

«Talento Tech»

Desarrollo Web 4

Clase 06





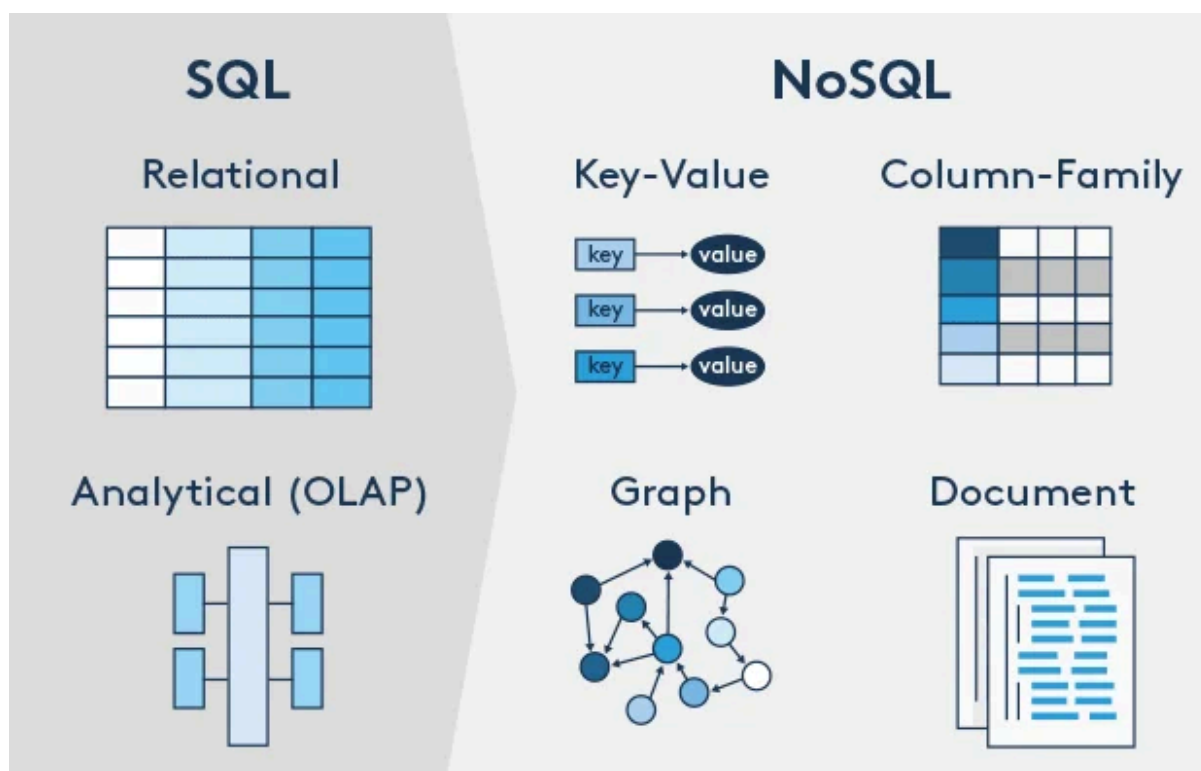
Clase N° 6 | Bases de datos No-SQL

Temario:

- Bases de Datos No sql (Mongo DB)
- Introducción: Mongo Db y su ODM (Mongoose)
- Introducción a Mongo Atlas y creación de una Cuenta
- Interactuar con Express.js y Mongo Atlas



Bases de Datos No SQL (Mongo Db)



En el mundo de la informática, las bases de datos son esenciales para almacenar y organizar datos de manera eficiente.

¿Qué son las Bases de Datos No Relacionales?

Las Bases de Datos No Relacionales o NoSQL (Not Only SQL), rompen con la estructura tradicional basada en tablas que encuentras en las bases de datos relacionales como las que utilizan **SQL**. En lugar de ello, adoptan un enfoque más flexible, permitiendo almacenar datos de diversas maneras, como pares clave-valor, documentos, columnas anchas o incluso grafos.

Diferencias entre Bases de Datos Relacionales (SQL) y las Bases de Datos No Relacionales (NoSQL)

La principal diferencia radica en cómo manejan la estructura y organización de los datos:

- **Lenguaje:** SQL utiliza un lenguaje estructurado y requiere esquemas definidos de antemano, mientras que NoSQL ofrece esquemas dinámicos y mayor flexibilidad.
- **Escalabilidad:** SQL escala verticalmente, aumentando la capacidad de un solo servidor, mientras que NoSQL escala horizontalmente, permitiendo manejar más tráfico al agregar más servidores.
- **Comunidad:** SQL cuenta con una comunidad establecida y amplia, mientras que NoSQL está en constante crecimiento y desarrollo.

Relevancia de las Bases de Datos No Relaciones

Las Bases de Datos No Relacionales son fundamentales en el mundo digital actual. Proporcionan soluciones efectivas para manejar grandes volúmenes de datos, permitiendo escalabilidad, flexibilidad y un rendimiento excepcional. Desde aplicaciones móviles hasta sistemas de gestión de contenido, NoSQL se ha convertido en una herramienta esencial en el kit del desarrollador.

En esta clase, exploramos cómo estas bases de datos revolucionan la forma en que almacenamos y accedemos a la información.

MongoDB: una base de datos NoSQL orientada a documentos

MongoDB es parte de la familia de bases de datos **NoSQL**, lo que significa que adopta un enfoque diferente a las bases de datos tradicionales basadas en **SQL**. Estas bases de datos NoSQL están diseñadas para manejar datos de manera más flexible y escalable.

Introducción: MongoDB y su ODM (Mongoose)

¿Qué hace a MongoDB especial?

- **Estructura Flexible**
MongoDB almacena los datos en documentos tipo JSON, en lugar de tablas, permitiendo una estructura más flexible. Esto significa que cada entrada de datos puede tener su propio formato único.
- **Escala Fácilmente**
La escalabilidad es clave en MongoDB. Puede manejar grandes volúmenes de datos y tráfico simplemente añadiendo más servidores, proporcionando un rendimiento excepcional.
- **JavaScript en el Núcleo**
MongoDB utiliza JavaScript como su lenguaje de consulta, facilitando su integración con aplicaciones web.

¿Cómo interactuamos con MongoDB?

Para aplicaciones más complejas, utilizamos **controladores o drivers de MongoDB** en lenguajes de programación como JavaScript (Node.js), Python, Java, etc. Estos controladores nos permiten interactuar con MongoDB desde nuestras aplicaciones.

ORM (Object-Relational Mapping): Comprendiendo la Interacción entre Objetos y Bases de Datos Relacionales

ORM o Mapeo Objeto-Relacional es un concepto que simplifica la interacción entre aplicaciones escritas en lenguajes de programación orientados a objetos y bases de datos relacionales. La idea principal es que las clases y objetos en el código se asignen directamente a tablas y filas en la base de datos, eliminando la necesidad de escribir consultas SQL directas. En lugar de lidiar con conjuntos de registros y tablas, los desarrolladores pueden trabajar con objetos y clases, facilitando el desarrollo y el mantenimiento del código.



Mongoose: ODM para MongoDB

En MongoDB, el concepto similar a ORM se llama ODM (Mapeo de Documento-Objeto), y Mongoose es una biblioteca popular que actúa como ODM.

Características de ODM y Mongoose en Mongo DB

Esquemas y Modelos

En lugar de definir tablas, se crean esquemas que describen la forma de los documentos. Mongoose proporciona modelos basados en estos esquemas.

Operaciones CRUD Simplificadas

Ofrece métodos sencillos para realizar operaciones de creación, lectura, actualización y eliminación en la base de datos.

Validaciones y Métodos Personalizados

Permite agregar validaciones y métodos personalizados directamente en el esquema. Relaciones Estructuradas (aunque MongoDB es flexible)

Middleware

Proporciona middleware para ejecutar funciones antes o después de eventos específicos, como guardar un documento.

En resumen, Mongoose simplifica la interacción con MongoDB al proporcionar una capa de abstracción que utiliza esquemas y modelos, facilitando operaciones de base de datos y ofreciendo características similares a un ORM en un entorno NoSQL.

Introducción a Mongo Atlas y Creación de una Cuenta

Introducción:

MongoDB Atlas es un servicio en la nube que proporciona una plataforma totalmente gestionada para implementar, escalar y administrar bases de datos MongoDB sin la necesidad de configurar infraestructura. Ofrece características como escalabilidad automática, seguridad avanzada, alta disponibilidad y compatibilidad con múltiples



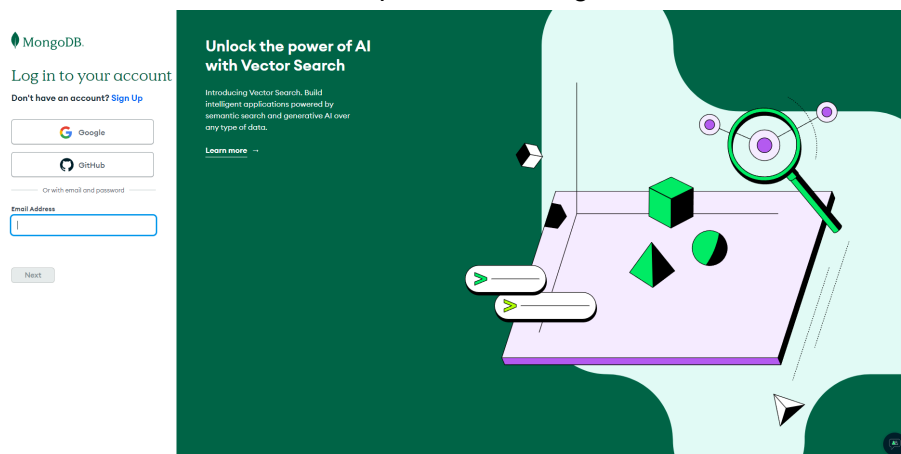
proveedores de nube, lo que permite a los desarrolladores centrarse en la construcción de aplicaciones sin preocuparse por la gestión de la base de datos subyacente.

Paso 0

Para poder crear una cuenta en mongo atlas debemos ir a la página de la misma:

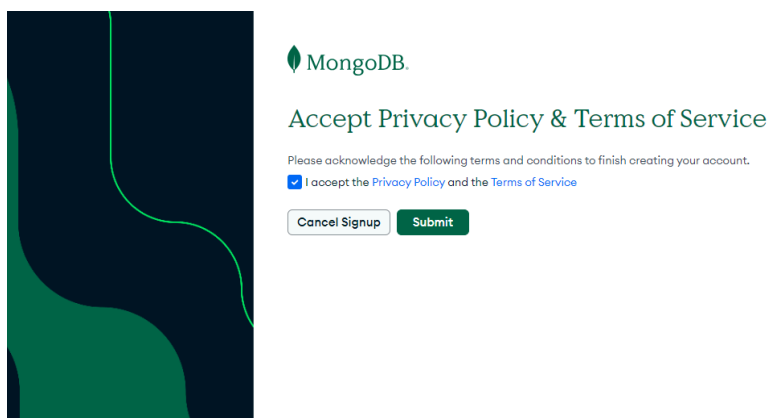
<https://account.mongodb.com/account/login>

Luego deberemos crearnos una cuenta, podemos usar gmail :



Paso 1

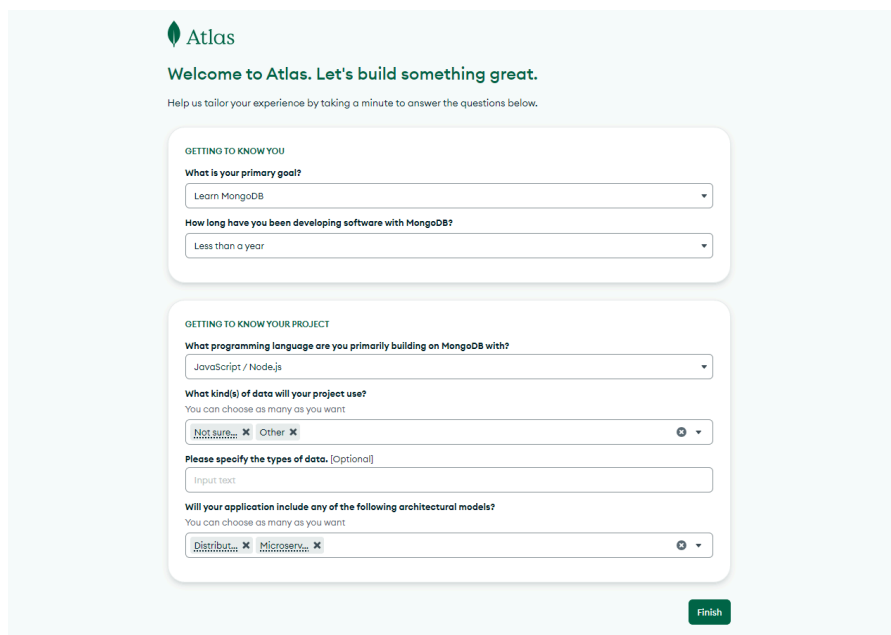
Aceptamos los términos de privacidad.





Paso 2

Seleccionamos preferencias de usuarios.



Atlas
Welcome to Atlas. Let's build something great.
Help us tailor your experience by taking a minute to answer the questions below.

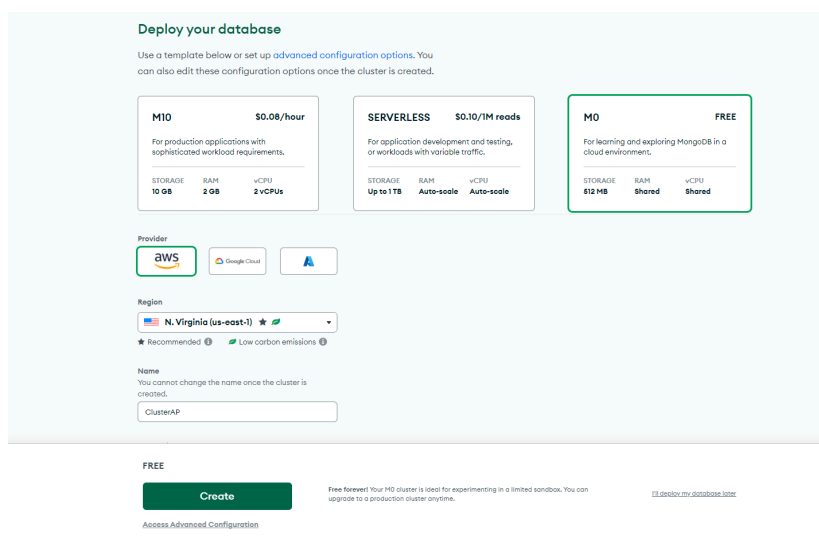
GETTING TO KNOW YOU
What is your primary goal?
Learn MongoDB
How long have you been developing software with MongoDB?
Less than a year

GETTING TO KNOW YOUR PROJECT
What programming language are you primarily building on MongoDB with?
JavaScript / Node.js
What kind(s) of data will your project use?
You can choose as many as you want.
Not sure... Other
Please specify the types of data. [Optional]
Input text
Will your application include any of the following architectural models?
You can choose as many as you want.
Distributed Microservices

Finish

Paso 3

Seleccionamos el plan y el proveedor.



Deploy your database
Use a template below or set up [advanced configuration options](#). You can also edit these configuration options once the cluster is created.

Plan	Price
M10 For production applications with sophisticated workload requirements.	\$0.08/hour
SERVERLESS For application development and testing, or workloads with variable traffic.	\$0.10/1M reads
M0 For learning and exploring MongoDB in a cloud environment.	FREE

Provider
☒ AWS
 ☐ Google Cloud
 ☐ Azure

Region
☒ N. Virginia (us-east-1)
 ☐ Low carbon emissions

Name
 You cannot change the name once the cluster is created.
 ClusterAP

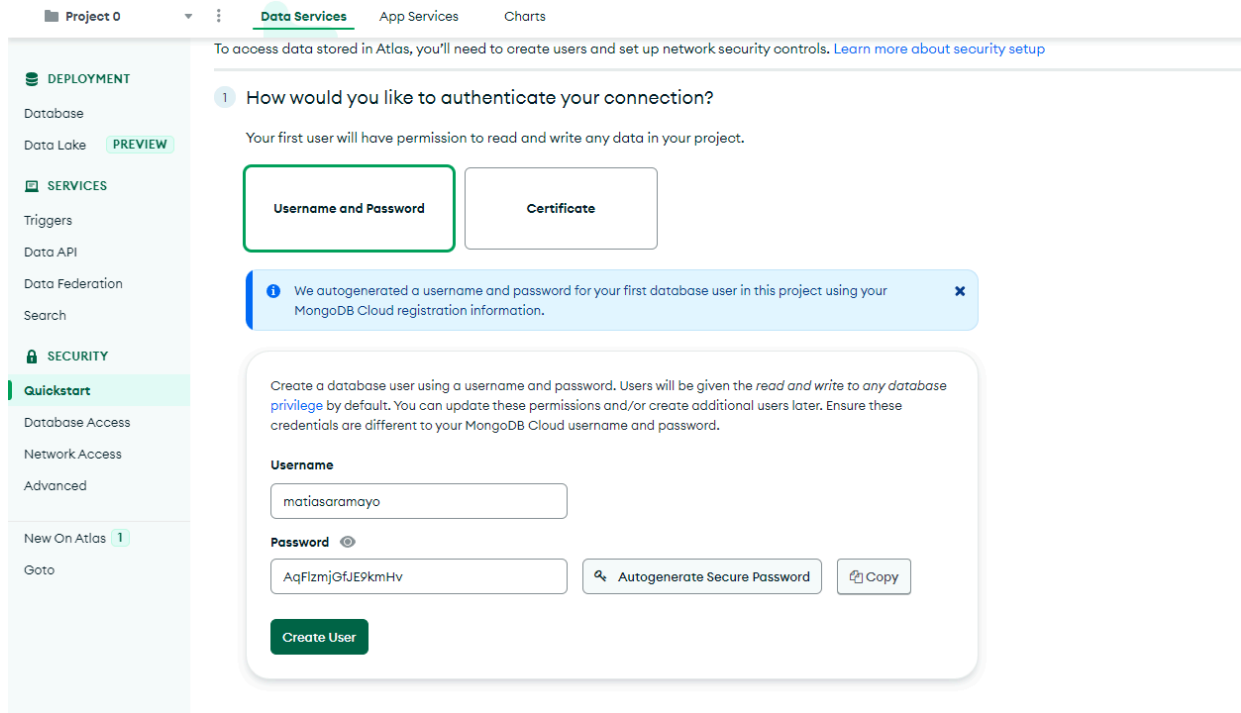
FREE
Create
 Free forever! Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

[Access Advanced Configuration](#)



Paso 4

Luego nos va a aparecer una vista tipo dashboard en la cual nos va a pedir que creamos un usuario.



Project 0 ▾ Data Services App Services Charts

To access data stored in Atlas, you'll need to create users and set up network security controls. [Learn more about security setup](#)

1 How would you like to authenticate your connection?

Your first user will have permission to read and write any data in your project.

Username and Password Certificate

i We autogenerated a username and password for your first database user in this project using your MongoDB Cloud registration information. **x**

Create a database user using a username and password. Users will be given the *read and write to any database privilege* by default. You can update these permissions and/or create additional users later. Ensure these credentials are different to your MongoDB Cloud username and password.

Username

matiasaramayo

Password **👁**

AqFlzmjGfJE9kmHv **🔍 Autogenerate Secure Password** **📋 Copy**

Create User

Paso 5

Le damos a crear usuario y por último vamos al final de la página y apretamos "Finish and Close". *Consejo: Recuerden la password que colocaron por que después Mongoose se las va a pedir para hacer la conexión.



info We added your current IP address. You can connect to your cluster locally from this device.

Add entries to your IP Access List

Only an IP address you add to your Access List will be able to connect to your project's clusters. You can manage existing IP entries via the [Network Access Page](#).

IP Address	Description
<input type="text" value="Enter IP Address"/>	<input type="text" value="Enter description"/>
<input type="button" value="Add My Current IP Address"/>	
<input type="button" value="Add Entry"/>	
This IP address has already been added.	

IP Access List	Description
181.105.83.109/32	My IP Address

Paso 6

Luego nos va a aparecer una ventana modal donde ya nos permite poder acceder a nuestra base de datos.

connect to your project's clusters. You can manage

Congratulations on setting up access rules!

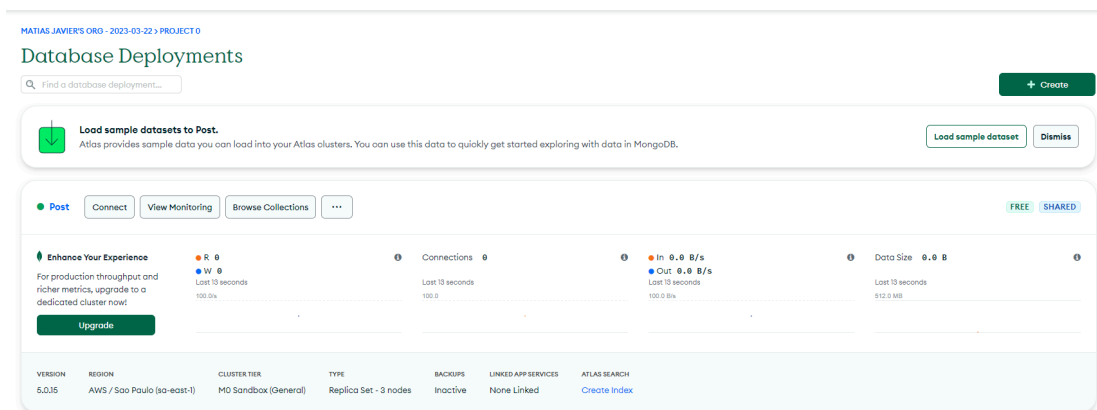
You will now be able to connect to your deployments. You can continue to add and update access rules in [Database Access](#) and [Network Access](#).

☒ Hide Quickstart guide in the navigation. You can visit [Project Settings](#) to access it in the future.



Paso 7

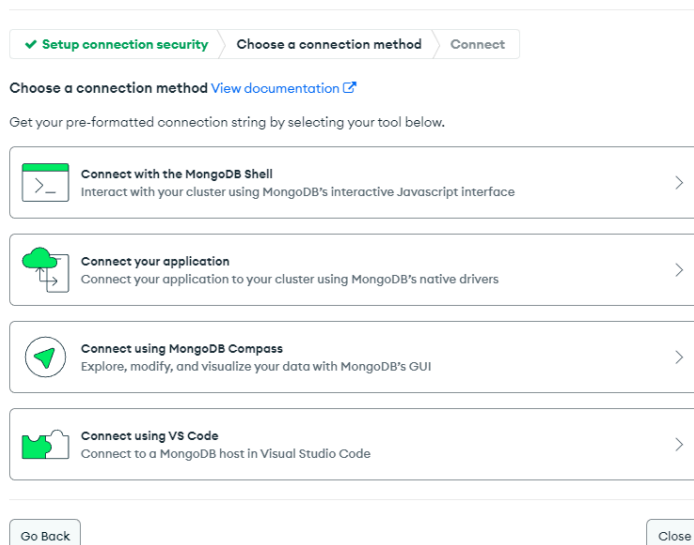
Una vez que cargue toda la información en el dashboard que nos provee Mongoose vamos a ir a “Connect”.



Paso 8

Luego hacemos clic en “Connect your application”.

Connect to Post





Paso 9

En el recuadro vamos a copiar la clave de que nos provee Mongoose para realizar la conexión con la DB
Para finalizar, hacemos click en

Connect to Post

✓ Setup connection security

✓ Choose a connection method

Connect

1 Select your driver and version

DRIVER

Node.js

VERSION

4.1 or later

2 Add your connection string into your application code

☐ Include full driver code example

```
mongodb+srv://matiasaramayo:<password>@post.go6mjdc.mongodb.net/?
retryWrites=true&w=majority
```

Replace **<password>** with the password for the **matiasaramayo** user. Ensure any option params are [URL encoded](#).

Having trouble connecting? [View our troubleshooting documentation](#)

Go Back

Close

Listo! ¡Ya tenés tu usuario en Mongo Atlas!

Interactuar con Express.js y Mongo Atlas

A continuación, mostramos un código como ejemplo de una aplicación de servidor utilizando Express y Mongoose, que se conecta a una base de datos MongoDB para



realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en una colección llamada "productos".

Hagamos el desglose línea por línea:

- **Importar dependencias**

Hasta ahora lo único que no hemos visto cómo instalar es Mongoose

```
npm i mongoose
```

Se importan las bibliotecas necesarias: Eexpress para el servidor web, Mmongoose para la interacción con MongoDB, Ddotenv para cargar variables de entorno desde un archivo .env, y Bbody-Parser para analizar los cuerpos de las solicitudes HTTP.

```
const express = require('express');
const mongoose = require('mongoose');
require('dotenv').config();
const bodyParser = require('body-parser');
const app = express();
const PORT = 3000;
```

- **Configurar middleware**

Se configuran los middlewares de body-parser para analizar los cuerpos de las solicitudes en formato urlencoded y json.

```
app.use(bodyParser.urlencoded({ extended: true }));
app.use(bodyParser.json());
```

- **Conectar a la base de datos MongoDB**

Se establece la conexión a la base de datos MongoDB utilizando la URL especificada en la variable de entorno DB

```
mongoose.connect(process.env.DB, {  
  useNewUrlParser: true,  
  useUnifiedTopology: true  
});
```

- Definir el esquema del producto y el modelo

Se define un esquema para los productos con tres campos: nombre, precio, y descripción. Luego, se crea un modelo llamado “Producto” basado en ese esquema.

```
const productoSchema = new mongoose.Schema({  
  nombre: String,  
  precio: Number,  
  descripción: String  
});  
  
const Producto = mongoose.model('Producto', productoSchema);
```

- Rutas para operaciones CRUD

Se definen las rutas para realizar operaciones CRUD en la colección de productos utilizando funciones asíncronas y el modelo Producto.

```
app.get('/productos', async (req, res) => {  
  // Operación para obtener todos los productos  
});  
  
app.post('/productos', async (req, res) => {  
  // Operación para agregar un nuevo producto  
});  
  
app.put('/productos/:id', async (req, res) => {  
  // Operación para actualizar un producto por ID  
});
```

```
});

app.delete('/productos/:id', async (req, res) => {
  // Operación para eliminar un producto por ID
});
```

- **Funciones para operaciones CRUD**

Se implementan las funciones asincrónicas para realizar operaciones CRUD utilizando el modelo “Producto”.

```
// Operación para obtener todos los productos
app.get('/productos', async (req, res) => {
  try {
    const productos = await Producto.find();
    res.json(productos);
  } catch (error) {
    console.error('Error al obtener productos:', error);
    res.status(500).send('Error al obtener productos');
  }
});

// Operación para agregar un nuevo producto
app.post('/productos', async (req, res) => {
  try {
    const { nombre, precio, descripcion } = req.body;
    const nuevoProducto = new Producto({ nombre, precio, descripcion });
    await nuevoProducto.save();
    res.status(201).json(nuevoProducto);
  } catch (error) {
    console.error('Error al agregar producto:', error);
    res.status(500).send('Error al agregar producto');
  }
});
```



```
// Operación para actualizar un producto por ID
app.put('/productos/:id', async (req, res) => {
  // ...
});

// Operación para eliminar un producto por ID
app.delete('/productos/:id', async (req, res) => {
  // ...
});
```

- **Configurar el servidor para escuchar en un puerto**

Se inicia el servidor Express y se configura para escuchar en el puerto especificado (en este caso, 3000).

En resumen, este código establece un servidor web con Express, se conecta a una base de datos MongoDB mediante Mongoose, define un esquema y un modelo para productos, y proporciona rutas para realizar operaciones CRUD en la colección de productos.

```
app.listen(PORT, () => {
  console.log(`Servidor Express escuchando en el puerto ${PORT}`);
});
```

Desafío #3

- **Crear un servidor con Express:**
Inicializar un proyecto npm con `npm init -y`.
Instalar Express mediante el comando `npm install express`.
Crear un archivo `server.js` y configurar un servidor básico con Express.
- **Instalar Mongoose:**
Instalar Mongoose mediante el comando `npm install mongoose`.
- **Definir un Schema de Producto:**
Crear un Schema para los productos con los campos nombre, precio, y descripción.
- **Definir un Schema de Usuario (opcional):**
Opcionalmente, crear un Schema para usuarios con los campos nombre, edad, y correo, u otros campos que desees agregar.
- **Configurar CRUD con Mongoose:**
Implementar los cuatro métodos HTTP (GET, POST, PUT, DELETE) para interactuar con la base de datos MongoDB utilizando Mongoose.
Para el caso de la colección de productos, implementar las operaciones CRUD para productos, utilizando los Schemas definidos.
Para el caso de la colección de usuarios (opcional), implementar las operaciones CRUD para usuarios, utilizando el Schema correspondiente.
- **Bonus Extra (COMPLEMENTARIO):**
Testear los endpoints en Postman.
Realizar capturas de pantalla que muestran el correcto funcionamiento de los endpoints y enviarlas como evidencia adicional.

Recuerda que para este desafío necesitarás tener una cuenta en **Mongo Atlas** y haber inicializado el proyecto **npm** con las dependencias de **Express y Mongoose** instaladas. Una vez completado, asegúrate de probar los endpoints en **Postman** para verificar su correcto funcionamiento.



Buenos Aires
aprende
Agencia de Actividades para el Futuro

BA Buenos
Aires
Ciudad