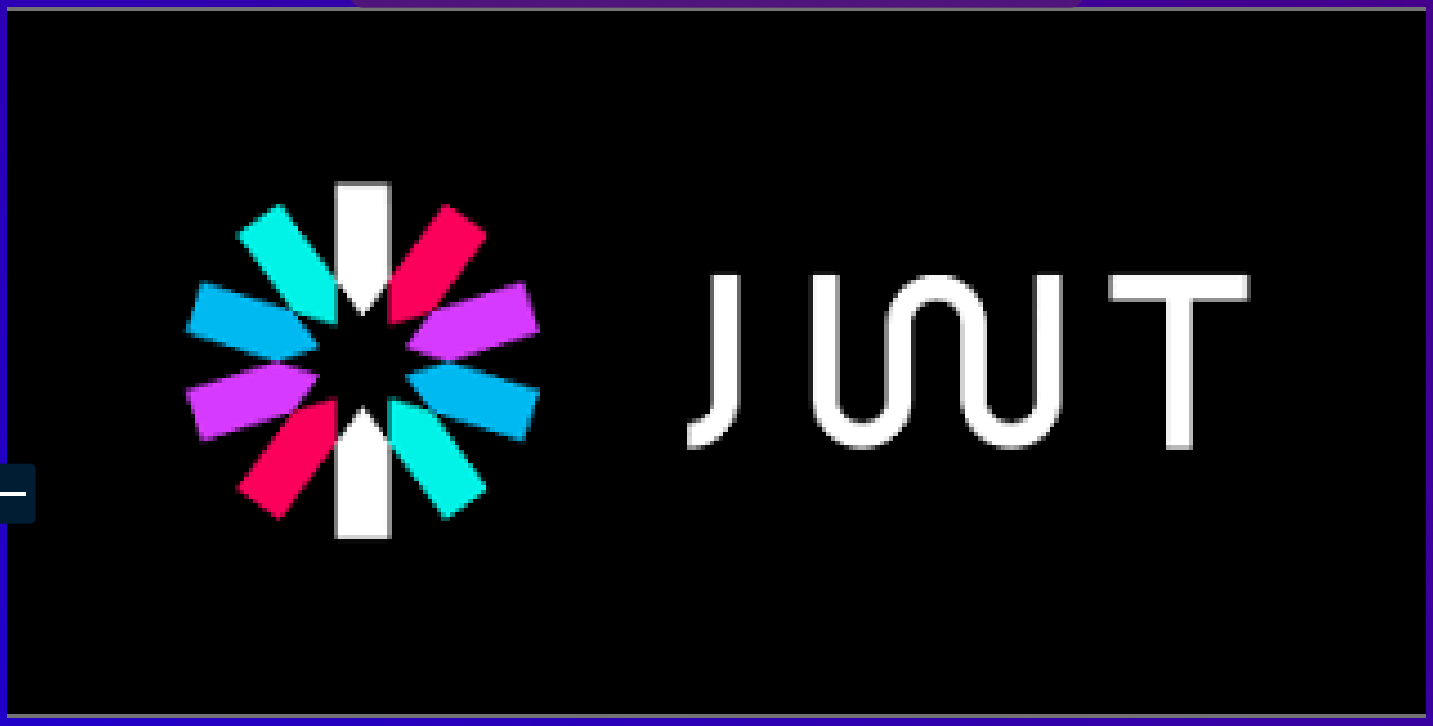
SESIÓN 25

15.04.2024 S25-M2

Json Web Token - Parte 2




```terminal
// terminar, ver sesión 25
```

Copiar código

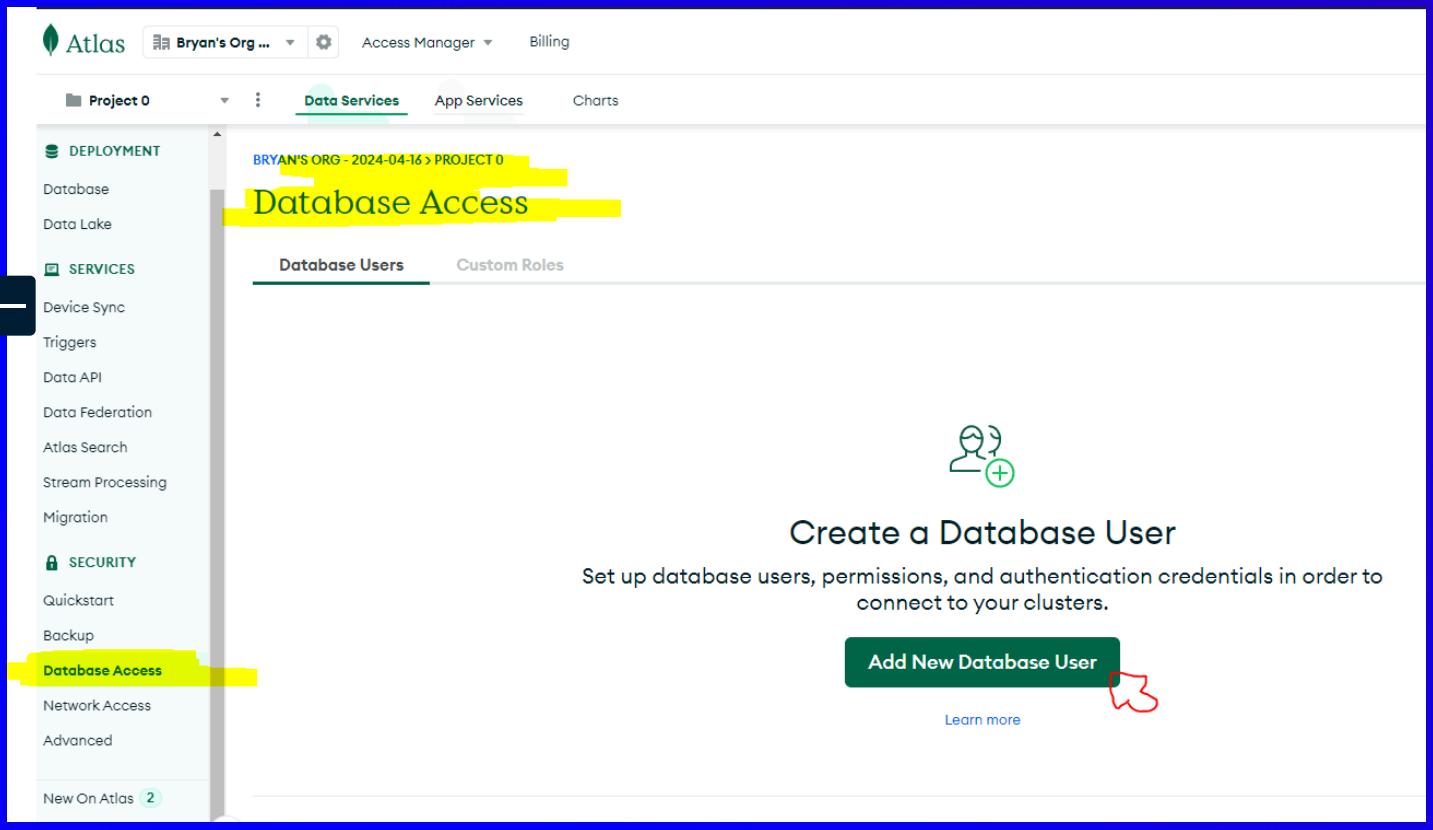
CRUD / Express y MongoDB - Parte 1

MongoDB logueo y configuración con Compass

Para este proyecto se trabajara con una base de datos no relacional, **MongoDb** y para la integración se usará **Express**

Lo primero sera la creación de la cuenta de MongoDb desde su página

Luego de crearse la cuenta, crea tu **usuario** y **contraseña**:



Add New Database User

Create a database user to grant an application or user access to databases and collections in your clusters in this Atlas project. Granular access control can be configured with default privileges or custom roles. You can grant access to an Atlas project or organization using the corresponding Access Manager

Authentication Method

Password	Certificate	AWS IAM (MongoDB 4.4 and up)	PREVIEW Federated Auth (MongoDB 7.0 and up)

MongoDB uses SCRAM as its default authentication method.

Password Authentication

user

••••••

SHOW

Autogenerate Secure Password | Copy

Database User Privileges

Configure role based access control by assigning database user a mix of one built-in role, multiple custom roles, and multiple specific privileges. A user will gain access to all actions within the roles assigned to them, not just the actions those roles share in common. **You must choose at least one role or privilege.** Learn more about roles.

Built-in Role 0 SELECTED

Select one built-in role for this user.

Built-in Role 1 SELECTED

Select one built-in role for this user.

Read and write to any database

Custom Roles

Select your pre-defined custom role(s). Create a custom role in the Custom Roles tab.

Specific Privileges

Select multiple privileges and what database and collection they are associated with. Leaving collection blank will grant this role for all collections in the database.

Restrict Access to Specific Clusters/Federated Database Instances/Stream Processing Instances

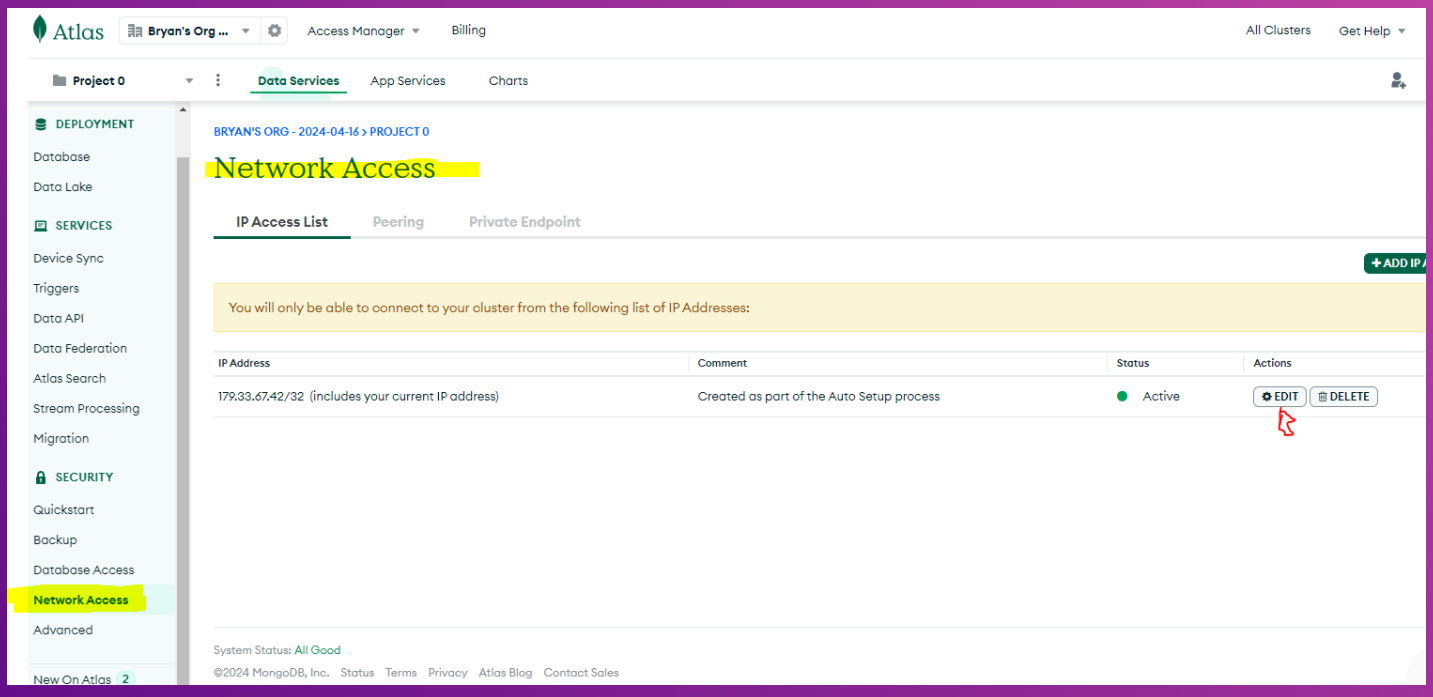
Enable to specify the resources this user can access. By default, all resources in this project are accessible.

Temporary User

This user is temporary and will be deleted after your specified duration of 6 hours, 1 day, or 1 week.

Cancel Add User

Luego de crearse un usuario, verifica que el acceso a la red no tenga restricción alguna, la mejor forma para evitar problemas es la siguiente:



Edit IP Access List Entry

Atlas only allows client connections to a cluster from entries in the project's IP Access List. Each entry should either be a single IP address or a CIDR-notated range of addresses. Learn more.

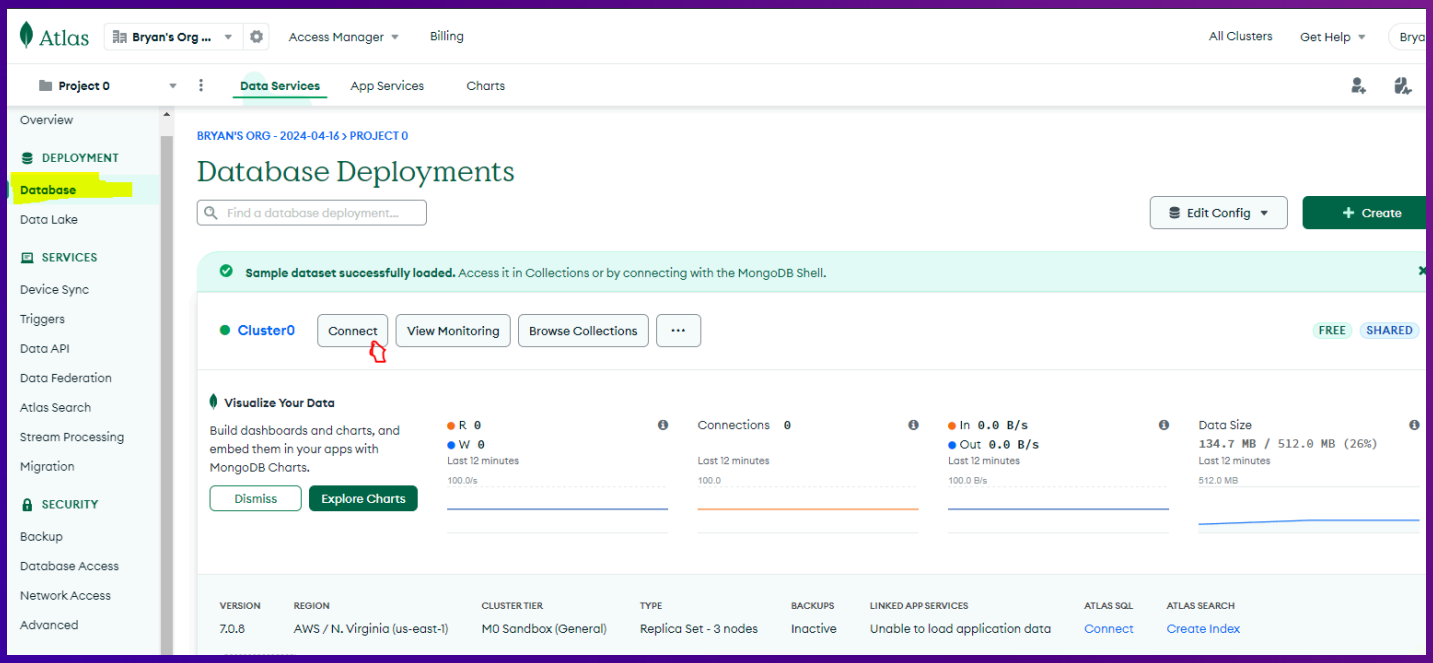
ALLOW ACCESS FROM ANYWHERE

Access List Entry: 0.0.0.0/0

Comment: ok

Cancel | Confirm

Ahora genera una nueva conexión usando **Compass**.



Connect to Cluster0

Set up connection security — 2 Choose a connection method — 3 Connect

Connect to your application

Drivers
Access your Atlas data using MongoDB's native drivers (e.g. Node.js, Go, etc.)

Access your data through tools

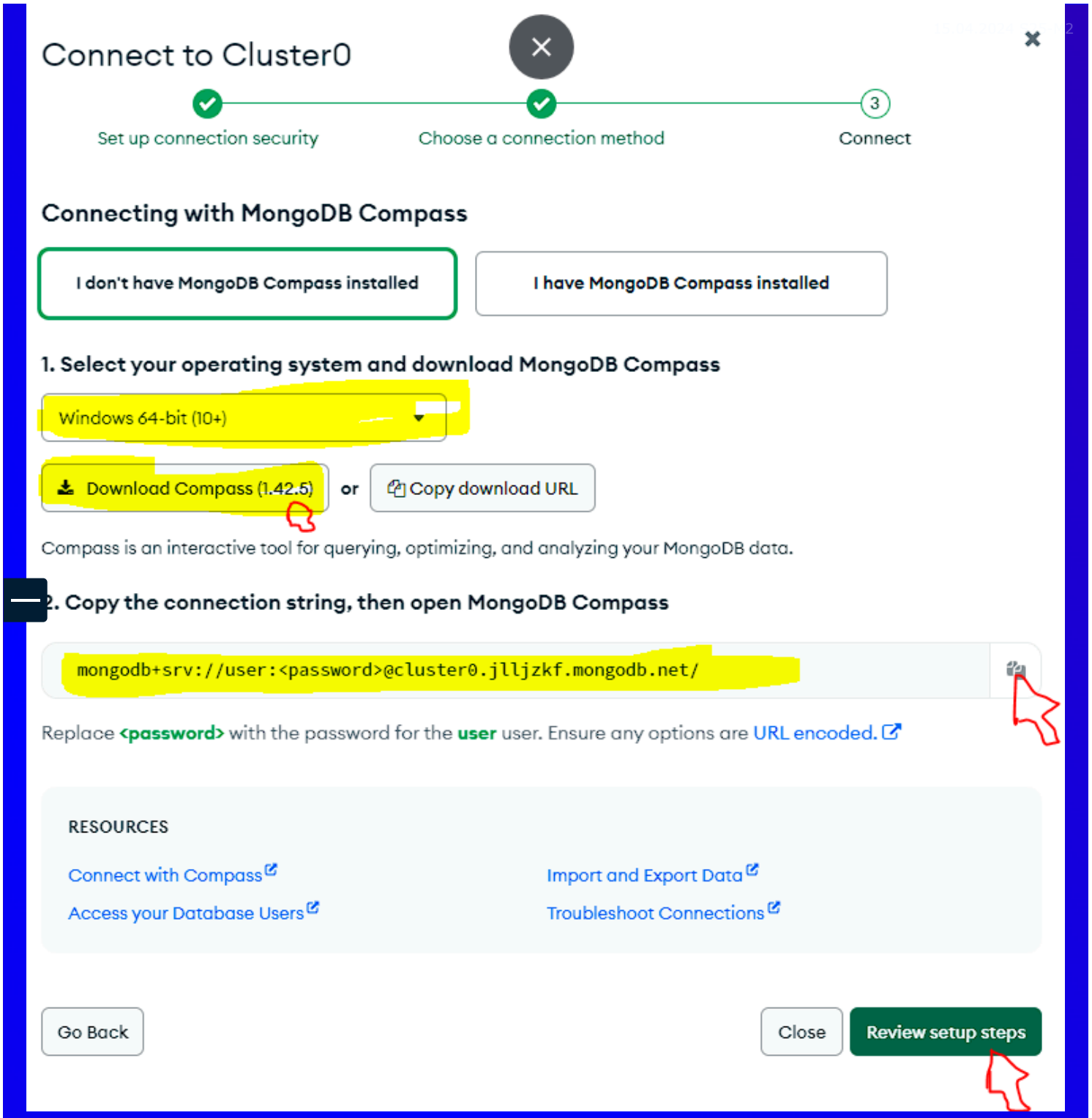
Data Explorer
Browse your Atlas collections without leaving the UI

Compass
Explore, modify, and visualize your data with MongoDB's GUI

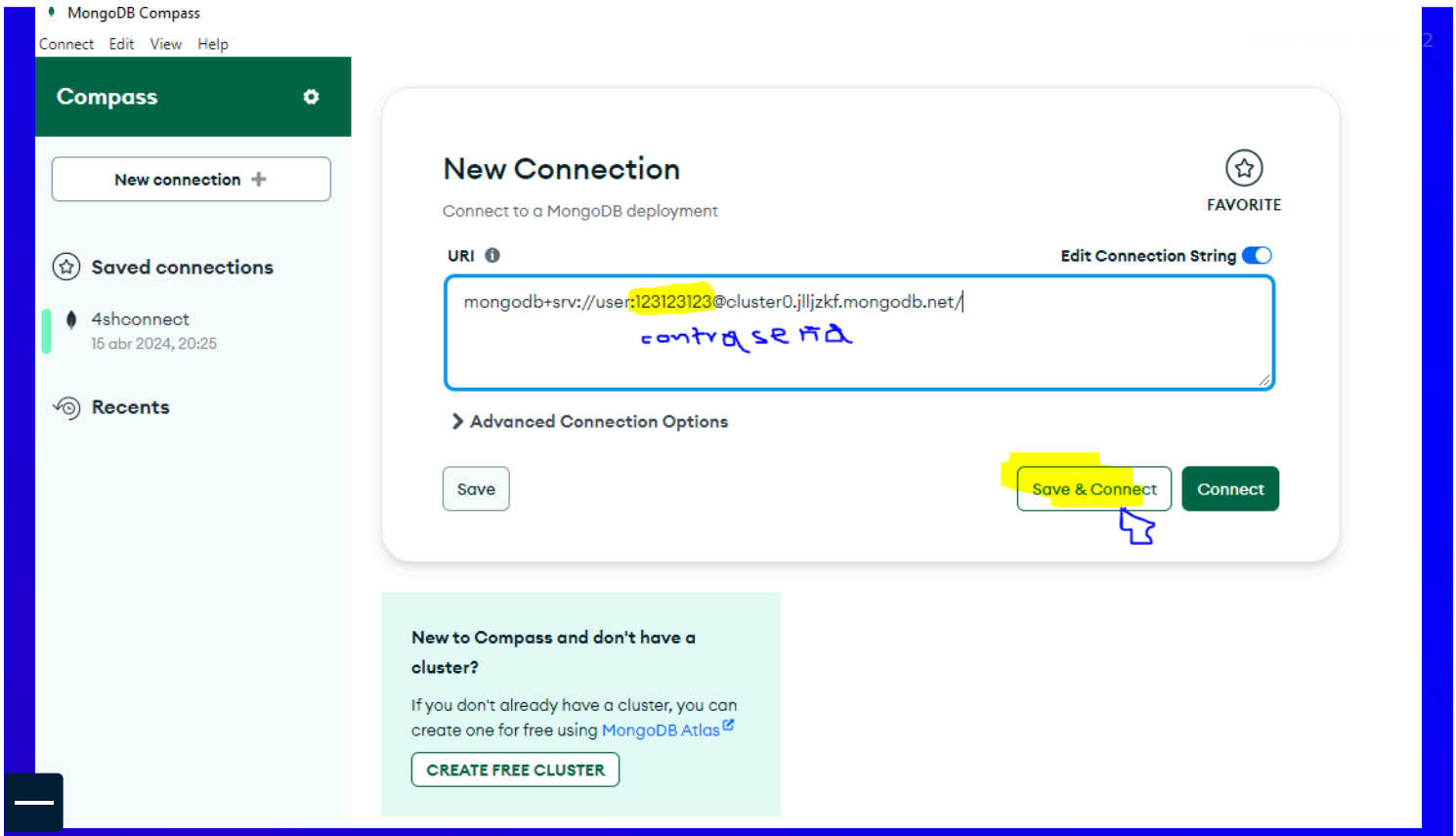
Shell
Quickly add & update data using MongoDB's Javascript command-line interface

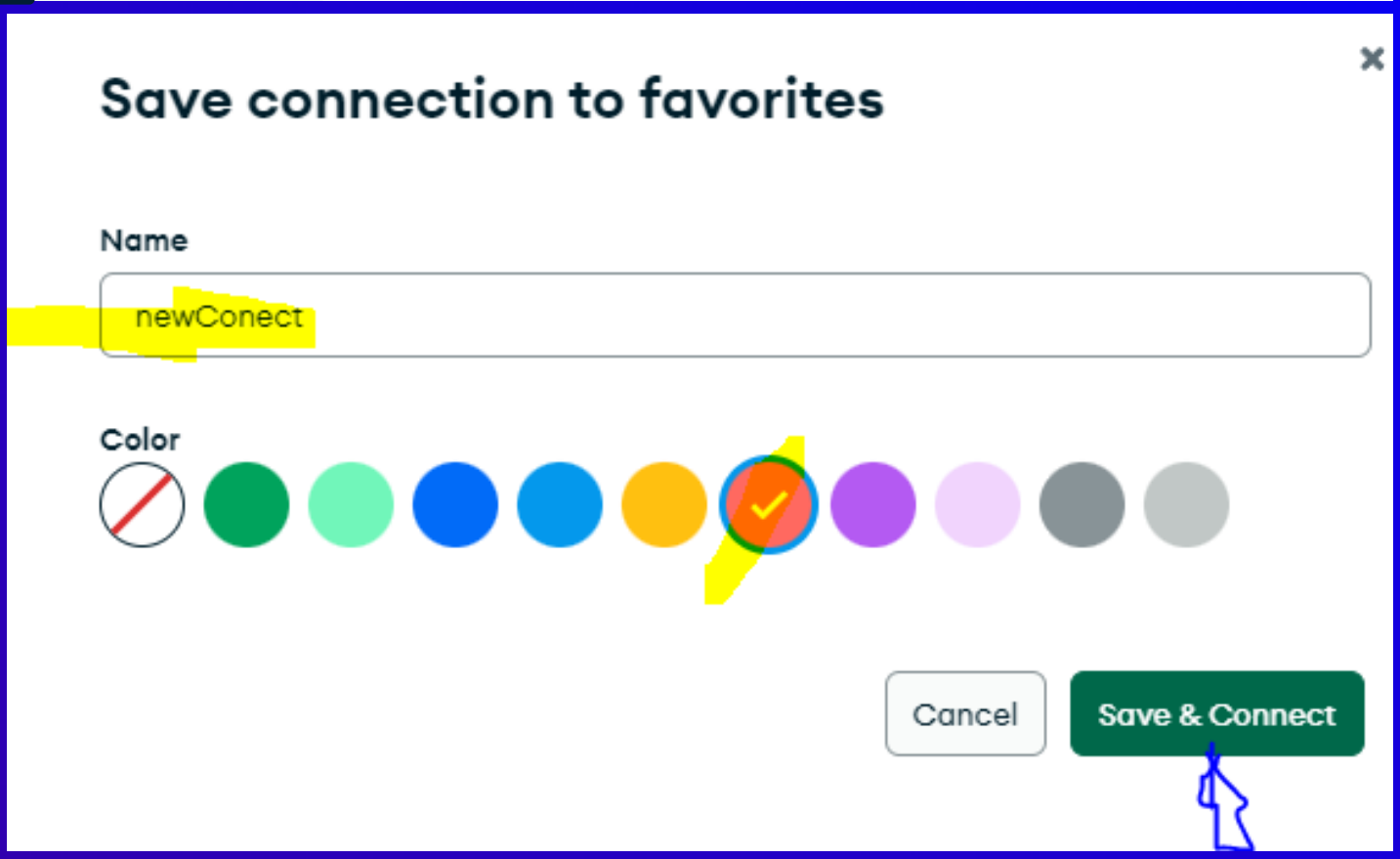
MongoDB for VS Code
Work with your data in MongoDB directly from your VS Code environment

Atlas SQL



Ahora para finalizar **abre Compass** y crea la nueva conexión, pegando el código que acabas de copiar





Para finalizar con esta parte crea una nueva base de datos

Creación de proyecto e integración con Express

Primero vamos a estructurar nuestro proyecto, crea las siguientes carpetas:

estructura de carpetas

Copiar código

```
// Raíz Proyecto
Proyecto
        backend-mongo
```

Ahora inicializamos nuestro proyecto backend node.js con express, y algunas otras dependencias en el directorio **backend-mongo**:

terminal

Copiar código

```
npm i -y // Inicializamos proyecto node.js con parámetros por default
npm i express
npm i mongoose dotenv cors
npm i -D nodemon // Trae los datos del servidor y los actualiza en tiempo real
```

Añadimos al proyecto las siguientes carpetas y archivos:

terminal

Copiar código

```
→ config // New
→ controllers // New
→ models // New
→ node_modules
→ routes // New
↓ src // New
    index.js // New
 .env // New
 package-lock.json
 package.json
```

Por el momento nuestro **index.js** presenta la siguiente configuración:

src/index.js

Copiar código

```
// Aquí iniciaremos express
const express = require('express');

const app = express();
const port = 5000;

app.listen(port, () => console.log('Nuestro servidor se encuentra conectado 😎 http://localhost:

app.get('/', (req, res) =>{
  res.send('Bienvenido, nuestro servidor esta configurado');
});
```

A nuestro **package.json** se lo ha modificado para que inicie nuestra app desde *src/index.js*

package.json

Copiar código

```
{
  "name": "backend-mongo",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "start": "nodemon src/index.js" // Modificado
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "cors": "^2.8.5",
    "dotenv": "^16.4.5",
    "express": "^4.19.2",
    "mongoose": "^8.3.1"
  },
  "devDependencies": {
    "nodemon": "^3.1.0"
  }
}
```

Por ahora en nuestras variables de entorno tenemos la uri de conexión con MongoDB

.env

Copiar código

```
MONGO_URL = mongodb+srv://4USER:4MIn4C_.ON@cluster0.4bxgl6w.mongodb.net/apiclientes
```

NOTA

'apiclientes' es el nombre de la base de datos a la cual nos vamos a conectar, la cual creaste por Compass

Finalmente para terminar con esta parte y ejecutar la app usa el siguiente código:

Ejecutar app

Copiar código

```
npm run start
```