.1
- Mongoose versión 5.12.11
- Morgan 1.9.1
- Express versión 4.17.1
- Express validator versión 8.5.1
- Bcrypt versión 3.0.6
- Nodemon versión 1.19.1
- Basic auth versión 2.0.1

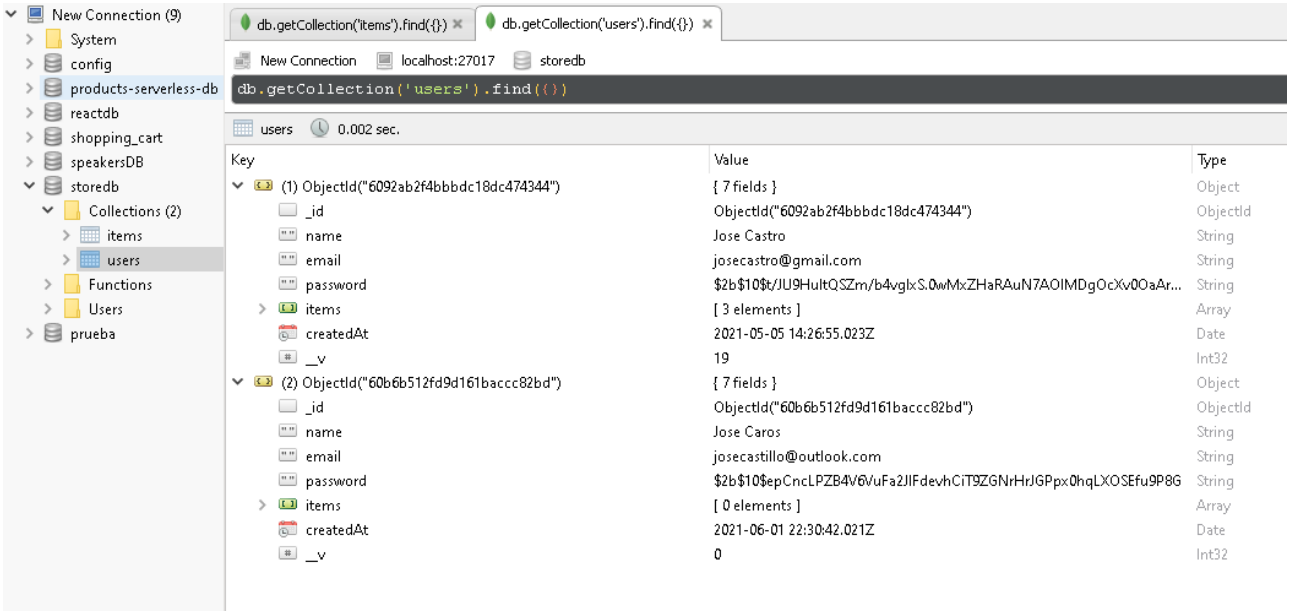
4. GESTIÓN DE LA BASE DE DATOS

Haciendo uso de robo 3T creamos nuestra nueva base de datos y establecemos el nombre (storedb) y el puerto, en nuestro caso dejamos el puerto establecido por defecto. Para nuestra aplicación fue necesaria la creación de dos colecciones, una en donde se almacenan los datos de los usuarios y otra en donde se almacenan los datos de los productos que serán reflejados en la vista del usuario, en Mongo los datos son guardados en documentos en lugar de registros, estos documentos son guardados en formato BSON que es una representación binaria del formato JSON, por otro lado en lugar de las clásicas tablas de SQL se implementan colecciones, este tipo de representación permite que no sea necesario seguir un esquema y como tal los documentos de una misma colección pueden tener esquemas diferentes

Usuarios



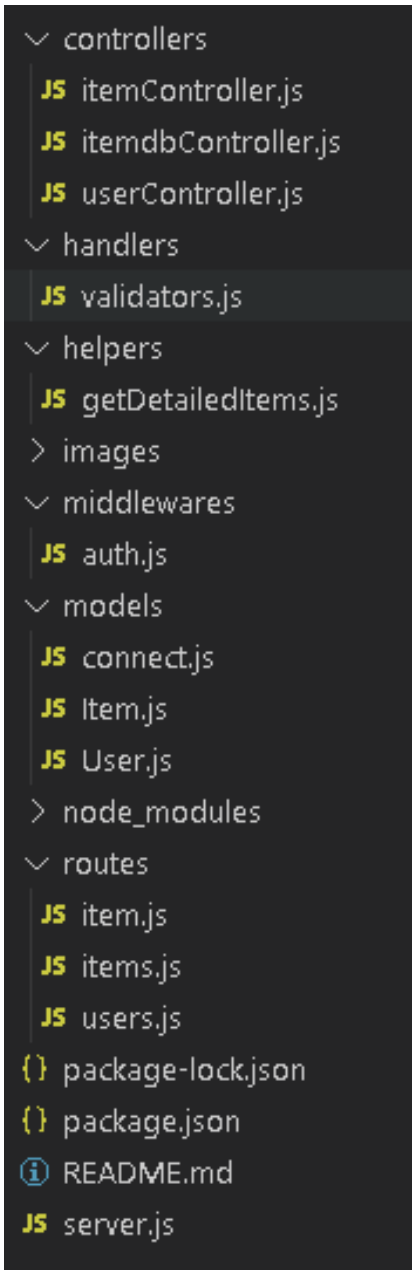
Productos



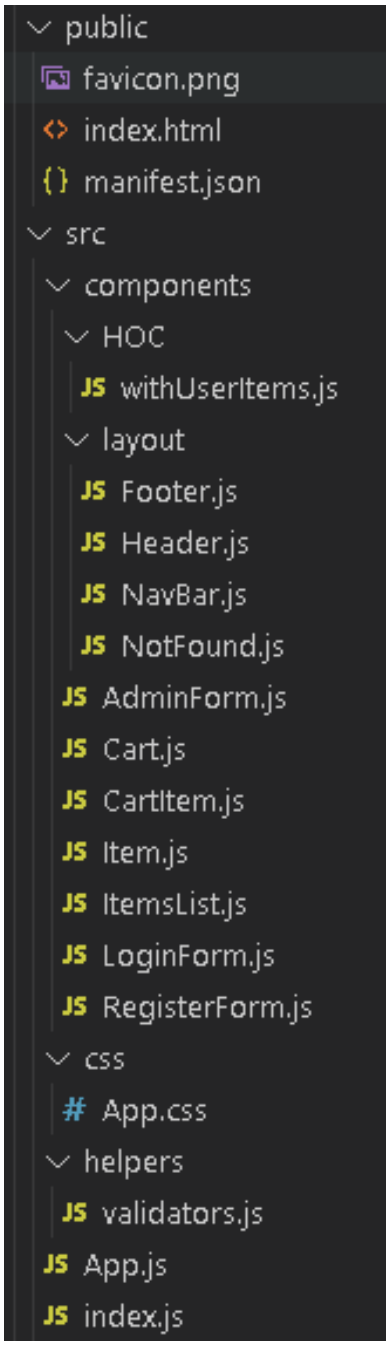
5. DESCRIPCIÓN DE LA APLICACIÓN

La aplicación está desarrollada en React siguiendo el patrón MVC (Modelo Vista Controlador) y el patrón de diseño interno de React Render props, este es un patrón de diseño para componentes de React que permite crear componentes reutilizables

Parte del servidor



Parte del cliente



En los siguientes puntos se explica un poco de las partes fundamentales para el funcionamiento de la aplicación

5.1. Modelo

Presenta el esquema principal de la conexión con la base de datos, además la representación de las características que tendrán los usuarios cuando estos se registren y las características que tendrán los productos en la base de datos.

Conexión con la base de datos

connect.js

models > connect.js > ...

```
const mongoose = require('mongoose');

mongoose
  .connect(
    process.env.MONGODB_URI || 'mongodb://localhost:27017/storedb',
    { useNewUrlParser: true, useCreateIndex: true }
  )
  .then(() => console.log('Connected to the DB.'))
  .catch(err => console.log(err));

```

Esquema de los productos

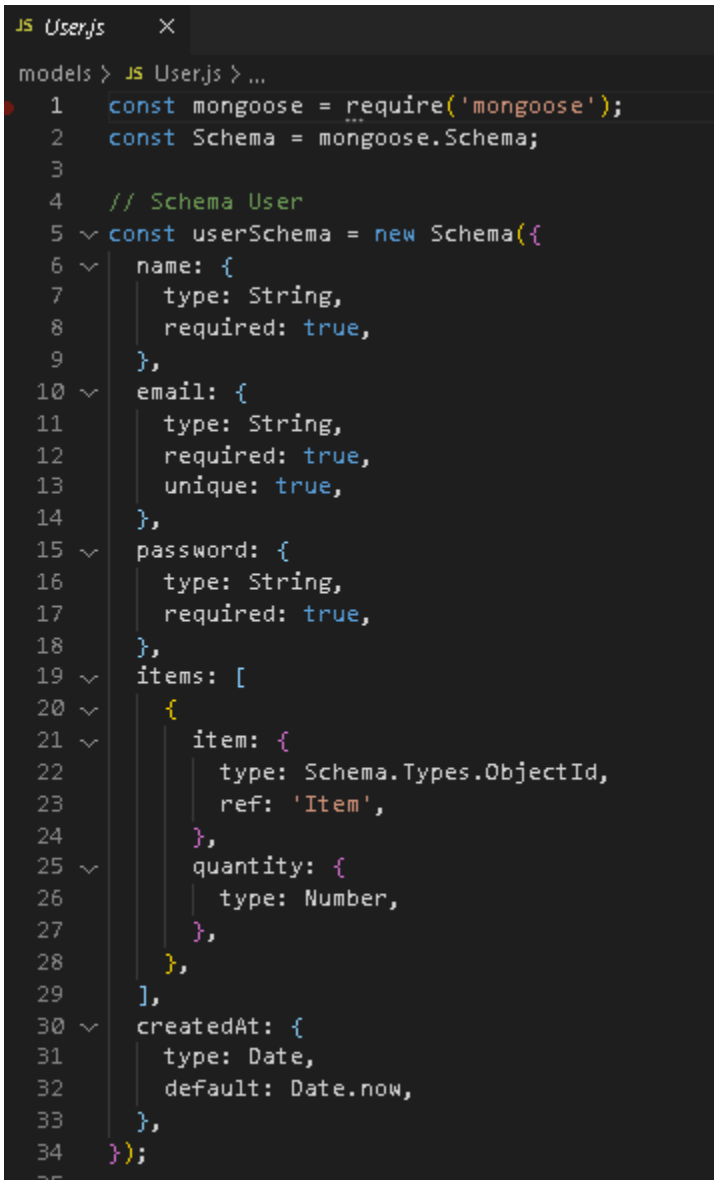
Item.js

models > Item.js > ...

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;

// Schema de los productos
const itemSchema = new Schema(
  {
    name: {
      type: String,
      required: true,
    },
    description: {
      type: String,
      default: '',
    },
    price: {
      type: Number,
      required: true,
    },
    imageUrl: {
      type: String,
    },
    addedAt: {
      type: Date,
      default: Date.now,
    },
  },
  {
    autoCreate: process.env.NODE_ENV !== 'production' ? true : false,
  }
);
```

Esquema de los usuarios



5.2 Vista

Representa la lógica que permite la vista final que tendrá el usuario en el navegador web, en este caso se definen los componentes de las diferentes rutas de la aplicación, como también aquellas partes que generen una mejor estética, barras de navegación, cabeceras entre otras. Todos los archivos que componen la vista se encuentran en la carpeta cliente del proyecto.

Vista de los productos

https://github.com/josecarloscas/Store-MERN-V2/blob/main/client/src/components/ItemsList.js

Vista del inicio de sesión

https://github.com/josecarloscas/Store-MERN-V2/blob/main/client/src/components/LoginForm.js

Vista del registro

https://github.com/josecarloscas/Store-MERN-V2/blob/main/client/src/components/RegisterForm.js

Vista del cart

https://github.com/josecarloscas/Store-MERN-V2/blob/main/client/src/components/Cart.js

Formulario para añadir productos

https://github.com/josecarloscas/Store-MERN-V2/blob/main/client/src/components/AdminForm.js

5.3 Controlador

Esta parte permite la comunicación entre los modelos y las vistas

Controlador del carrito

https://github.com/josecarloscas/Store-MERN-V2/blob/main/controllers/itemController.js

Controlador de los productos

https://github.com/josecarloscas/Store-MERN-V2/blob/main/controllers/itemdbController.js

Controlador de los usuarios

https://github.com/josecarloscas/Store-MERN-V2/blob/main/controllers/userController.js