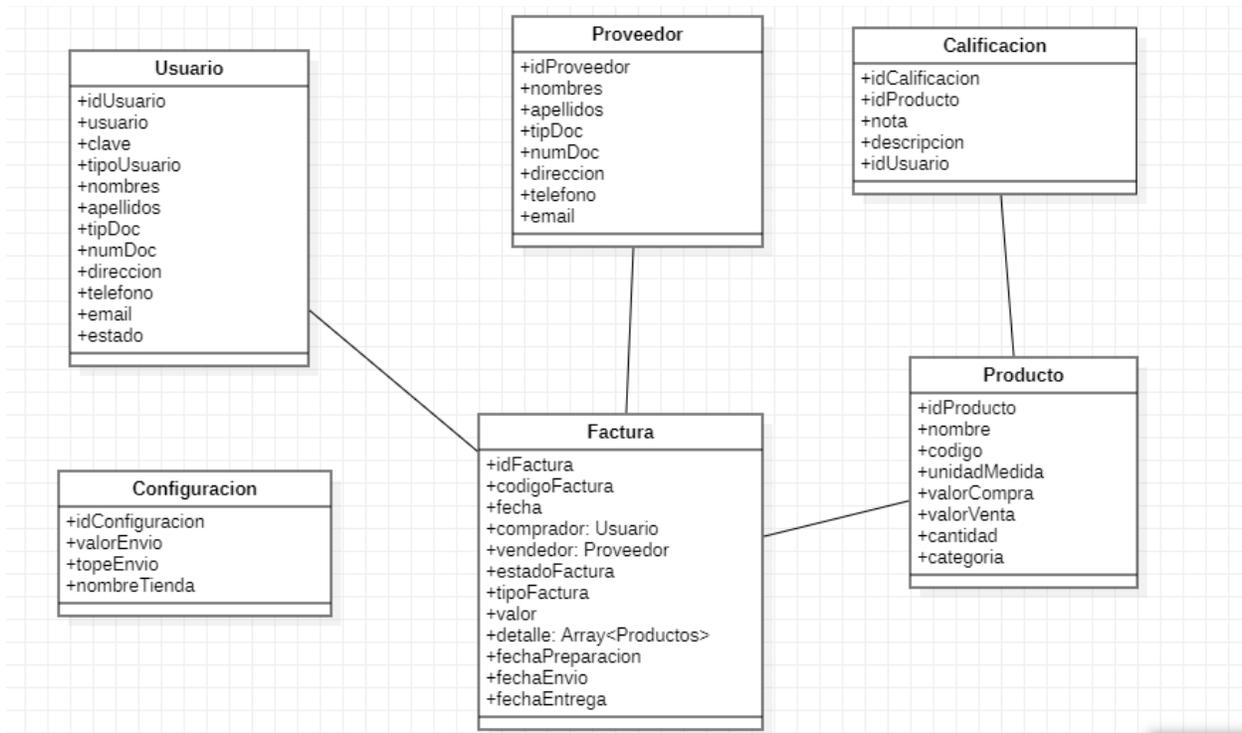


SPRINT 2 – DESARROLLANDO EL BACKEND

EQUIPO: U29_MiDulceOnline_10

DIAGRAMA DE COLECCIONES DE LA BASE DE DATOS



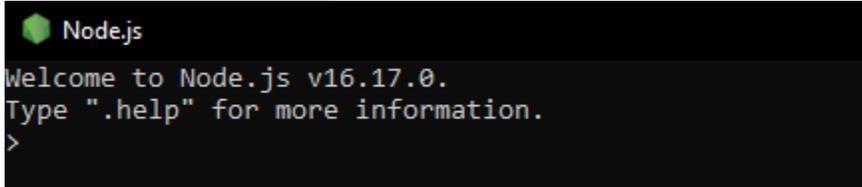
COMANDOS BASICOS DE LA ESTRUCTURA BACKEND Y DEPENDENCIAS REQUERIDAS

```
{
  "name": "backend",
  "version": "1.0.0",
  "description": "",
  "main": "app.js",
  "scripts": {
    "start": "nodemon app.js"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "cors": "^2.8.5",
    "express": "^4.18.2",
    "mongoose": "^6.6.7"
  },
  "devDependencies": {
    "nodemon": "^2.0.20"
  }
}
```

DOCUMENTACIÓN DEL PROCESO REALIZADO:

Descripción detallada del proceso realizado para el montaje del backend, incluyendo las dificultades presentadas.

a) Instalación de nodejs



```
Node.js
Welcome to Node.js v16.17.0.
Type ".help" for more information.
>
```

b) Configuración de Dependencias en VScode

Paso 1. Se Crea un directorio en el repositorio donde se llevan el proyecto, con su nombre en minúscula, separado por guiones.

Paso 2. Se abre la carpeta creada con el editor de código Vscod (Visual Studio Code)

```

EXPLORER
  OPEN EDITORS
    package.json
  CRUDTICS
    node_modules
    app.js
    package-lock.json
    package.json
  package.json
1 {
2   "name": "node-api-rest-example",
3   "version": "2.0.0",
4   "dependencies": {
5     "mongoose": "~3.6.11",
6     "express": "^4.7.1",
7     "method-override": "^2.1.2",
8     "body-parser": "^1.5.1"
9   }
10 }
11

```

Desde la terminal, se inicializa este paquete desde la terminal con el comando:
`npm init`

se crea automáticamente los paquetes json y se inicia la instalación de los paquetes necesarios para desarrollar el backend con los siguientes comandos:

- `npm install express --save`, se generará la carpeta `node_modules`, en la cual podrás almacenar los módulos necesarios para tu proyecto
- `npm install nodemon -D --save`
- `npm install cors --save`
- `npm install mongoose --save`
- modificar script `package.json` con el siguiente script para inicializar archivo `nodemon`:

```

"scripts": {
  "start": "nodemon app.js"
}

```

c) creación de la estructura base del backend, con las siguientes carpetas:

```

1  const mongoose = require("../DB/ConexionDB");
2
3
4  const usuarioSchema = mongoose.Schema({
5    usuario: {
6      type: String,
7      require: true
8    },
9    clave: {
10     type: String,
11     require: true
12   },
13   rol: {

```

- routers: En esta se especificarán cada uno de los manejadores de rutas montables y modulares.
- models: Especificarán los modelos donde se trabaja con los datos, por tanto, contendrá mecanismos para acceder a la información y también para actualizar su estado.
- controllers: funciona para obtener los datos solicitados de los modelos y crear la información necesaria para devolvérsela a los clientes en formato json.

d) **Creamos un archivo index.js** , app.js

creamos los archivos de index.js, el cual será nuestro archivo de punto de entrada inicial, en el cual tiene el siguiente código:

```

//Este archivo es el punto de entrada del proyecto
//Llamado a la app la cual se encuentra en el archivo app.js
var app = require("./app");
var mongoose = require("./conexBD/db");
var port = process.env.PORT || 4000; //Llamado al puerto por variables de
entorno y asignación de puerto en caso de no haberla

app.listen(port, () => { //Ejecución de la función que pone en ejecución la
aplicación en el puerto indicado
  console.log("servidor corriendo ok");
});

```

El archivo app.js es creado como archivo en donde se definen las importaciones

```
var express = require("express");
var cors = require("cors");
var app = express();

app.use(cors());
app.use(express.json());

app.use(require('./routers/routers'));

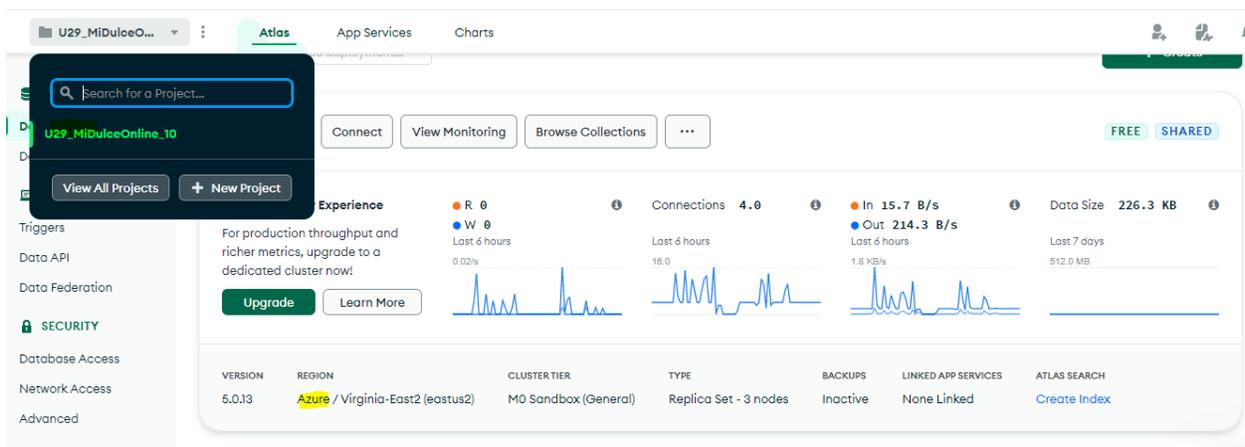
app.use(express.urlencoded({
  extended: true
}));

//Comando para indicar indicar condifacion de externo
app.use((req, res, next) => {
  res.header("Access-Control-Allow-Origin", "*");
  res.header(
    "Access-Control-Allow-Headers",
    "Authorization",
    "X-API-KEY",
    "Origin",
    "X-Requested-With",
    "Content-Type, Accept",
    "Access-Control-Allow-Request-Method"
  );
  res.header("Access-Control-Allow-Methods", "GET, POST, OPTIONS, PUT, DELETE");
  res.header("Allow", "GET, POST, OPTIONS, PUT, DELETE");
  next();
});

module.exports = app;
```

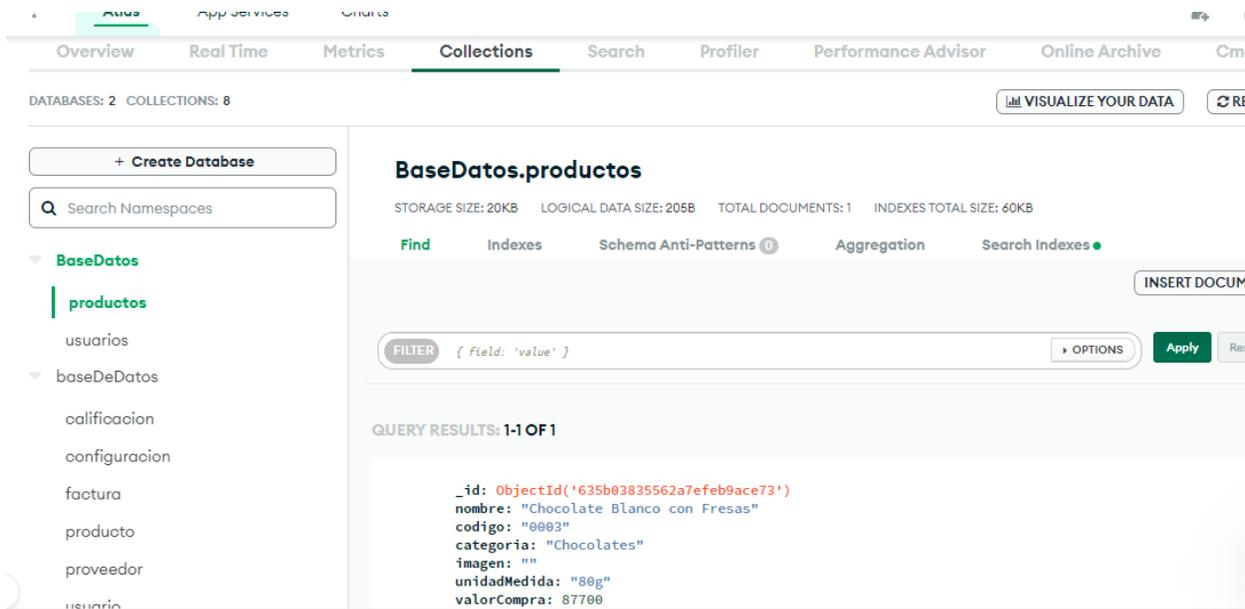
e) **Creación de colecciones y documentos desde la plataforma online de MongoDB:**

Creamos un usuario y agregamos el clúster en donde se implementará la base de datos del proyecto, para este caso se utilizó la tecnología de AZURE, se modificó el nombre del proyecto:



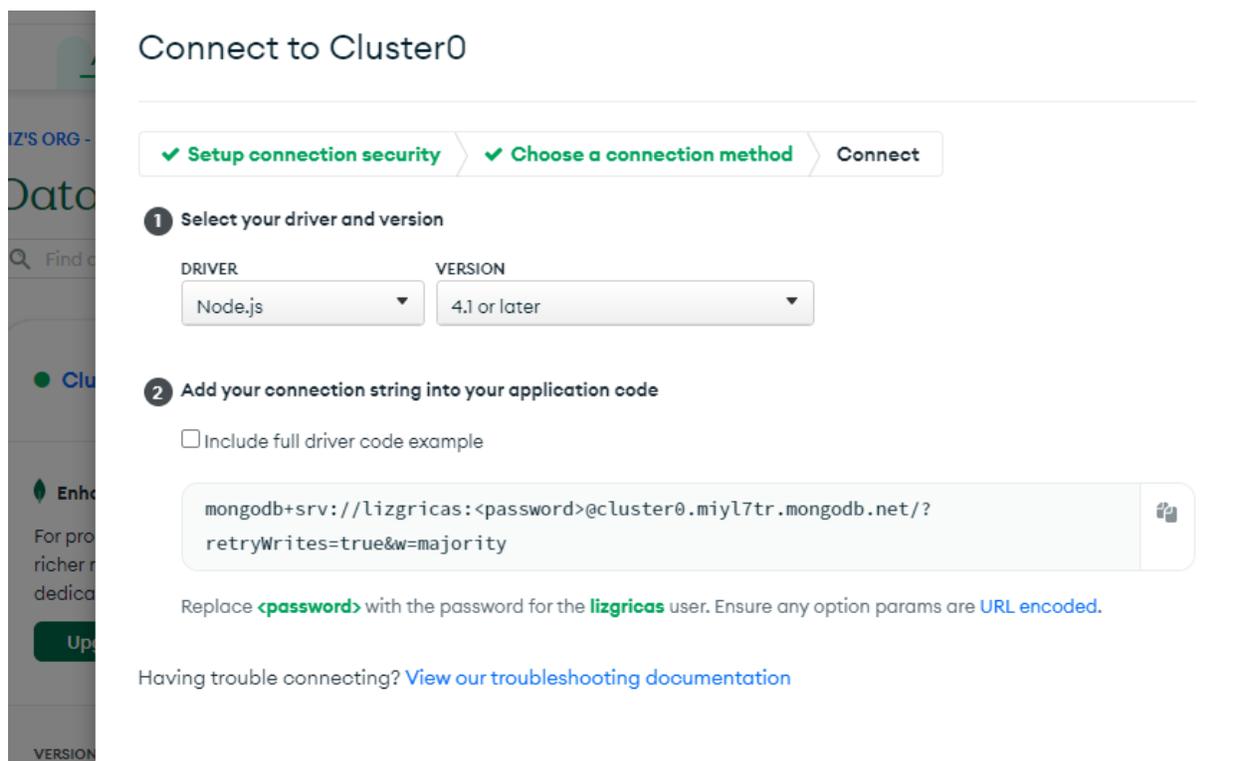
f) Creación de base de datos de prueba:

Creamos una base de datos de prueba directamente en la plataforma ingresando las colecciones de acuerdo al diagrama de clases creado en el sprint 1:



g) Conexión de Servidor y Base de datos:

Desde el archivo ConnectionDB.js, creado en VScode, se ejecutó la conexión con la cuenta creada en MongoDB, añadiendo el url arrojado por la plataforma en el ítem connect – connect your application; se modificó igualmente el password según los datos ingresados inicialmente en la creación del clúster:



Cuando probamos la conexión al clúster de MongoDB con el comando npm start desde la terminal, ésta arroja un error de autenticación fallida:

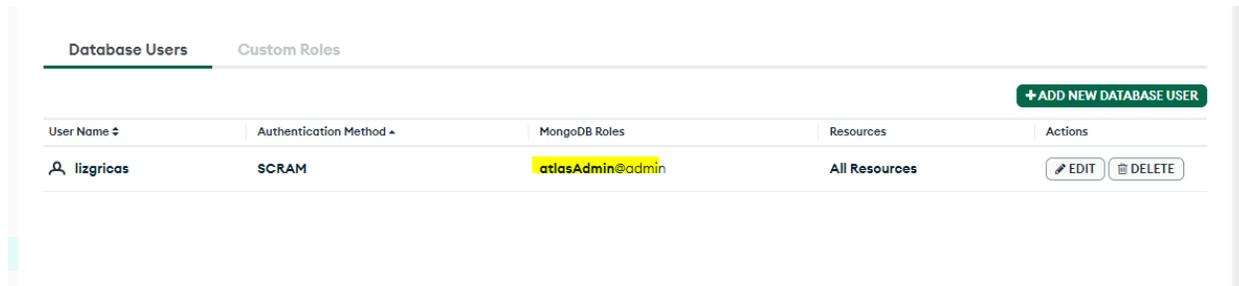
```

[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
MongoServerError: bad auth : Authentication failed.
    at Connection.onMessage (C:\Users\user\Desktop\MISION TIC\U29_MiDulceOnline_10\u29_midulceonline_10\node_modules\mongodb\lib\cmap\connection.js:207:30)
    at MessageStream.<anonymous> (C:\Users\user\Desktop\MISION TIC\U29_MiDulceOnline_10\u29_midulceonline_10\node_modules\mongodb\lib\cmap\connection.js:60:60)
    at MessageStream.emit (node:events:513:28)
    at processIncomingData (C:\Users\user\Desktop\MISION TIC\U29_MiDulceOnline_10\u29_midulceonline_10\node_modules\mongodb\lib\cmap\message_stream.js:132:20)
    at MessageStream._write (C:\Users\user\Desktop\MISION TIC\U29_MiDulceOnline_10\u29_midulceonline_10\node_modules\mongodb\lib\cmap\message_stream.js:33:9)
    at writeOrBuffer (node:internal/streams/writable:391:12)
    at _write (node:internal/streams/writable:332:10)
    at MessageStream.Writable.write (node:internal/streams/writable:336:10)
    at TLSSocket.ondata (node:internal/streams/readable:754:22)
    at TLSSocket.emit (node:events:513:28) {
  ok: 0,
  code: 8000,
  codeName: 'AtlasError',
  [Symbol(errorLabels)]: Set(1) { 'HandshakeError' }
}

```

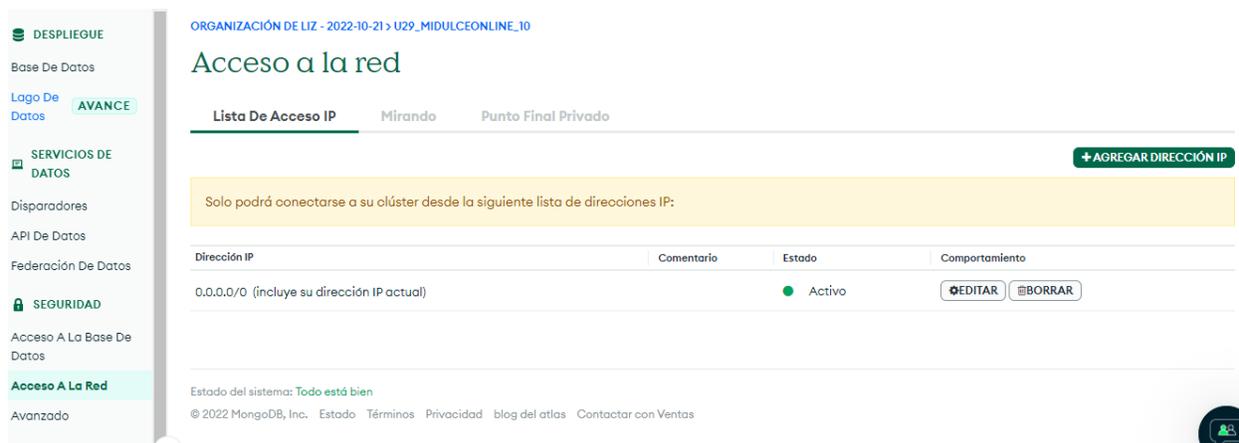
Para este inconveniente se debió realizar las siguientes modificaciones en la configuración de acceso del clúster en MongoDB:

Se modificó la clave (y se actualizó este dato en el archivo ConnectionDB.js) y el rol de usuario por administrador para tener todos los permisos:



User Name	Authentication Method	MongoDB Roles	Resources	Actions
lizgricas	SCRAM	atlasAdmin@admin	All Resources	EDIT DELETE

Se habilitó el acceso abierto a las IP que deseen conectarse al clúster, sin discriminar la lista de direcciones IP:

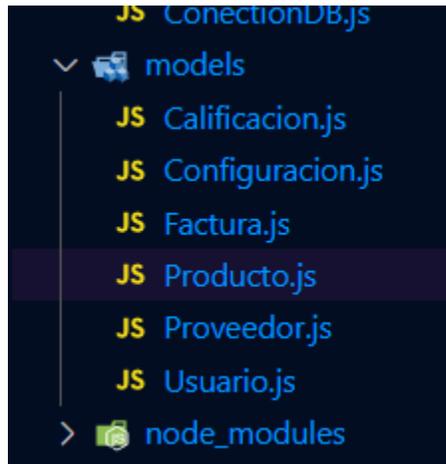


Dirección IP	Comentario	Estado	Comportamiento
0.0.0.0/0 (incluye su dirección IP actual)		Activo	EDITAR BORRAR

Se ejecutó nuevamente desde la terminal `npm start` y se estableció conexión con el clúster satisfactoriamente.

h) Creación de modelos de las colecciones requeridas en el proyecto:

se diseñó la representación lógica de las entidades (Tablas) en la carpeta models, agregando un archivo con extensión js por cada entidad requerida, es decir por producto se creó el archivo `Producto.js`, con la siguiente estructura, y así sucesivamente hasta completar todas las clases:



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

const productoSchema = new Schema({
  nombre: {
    type: String,
    required: true,
    unique: true
  },
  codigo: {
    type: String,
    required: true,
    unique: true
  },
  categoria: {
    type: String,
    required: true
  },
  imagen: {
    type: String,
  },
  unidadMedida: {
    type: String,
    required: true
  },
  descripcion: {
    type: String,
    required: true
  },
  valorCompra: {
```

```
type: Number,  
  required: true  
},  
valorVenta: {  
  type: Number,  
  required: true  
},  
sinRebaja: {  
  type: Number,  
  required: true  
},  
cantidad: {  
  type: Number,  
  required: true  
},  
calificacion: {  
  type: Number  
},  
proveedor: {  
  type: String,  
  required: true  
}  
},  
{  
  collection: 'productos',  
  versionKey: false  
}  
);  
  
var Producto = mongoose.model('productos', productoSchema);  
module.exports = Producto
```

i) **Creación de controladores:**

se crea la carpeta controller y se ingresa por cada clase un archivo de extensión js con la siguiente estructura, definiendo los métodos get, post, delete, update, patch que serán las rutas para las peticiones al backend según lo requiera cada entidad:

```
const Producto = require("../models/Producto");

function all(req, res) {
  Producto.find()
    .then((data) => {
      res.json(data);
    })
    .catch((err) => {
      res.json({ message: err })
    });
}

function search(req, res) {
  let hopa = req.params;
  Producto.findById(hopa.id)
    .then((data) => {
      res.json(data);
    })
    .catch((err) => {
      res.json({ message: err })
    });
}

function save(req, res) {
  const producto = new Producto(req.body);

  console.log(producto)
  producto.save()
    .then((data) => {
      res.json(data);
    })
    .catch((err) => {
      res.json({ message: err })
    });
}

function destroy(req, res) {
  Producto.deleteOne({ _id: req.params.id })
    .then((data) => {
      res.json(data);
    })
    .catch((err) => {
```



```

1  const express = require("express");
2  const usuariosCtrl = express.Router();
3  const Usuario = require("../models/Usuario"); //clase usuario
4
5  // este archivo cntrlador permite la creacion de funciones del backend
6  usuariosCtrl.get("/", (req,res)=> {
7    usuario.find()
8    .then((data) => {
9      res.json(data);
10   })
11   .catch((err) => {
12     res.json({ message: err })
13   });
14 });
15
16 //*****LISTAR USUARIO */

```

- se ejecuta nuevamente luego de realizar las modificaciones y se conecta a la base de datos satisfactoriamente, se procede a realizar las pruebas en el postman, de las colecciones creadas.

LISTADO DE LOS ENDPOINTS DEL PROYECTO:

El listado de los endpoints del backend, junto con algunas capturas de pantalla del funcionamiento de estos en Postman, se presentan a continuación:

MODELOS Y CONTROLADORES EN BACKEND DEL PROYECTO

```

1  const express = require('express');
2  const app = express();//este archivo llama a los modelos y controladores
3  const cors = require('cors'); // las cors son elementos que se utilizan para restringir el acceso
4  const bodyParser = require("body-parser");
5
6  const ctrlUsuarios = require("./controller/UsuariosCtrl");
7  /* const ctrlCalificaciones = require("./controller/CalificacionesCtrl");//llamamos el objeto
8  const ctrlConfiguraciones = require("./controller/ConfiguracionesCtrl");//importacion que permite
9  const ctrlFacturas = require("./controller/FacturasCtrl"); */
10 const ctrlProductos = require("./controller/ProductosCtrl");
11 /* const ctrlProveedores = require("./controller/ProveedoresCtrl"); */
12
13 app.use(express.json());
14 app.use(express.urlencoded({
15   extended: true

```

```

at next (C:\Users\user\Desktop\MISION TIC\U29_MiDulceOnline_10\U29_midulceonline_10\node_modules\express\lib\router\index.js:280:10)
at Function.handle (C:\Users\user\Desktop\MISION TIC\U29_MiDulceOnline_10\U29_midulceonline_10\node_modules\express\lib\router\index.js:175:3)
at router (C:\Users\user\Desktop\MISION TIC\U29_MiDulceOnline_10\U29_midulceonline_10\node_modules\express\lib\router\index.js:47:12)
[nodemon] restarting due to changes...
[nodemon] starting node app.js
Conectado con mongo

```

GET – POST COLECCIÓN USUARIOS

U29_MiDulceOnline_10 / GET-POST USUARIO

POST http://localhost:4000/usuarios/

```

1  {
2    "usuario": "jhonLT",
3    "clave": "j1234hTN",
4    "rol": "user",
5    "nombres": "Jhon",
6    "apellidos": "Lopez Tabares",
7    "tipDoc": "CC",
8    "numDoc": "43426730",
9    "direccion": "Calle 80F No. 24 -11",
10   "telefono": "456 38 25",
11   "email": "jholyT@gmail.com",

```

Status: 200 OK Time: 143 ms Size: 536 B

U29_MiDulceOnline_10 / GET-POST USUARIO

GET http://localhost:4000/usuarios/ Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```

1
2 ..... "usuario": "jhonLT",
3 ..... "clave": "j1234hTN",
4 ..... "rol": "user",
5 ..... "nombres": "Jhon",
6 ..... "apellidos": "Lopez Tabares",
7 ..... "tipDoc": "CC",

```

Body Cookies Headers (8) Test Results Status: 200 OK Time: 122 ms Size: 831 B Save Response

Pretty Raw Preview Visualize JSON Find

```

3 ..... "_id": "6359f3cc45668d34c9ec8097",
4 ..... "usuario": "lizgricas",
5 ..... "clave": "Lgc1234",
6 ..... "rol": "Administrador",
7 ..... "nombres": "Lizbeth",
8 ..... "apellidos": "Grisales Castro",
9 ..... "tipDoc": "CC",
10 ..... "numDoc": "1036626480",
11 ..... "direccion": "Calle 40 a sur No. 34b -3",
12 ..... "telefono": "3338802",
13 ..... "email": "lizgricas@gmail.com",

```

U29_MiDulceO... Atlas App Services Charts

DEPLOYMENT **Database** PREVIEW

- BaseDatos
 - productos
 - usuarios**
 - baseDeDatos

DATA SERVICES Preview

- Loggers
- API
- Federation

SECURITY

- Database Access
- Work Access
- Advanced

Find Indexes Schema Anti-Patterns Aggregation Search Indexes INSERT DOCUMENT

FILTER { field: 'value' } OPTIONS Apply Reset

```

_id: ObjectId('635af7b10613eb31e9deb7af')
usuario: "jhonLT"
clave: "j1234hTN"
rol: "user"
nombres: "Jhon"
apellidos: "Lopez Tabares"
tipDoc: "CC"
numDoc: "43426730"
direccion: "Calle 80F No. 24 -11"
telefono: "456 38 25"
email: "iholvT@email.com"

```

System Status: All Good

©2022 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales



```
localhost:4000/usuarios/
Extensión: Entrar al... PARA ESCUCHAR ... Educación Comfer

[
  {
    _id: "6359f3cc45668d34c9ec8097",
    usuario: "lizgricas",
    clave: "Lgc1234",
    rol: "Administrador",
    nombres: "Lizbeth",
    apellidos: "Grisales Castro",
    tipDoc: "CC",
    numDoc: "1036626480",
    direccion: "Carrer 40 a sur No. 34b -3",
    telefono: "3330802",
    email: "lizgricas@gmail.com",
    estado: "activo"
  },
  {
    _id: "635af7b10613eb31e9deb7af",
    usuario: "jhonLT",
    clave: "j1234hTN",
    rol: "user",
    nombres: "Jhon",
    apellidos: "Lopez Tabares",
    tipDoc: "CC",
    numDoc: "43426730",
    direccion: "Calle 80F No. 24 -11",
    telefono: "456 38 25",
    email: "jholyT@gmail.com",
    estado: "activo"
  }
]
```

GET – POST COLECCIÓN PRODUCTOS



Ciclo 4A

Workspaces ▾ Explore Sign In Create Account

New Import < GET POS DEL DELI PATCH Pz Prueb U29_M GET GET > + ... No Environment ▾

U29_MiDulceOnline_10 / GET-POST PRODUCTOS Save ...

GET http://localhost:4000/productos/ Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON ▾

```
5  ...."imagen": "",
6  ...."unidadMedida": "80g",
7  ...."valorCompra": "87700",
8  ...."valorVenta": "95000",
9  ...."cantidad": "20",
10  ...."calificacion": "5"
11
```

Body Cookies Headers (8) Test Results Status: 200 OK Time: 205 ms Size: 267 B Save Response ▾

Pretty Raw Preview Visualize JSON ▾

1

Home Workspaces Explore Search Postman Sign In Create Account

U29_MiDulceOnline_10 / GET-POST PRODUCTOS

POST http://localhost:4000/productos/ Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```

5 ..... "imagen": "",
6 ..... "unidadMedida": "80g",
7 ..... "valorCompra": "87700",
8 ..... "valorVenta": "95000",
9 ..... "cantidad": "20",
10 ..... "calificacion": "5"
11

```

Body Cookies Headers (8) Test Results Status: 200 OK Time: 144 ms Size: 485 B Save Response

Pretty Raw Preview Visualize JSON

```

1
2 "nombre": "Chocolate Blanco con Fresas",
3 "codigo": "0003",
4 "categoria": "Chocolates",
5 "imagen": "",
6 "unidadMedida": "80g",
7 "valorCompra": "87700",
8 "valorVenta": "95000",
9 "cantidad": "20",
10 "calificacion": "5",
11 "_id": "635b03835562a7efeb9ace73"

```

U29_MiDulceO... Atlas App Services Charts

DEPLOYMENT Database Data Lake DATA SERVICES Triggers Data API Data Federation SECURITY Database Access Network Access Advanced

BaseDatos productos usuarios baseDeDatos

Find Indexes Schema Anti-Patterns Aggregation Search Indexes

INSERT DOCUMENT

FILTER { field: 'value' } OPTIONS Apply Reset

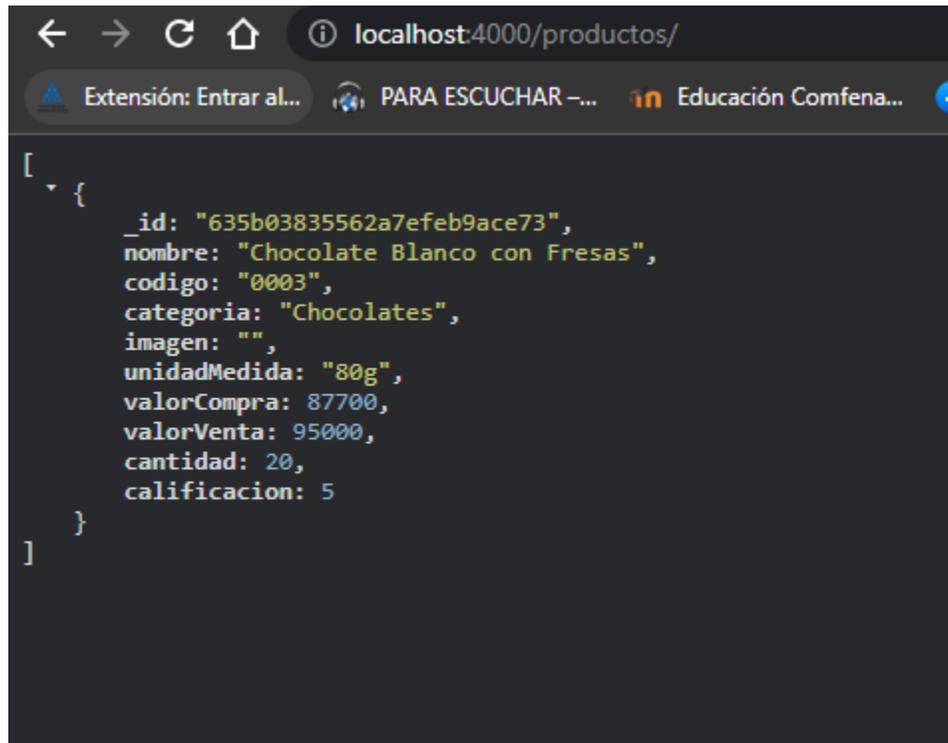
QUERY RESULTS: 1-1 OF 1

```

_id: ObjectId('635b03835562a7efeb9ace73')
nombre: "Chocolate Blanco con Fresas"
codigo: "0003"
categoria: "Chocolates"
imagen: ""
unidadMedida: "80g"
valorCompra: 87700
valorVenta: 95000
cantidad: 20
calificacion: 5

```

System Status: All Good ©2022 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales



A screenshot of a web browser displaying a JSON response. The address bar shows 'localhost:4000/productos/'. The JSON data is as follows:

```
[
  {
    _id: "635b03835562a7efeb9ace73",
    nombre: "Chocolate Blanco con Fresas",
    codigo: "0003",
    categoria: "Chocolates",
    imagen: "",
    unidadMedida: "80g",
    valorCompra: 87700,
    valorVenta: 95000,
    cantidad: 20,
    calificacion: 5
  }
]
```

EVIDENCIAS DE USO DE JIRA:

Capturas de pantalla del uso de Jira y de historias de usuario

Jira Software Tu trabajo Proyectos Filtros Paneles Personas Aplicaciones Crear

U29_MiDulceOnline_10 Proyecto de software

PLANIFICACIÓN

- Hoja de ruta
- Backlog**
- Tablero

DESARROLLO

- Código
- Páginas de proyectos
- Añadir acceso rápido
- Configuración del proyecto

Estás en un proyecto gestionado por el equipo Más información

Proyectos / U29_MiDulceOnline_10

Backlog

LC SE J OC HS Epic Tipo Insights

- U2910-32 Diseñar el modelo lógico DESARROLLO DE BASE DE DATOS FINALIZADA LC
- U2910-33 Desarrollar el modelo físico DESARROLLO DE BASE DE DATOS FINALIZADA J
- U2910-34 Pruebas de Funcionamiento ANALISIS Y PLANIFICACIÓN DEL P... FINALIZADA SE
- U2910-39 Back End RQF 01-HU 2 DESARROLLO FRONTED Y BACKEND FINALIZADA J
- U2910-36 Back End RQF 01-HU 1 DESARROLLO FRONTED Y BACKEND FINALIZADA HS
- U2910-11 Como administrador del sistema quiero ver los productos, sus precios y sus cantidad... FINALIZADA OC
- U2910-12 Como operario quiero registrar ingresos de productos junto con su precio y cantidad, ... FINALIZADA J
- U2910-13 Como operario quiero registrar salidas de productos junto con su precio y cantidad, q... FINALIZADA SE
- U2910-14 Como administrador del sistema quiero visualizar la hora y fecha de cada transa... TAREAS POR HACER LC

Jira Software Tu trabajo Proyectos Filtros Paneles Personas Aplicaciones Crear

U29_MiDulceOnline_10 Proyecto de software

PLANIFICACIÓN

- Hoja de ruta**
- Backlog
- Tablero

DESARROLLO

- Código
- Páginas de proyectos
- Añadir acceso rápido
- Configuración del proyecto

Estás en un proyecto gestionado por el equipo Más información

Proyectos / U29_MiDulceOnline_10

Hoja de ruta

Dar feedback Compartir Exportar

LC SE J HS +2 Categoría de est...

	SEP	OCT - DIC	ENE - MAR '23
Sprints			
U2910-72 Analisis y Planificación del Proyecto			
U2910-73 Desarrollo de Base de datos			
U2910-74 Desarrollo Fronted y Backend			
U2910-75 Plan de Pruebas, Despliegue y Documentación			
+ Crear Epic			

Hoy Semanas Meses Trimest...

The screenshot shows the Visual Studio Code interface for a Node.js project named 'U29_MIDULCEONLINE_10'. The Explorer sidebar on the left shows the project structure with folders for 'controller', 'DB', 'models', 'resources', and 'routers'. The main editor displays the 'app.js' file with the following JavaScript code:

```
1 const express = require('express');
2 const app = express();//este archivo llama a Los modelos y controladores
3 const cors = require('cors'); // las cors son elementos que se utilizan para restringir el a
4 const bodyParser = require("body-parser");
5
6 const ctrlUsuarios = require("./controller/UsuariosCtrl");
7 /* const ctrlCalificaciones = require("./controller/CalificacionesCtrl");//llamamos el objeto
8 const ctrlConfiguraciones = require("./controller/ConfiguracionesCtrl");//importacion que pe
9 const ctrlFacturas = require("./controller/FacturasCtrl"); */
10 const ctrlProductos = require("./controller/ProductosCtrl");
11 /* const ctrlProveedores = require("./controller/ProveedoresCtrl"); */
12
13 app.use(express.json());
14 app.use(express.urlencoded({
15   extended: true
16 }));
```

The terminal window at the bottom shows the following output:

```
at next (C:\Users\user\Desktop\MISION TIC\U29_MiDulceOnline_10\u29_midulceonline_10\node_modules\express\lib\router\index.js:280:10)
at Function.handle (C:\Users\user\Desktop\MISION TIC\U29_MiDulceOnline_10\u29_midulceonline_10\node_modules\express\lib\router\index.js:175:3)
at router (C:\Users\user\Desktop\MISION TIC\U29_MiDulceOnline_10\u29_midulceonline_10\node_modules\express\lib\router\index.js:47:12)
[nodemon] restarting due to changes...
[nodemon] starting node app.js
Conectado con mongo
```