



TECNOLOGICO DE ESTUDIOS SUPERIORES DE ECATEPEC

INGENIERIA EN SISTEMAS COMPUTACIONALES

PROFA.: GRISelda CORTES BARRERA

**MATERIA: BASE DE DATOS PARA DISPOSITIVOS
MOVILES**

SANCHEZ PABLO CAROLINA

Documento de datos personal (Excel)

GRUPO:5801

FECHA: 24/01/2022

DOCUMENTO DE DATOS PERSONAL (EXCEL)

REQUERIMIENTOS Y NECESIDADES

Crear componente

- Cargar el archivo y que se empiece a leer el archivo
- Ya sea que mande a la base o que muestre los datos en pantalla

PROBLEMÁTICA

Crear un componente para el proyecto de la materia base de datos para dispositivos móviles, en el componente a realizar se tendrá que seleccionar un archivo en formato Excel el cual tenga algunos datos específicos del empleado, una vez seleccionado el archivo se podrán ver los datos en pantalla.

OBJETIVOS

- Generales
 - Creación de componente
- Específicos
 - Creación de un componente en él se podrá visualizar la información de un archivo en formato Excel.
 - El archivo tendrá la información de los empleados de la empresa (cliente).
 - Identificar los datos o campos para la construcción de la base de datos en excel

ANALISIS

Check-in en empresas

Para implementar la información para el check-in debemos de tener en cuenta los tipos de datos que vamos a necesitar.

Tipos de datos internos

- Estructurados

Los datos estructurados se clasifican con mayor frecuencia como datos internos cuantitativos, y es el tipo de datos con el que la mayoría de nosotros estamos acostumbrados a trabajar. Piensa en datos que encajen perfectamente en campos y columnas fijos en bases de datos relacionales y hojas de cálculo.

Los ejemplos de datos estructurados incluyen nombres, fechas, direcciones, números de tarjetas de crédito, información bursátil, geolocalización, etc.

Los datos estructurados están muy organizados y se comprenden fácilmente mediante el lenguaje de máquina. Quienes trabajan con bases de datos relacionales pueden ingresar, buscar y manipular datos estructurados con relativa rapidez. Esta es la característica más atractiva de los datos estructurados.

- Semi-estructurados

Los datos semiestructurados son los datos que no se ajustan a un modelo de datos, pero tienen alguna estructura. Carece de un esquema fijo o rígido. Son los datos que no residen en una base de datos racional pero que tienen algunas propiedades organizativas que facilitan su análisis. Con algún proceso, podemos almacenarlos en la base de datos relacional.

- No estructurados

Los datos no estructurados se clasifican con mayor frecuencia como datos internos cualitativos y no pueden procesarse y analizarse utilizando herramientas y métodos convencionales.

Los ejemplos de datos no estructurados incluyen texto, video, audio, actividad móvil, actividad en redes sociales, imágenes satelitales, imágenes de vigilancia; la lista sigue y sigue.

Los datos no estructurados son difíciles de deconstruir porque no tienen un modelo predefinido, lo que significa que no se pueden organizar en bases de datos relacionales. En cambio, las bases de datos no relacionales o NoSQL son las más adecuadas para administrar datos no estructurados.

Bases de datos

Se trata de un conjunto organizado de archivos que pueden usarse entre sí, teniendo en cuenta dos categorías de datos: internos y externos. Los datos internos se refieren a la información que la empresa obtiene de forma regular de sus actividades. Los datos externos son la información obtenida de fuentes ajenas a la empresa.

DISEÑO

Propuesta de diseño para el módulo en el que se podrá ver la información del archivo Excel que se usará.

1. En la siguiente imagen se muestra un ejemplo de cómo será la primera ventana al dar clic para visualizar el archivo Excel, en esta ventana saldrá

una advertencia en la cual dará a conocer la información que tendrá que tener el archivo que se cargará.

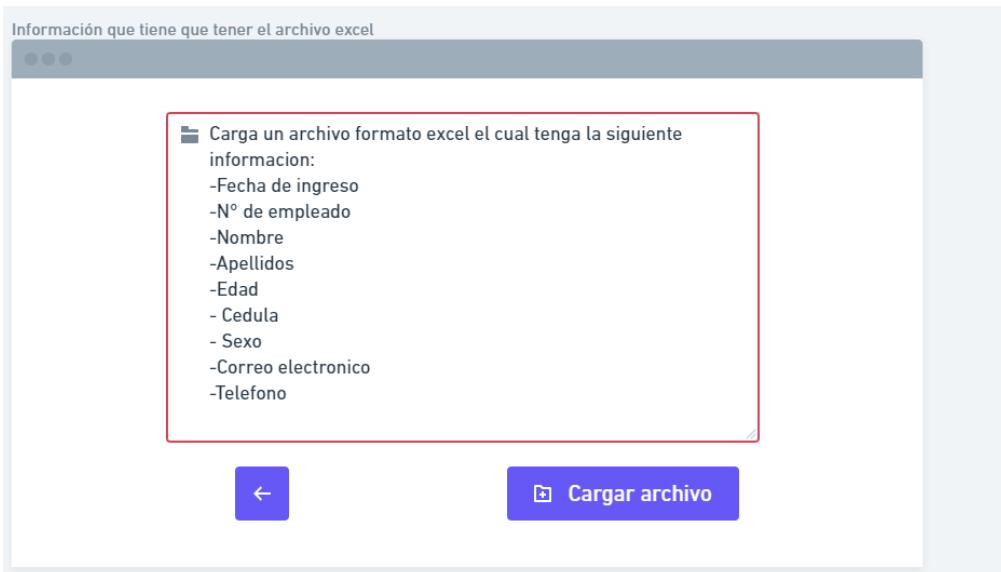


Ilustración 1 Advertencia con especificaciones de archivo

2. En la siguiente imagen se muestra el diseño que arrogara al dar clic en el botón cargar archivo de la ilustración 1. En esta ventana se tendrá un cuadro de texto en el cual mostrara el nombre del archivo que se ha seleccionado, después del está cuadro de texto tendremos n botón el cual nos permitirá examinar nuestros documentos y así seleccionar el archivo a visualizar.

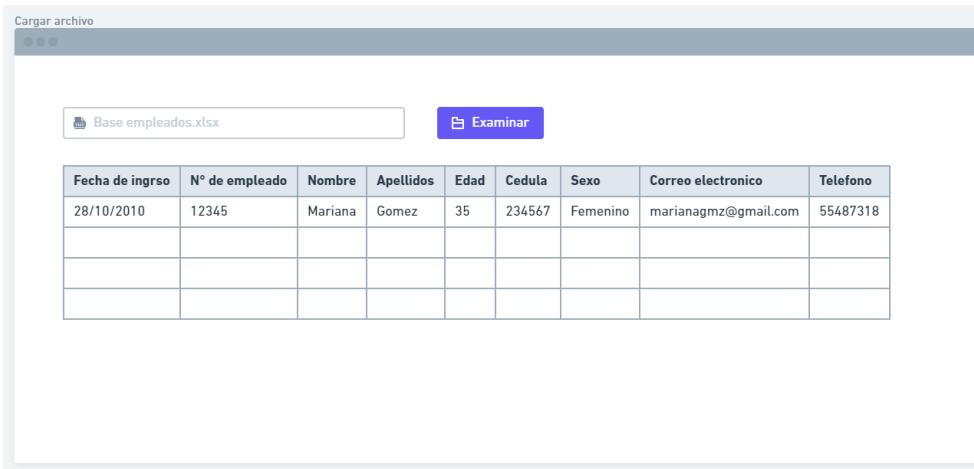


Ilustración 2 Cargar archivos

MANUAL TECNICO

HERRAMIENTAS DE TRABAJO

-ANGULAR CLI:

Angular Cli es la herramienta con la que vamos a poder generar aplicaciones donde nos permite crear, depurar y publicar. Además, Angular Cli es muy potente, versátil y sencillo de dominar los aspectos básicos.

-BOOTSTRAP:

Esta es una herramienta que proporciona interactividad en la página, por lo que ofrece una serie de componentes que facilitan la comunicación con el usuario, como menús de navegación, controles de página, barras de progreso y más.

-WEBSTORM

WebStorm es un IDE para JavaScript que facilita el desarrollo en dicho lenguaje. Para cada objeto muestra sus posibles métodos y propiedades (objetos predefinidos del lenguaje). También tiene un editor HTML que muestra el CSS asociado a cada etiqueta de la página que estamos editando.

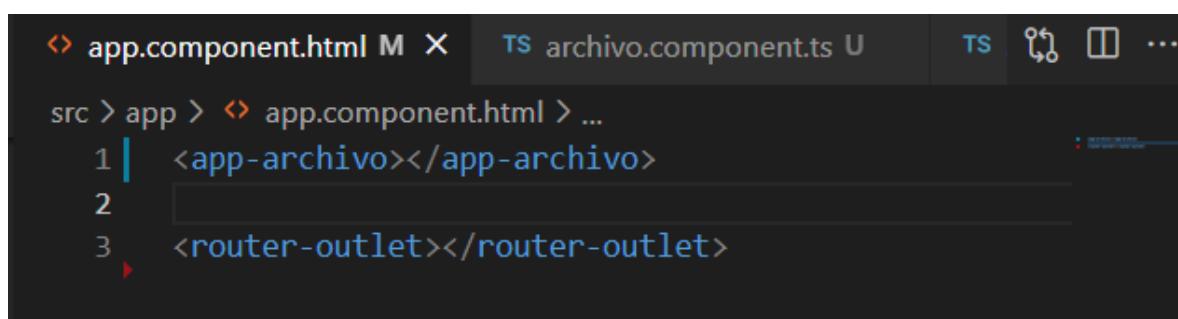
-VISUAL STUDIO CODE

editor de código fuente que permite trabajar con diversos lenguajes de programación, admite gestionar tus propios atajos de teclado y refactorizar el código. Es gratuito, de código abierto y nos proporciona una utilidad para descargar y gestionar extensiones con las que podemos personalizar y potenciar esta herramienta.

CODIGO DEL PROYECTO

El proyecto consta de un modulo llamado abrir en el cual se encuentran dos componentes el primero de ellos es el componente archivo y el segundo es seleccionar:

app.component.html



```
src > app > app.component.html > ...
1 | <app-archivo></app-archivo>
2 |
3 | <router-outlet></router-outlet>
```

archivo.component.ts

```
↳ app.component.html M    TS archivo.component.ts U X    TS app.component.ts M    ↳ archivo.component.html U
src > app > archivo > TS archivo.component.ts > ...
1 import { Component, OnInit } from '@angular/core';
2 import * as XLSX from 'xlsx';
3
4 type AOA= any [][] ;
5
6 @Component({
7   selector: 'app-archivo',
8   templateUrl: './archivo.component.html',
9   styleUrls: ['./archivo.component.css']
10 })
11 export class ArchivoComponent implements OnInit {
12
13   data: AOA = [
14     [1, 2],
15     [3, 4],
16   ];
17   wopts: XLSX.WritingOptions = { bookType: 'xlsx', type: 'array' };
18   fileName: string = 'SheetJS.xlsx';
19
20   constructor() { }
21
22   ngOnInit(): void {
23   }
24
25   onFileChange(evt:any){
26
27     //coneccion al lector de archivos
28
29     const target: DataTransfer= <DataTransfer>(evt.target);
30
31     if(target.files.length !== 1) throw new Error('No se pueden usar multiples archivos');
32     const reader: FileReader = new FileReader();
33
34     reader.onload=(e:any)=>{
35       //lee el libro de trabajo
36       const bstr:string=e.target.result;
37       const wb : XLSX.WorkBook=XLSX.read(bstr,{type:'binary'});
38
39       //toma la primera hoja
40       const wsname:string=wb.SheetNames[0];
41       const ws:XLSX.WorkSheet=wb.Sheets[wsname];
42       console.log(ws);
43       //guarda los datos
44       this.data=<AOA>(XLSX.utils.sheet_to_json(ws,{header:1}));
45       console.log(this.data);
46     };
47     reader.readAsBinaryString(target.files[0]);
48   }
}
```

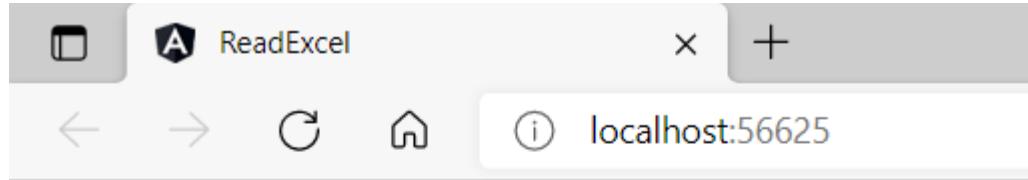
archivo.component.html

```
src > app > archivo > archivo.component.html > ...
1   <p>SELECCION DE ARCHIVO EN FORMATO EXCEL</p>
2   <input type = "file" (change)="onFileChange($event)" multiple="false"/>
3   <table class="sjs-table">
4       <tbody>
5           <tr *ngFor="let row of data">
6               <td *ngFor="let val of row">
7                   {{val}}
8               </td>
9           </tr>
10      </tbody>
11  </table>
12
```

MANUAL DE USUARIO

El presente manual le ayudara al usuario a manejar de una manera eficiente al sistema en el cual se podrá car un archivo Excel, para que de esta forma el usuario no tenga ningún problema para cargar su archivo.

1. en nuestra primera venta podremos observa un boten con el cual podremos seleccionar nuestro archivo en xlsx



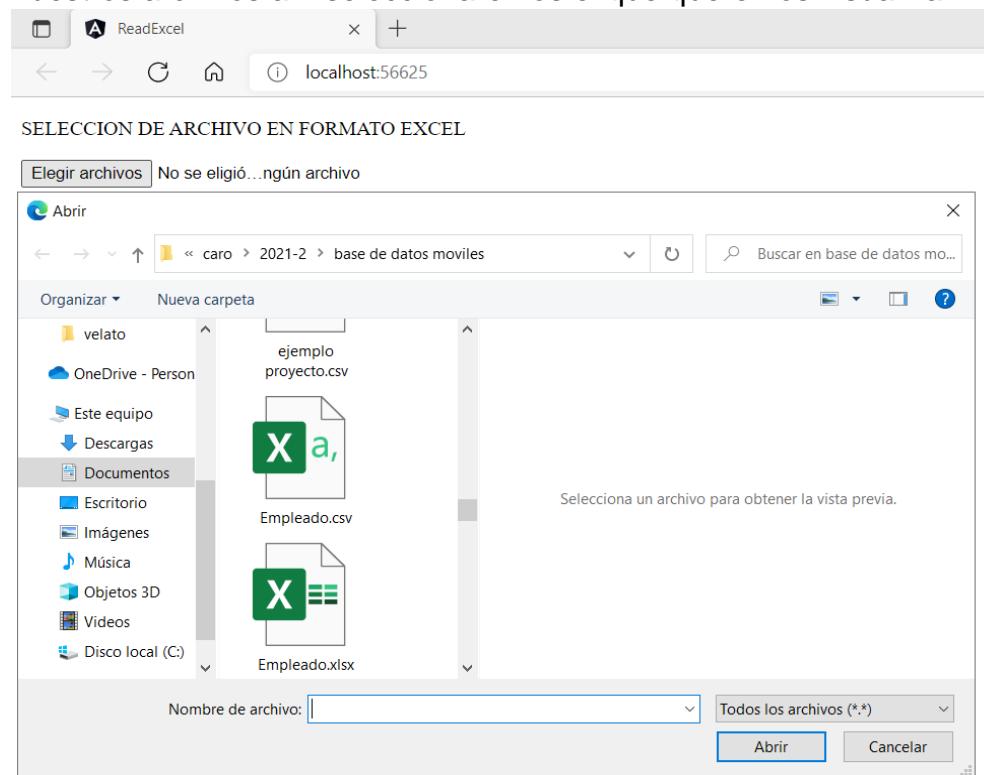
SELECCION DE ARCHIVO EN FORMATO EXCEL

Elegir archivos No se eligió...ngún archivo

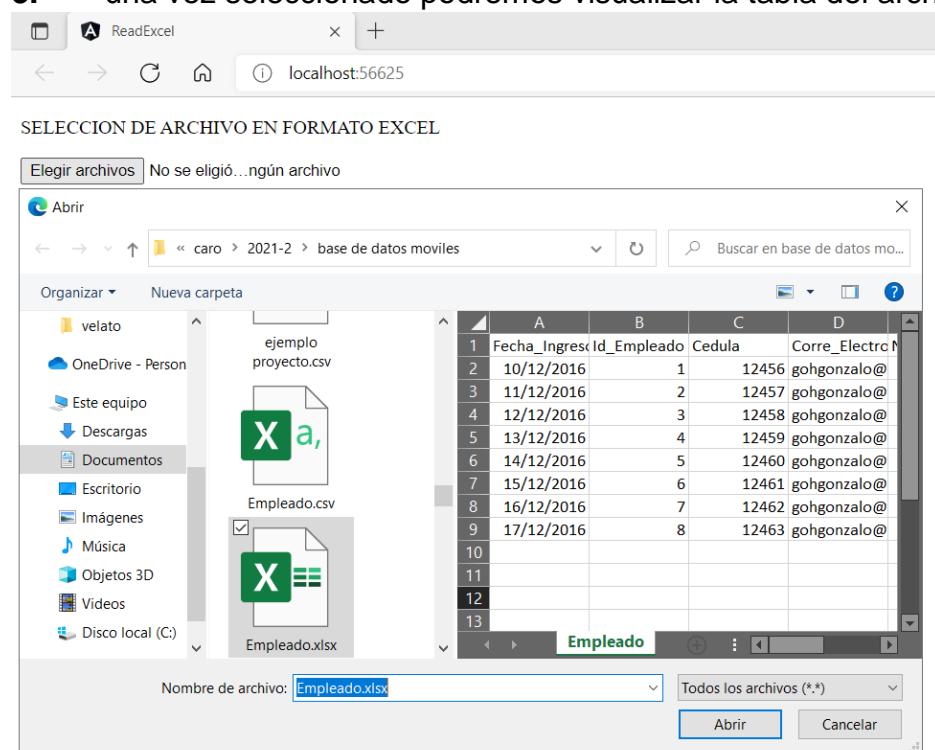
1 2

3 4

2. Al dar clic al botón se abrirá una ventana la cual nos dará acceso a nuestros archivos ahí seleccionaremos el que queremos visualizar.



3. Una vez seleccionado podremos visualizar la tabla del archivo.



Fecha	Ingreso	Id_Empleado	Cedula	Corre_Electronico	N_SeguroSocial	Empresa	Puesto	Id_Datos_Generales
42654.99958333333	1	12456	gohgonzalo@gmail.com	123456		corpotacio sa cv tester	2	
42685.99958333333	2	12457	gohgonzalo@gmail.com	123460		corpotacio sa cv tester	4	
42715.99958333333	3	12458	gohgonzalo@gmail.com	123464		corpotacio sa cv tester	6	
13/12/2016	4	12459	gohgonzalo@gmail.com	123468		corpotacio sa cv tester	8	
14/12/2016	5	12460	gohgonzalo@gmail.com	123472		corpotacio sa cv tester	10	
15/12/2016	6	12461	gohgonzalo@gmail.com	123476		corpotacio sa cv tester	12	
16/12/2016	7	12462	gohgonzalo@gmail.com	123480		corpotacio sa cv tester	14	
17/12/2016	8	12463	gohgonzalo@gmail.com	123484		corpotacio sa cv tester	16	

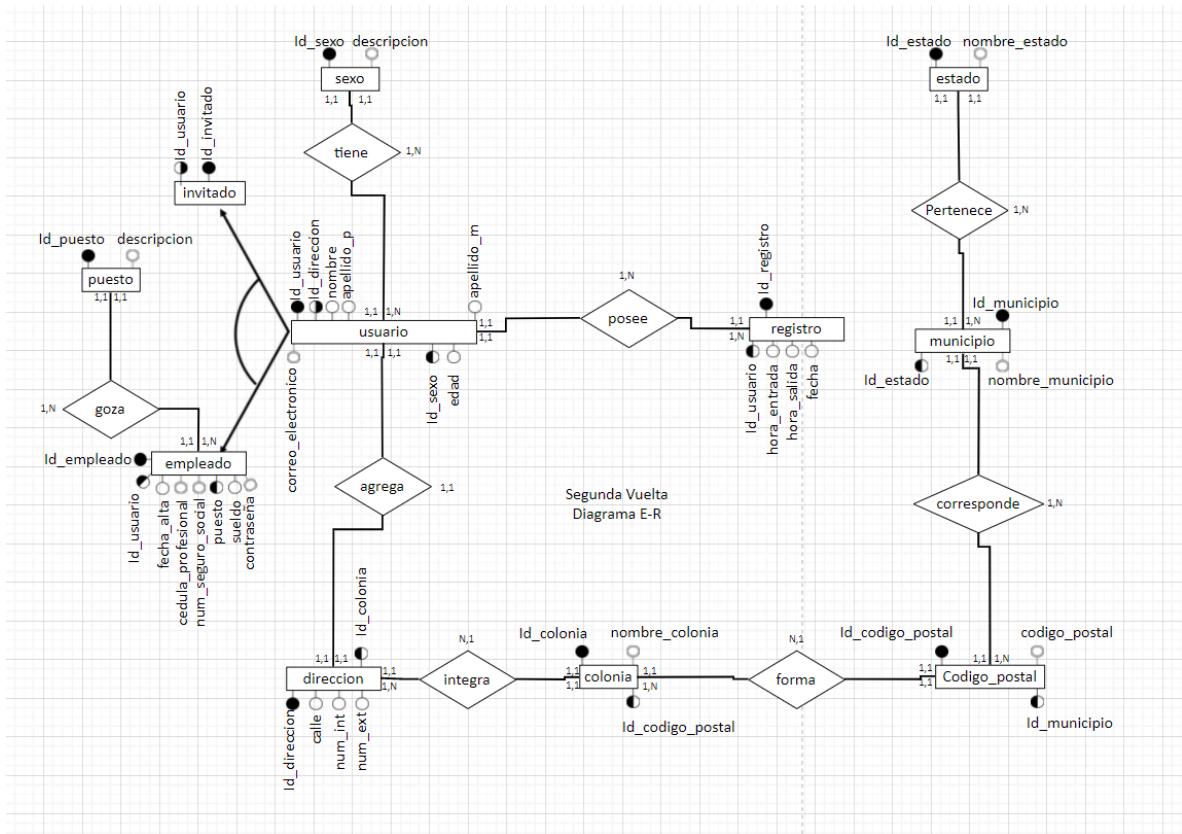
MONGODB

- MongoDB almacena datos en documentos flexibles similares a JSON, por lo que los campos pueden variar entre documentos y la estructura de datos puede cambiarse con el tiempo
- El modelo de documento se asigna a los objetos en el código de su aplicación para facilitar el trabajo con los datos
- Las consultas ad hoc, la indexación y la agregación en tiempo real ofrecen maneras potentes de acceder a los datos y analizarlos
- MongoDB es una base de datos distribuida en su núcleo, por lo que la alta disponibilidad, la escalabilidad horizontal y la distribución geográfica están integradas y son fáciles de usar
- MongoDB es de uso gratuito

MONGO DB (LOCAL)

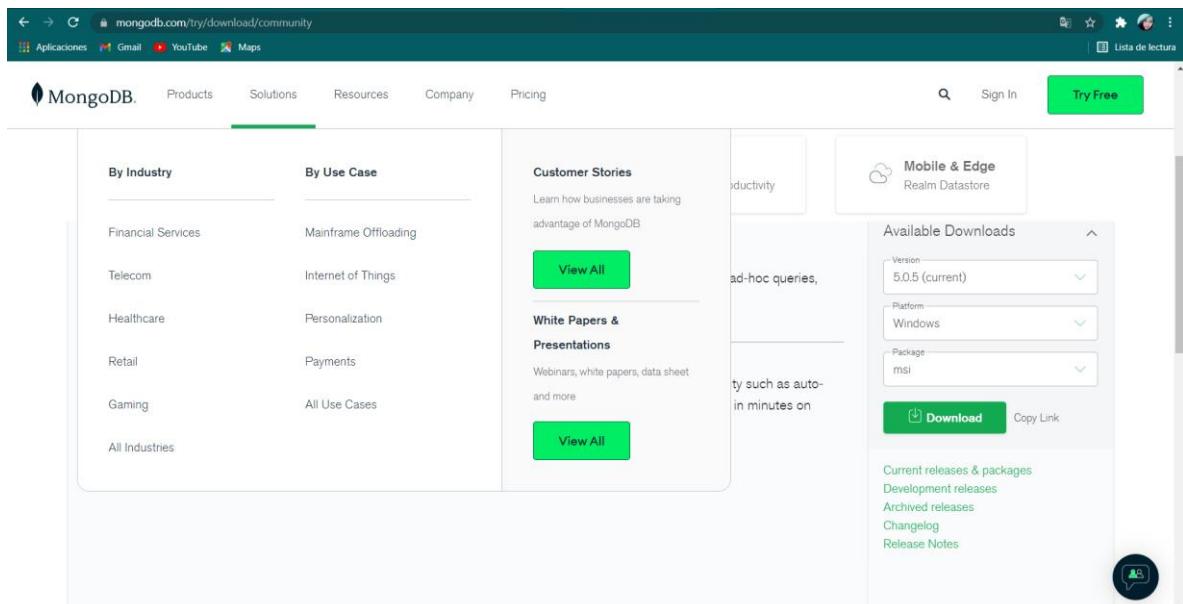
DIAGRAMA DE ENTIDAD-RELACION

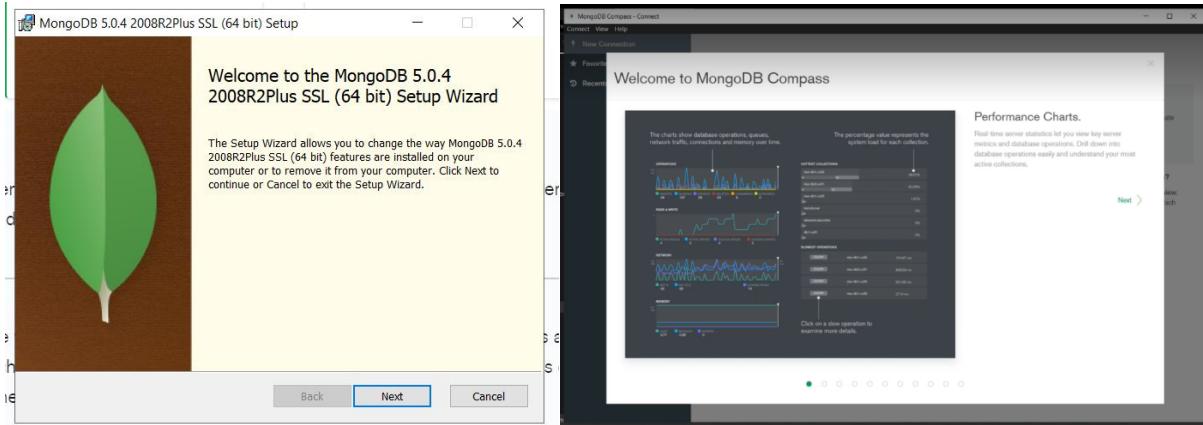
El modelo fue hecho en base a todos los demás módulos, así crean un solo modelo para el uso de todos los equipos.



CREACION DE BASE DE DATOS CON MONGODB COMPASS (LOCAL)

1. Descarga e instalación de aplicación de Mongo DB





2. Creación de la base de datos en MongoDB que se llama Direcciones con la colección Ciudad

Database Name	Storage Size	Collections	Indexes
Direcciones	86.0KB	4	4
admin	20.5KB	0	1
config	12.3KB	0	2
local	20.5KB	1	1

3. Creación de las colecciones de Municipios, Estados y CP y colonias

The image shows three separate 'Create Collection' dialog boxes, each with a different collection name and slightly different content based on the specific collection type.

Municipios Dialog:

- Collection Name: Municipios
- Capped Collection

Fixed-size collections that support high-throughput operations that insert and retrieve documents based on insertion order. ⓘ
- Use Custom Collation

Collation allows users to specify language-specific rules for string comparison, such as rules for lowercase and accent marks. ⓘ
- Time-Series

Time-series collections efficiently store sequences of measurements over a period of time.

Estados Dialog:

- Collection Name: Estados
- Capped Collection

Fixed-size collections that support high-throughput operations that insert and retrieve documents based on insertion order. ⓘ
- Use Custom Collation

Collation allows users to specify language-specific rules for string comparison, such as rules for lowercase and accent marks. ⓘ
- Time-Series

Time-series collections efficiently store sequences of measurements over a period of time.

CP-Col Dialog:

- Collection Name: CP-Col
- Capped Collection

Fixed-size collections that support high-throughput operations that insert and retrieve documents based on insertion order. ⓘ
- Use Custom Collation

Collation allows users to specify language-specific rules for string comparison, such as rules for lowercase and accent marks. ⓘ
- Time-Series

Time-series collections efficiently store sequences of measurements over a period of time.

4. Subir archivos de Excel con extencion .csv a cada colección

MongoDB Compass - localhost:27017/Direcciones.Ciudad

Connect View Collection Help

Local

HOST localhost:27017

CLUSTER Standalone

EDITION MongoDB 5.0.4 Community

Filter your data

- Direcciones
 - CP-Col
 - Ciudad
 - Estados
 - Municipios
- admin
- config
- local

_MONGOSH

Direcciones.Ciudad

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

ADD DATA

VIEW

OPTIONS FIND RESET

DOCUMENTS 2 TOTAL SIZE 110B AVG. SIZE 55B INDEXES 1 TOTAL SIZE 36.0KB AVG. SIZE 36.9KB

Import To Collection Direcciones.Ciudad

Select File

Select a file...

CSV

CANCEL IMPORT

MongoDB Compass - localhost:27017/Direcciones.Estados

Connect View Collection Help

Local

HOST localhost:27017

CLUSTER Standalone

EDITION MongoDB 5.0.4 Community

Filter your data

- Direcciones
 - CP-Col
 - Ciudad
 - Estados
 - Municipios
- admin
- config
- local

Direcciones.Estados

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

ADD DATA

VIEW

OPTIONS FIND RESET

DOCUMENTS 33 TOTAL SIZE 2.3KB AVG. SIZE 69B INDEXES 1 TOTAL SIZE 20.0KB AVG. SIZE 20.5KB

Import To Collection Direcciones.Estados

Select File

Select a file...

CSV

CANCEL IMPORT

The screenshot shows the MongoDB Compass interface connected to the 'Local' database at 'localhost:27017'. The 'Direcciones' collection is selected, displaying 162 documents. A modal window titled 'Import To Collection Direcciones.Municipios' is open, prompting for a file to import. The file 'Catalogo_mun.csv' is selected in the 'Select File' dialog. The modal includes 'CANCEL' and 'IMPORT' buttons.

5. Creación de Vistas

MongoDB Compass - localhost:27017/Direcciones.CP-Col

Local

4 DBS 5 COLLECTIONS

HOST localhost:27017

CLUSTER Standalone

EDITION MongoDB 5.0.4 Community

Filter your data

▼ Direcciones

- CP-Col
- Ciudad
- Estados
- Municipios

admin

config

local

▼ Direcciones.CP-Col

Aggregations

Schema Explain Plan Indexes Validation

COLLATION Estados-Col_Cp

SAVE

DOCUMENTS 775 TOTAL SIZE 75.4KB AVG. SIZE 97B INDEXES 1 TOTAL SIZE 4.1KB AVG. SIZE 4.1KB

SAMPLE MODE AUTO PREVIEW

Preview of Documents in the Collection

Select an operator to construct expressions used in the aggregation pipeline stages. [Learn more](#)

Output after \$lookup stage (Sample of 20 documents)

```

1 /**
2  * from: The target collection.
3  * localField: The local join field.
4  * foreignField: The target join field.
5  * as: The name for the results.
6  * pipeline: The pipeline to run on the joined collection.
7  * let: optional variables to use in the pipeline field
8 */
9 +
10 from "Estados"
11 localField: "idEsto",
12 foreignField: "IdEsto",
13 as: "Estados-Col_Cp"
14 }
    
```

ADD STAGE

MongoDB Compass - localhost:27017/Direcciones.CP-Col

Local

4 DBS 5 COLLECTIONS

HOST localhost:27017

CLUSTER Standalone

EDITION MongoDB 5.0.4 Community

Filter your data

▼ Direcciones

- CP-Col
- Ciudad
- Estados
- Municipios

admin

config

local

▼ Direcciones.CP-Col

Aggregations

Schema Explain Plan Indexes Validation

COLLATION CP_Col-Municipio

SAVE

DOCUMENTS 775 TOTAL SIZE 75.4KB AVG. SIZE 97B INDEXES 1 TOTAL SIZE 4.1KB AVG. SIZE 4.1KB

SAMPLE MODE AUTO PREVIEW

Preview of Documents in the Collection

Select an operator to construct expressions used in the aggregation pipeline stages. [Learn more](#)

Output after \$lookup stage (Sample of 20 documents)

```

1 /**
2  * from: The target collection.
3  * localField: The local join field.
4  * foreignField: The target join field.
5  * as: The name for the results.
6  * pipeline: The pipeline to run on the joined collection.
7  * let: optional variables to use in the pipeline field
8 */
9 +
10 from "Municipios"
11 localField: "idMun",
12 foreignField: "idMun",
13 as: "CP-Col"
14 }
    
```

ADD STAGE

The screenshot shows the MongoDB Compass interface with the 'Aggregations' tab selected for the 'Direcciones.Estados' collection. The pipeline stage '\$lookup' is highlighted. The code editor displays the following aggregation pipeline:

```

1 /**
2  * from: The target collection.
3  * localField: The local join field.
4  * foreignField: The target join field.
5  * as: The resulting field name.
6  * pipeline: The pipeline to run on the joined collection.
7  * let: Optional variables to use in the pipeline field.
8 */
9 *
10 from: "Municipios"
11 localField: "Iddo",
12 foreignField: "IdEstado",
13 as: "Municipios"
14 }

```

The output stage shows a sample of 20 documents, each containing fields like '_id', 'Iddo', 'Edo', 'idCd', and 'Municipios'. One document is expanded to show its 'Municipios' array.

MONGODB ATLAS

MongoDB Atlas es un servicio global de base de datos de documentos en la nube para las aplicaciones modernas. Implementar una MongoDB completamente administrada garantiza la disponibilidad, la escalabilidad y la conformidad con las normas de seguridad gracias a la automatización inteligente para mantener el rendimiento a escala a medida que las aplicaciones evolucionan. Amplíe sus datos para gestionar cualquier carga de trabajo que use la plataforma de datos de aplicaciones de MongoDB, como la búsqueda de texto completo y el análisis en tiempo real.

CREACION DE SERVIDOR MONGODB ATLAS

1. Buscar en nuestro navegador de internet MongoDB Atlas, dar clic en el enlace

The screenshot shows a Google search results page for the query "mongodb atlas". The top result is the official MongoDB Atlas website, titled "MongoDB Atlas Database | Multi-Cloud Database Service". Below it, there are several other links related to MongoDB and its services. A sidebar on the right lists "Preguntas relacionadas" (Related questions) with expandable dropdowns for various MongoDB topics.

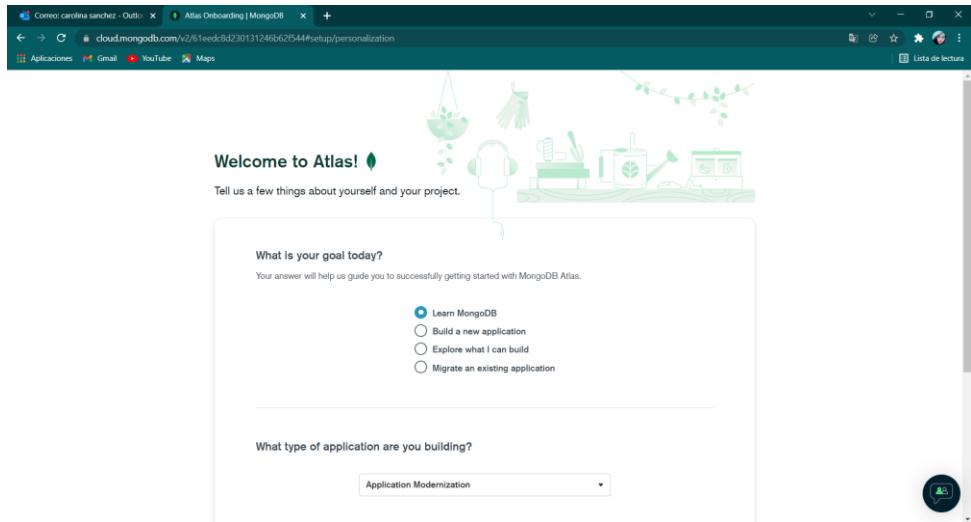
2. Llena el formulario para registrarse y acepta los términos y condiciones

Get started free
No credit card required

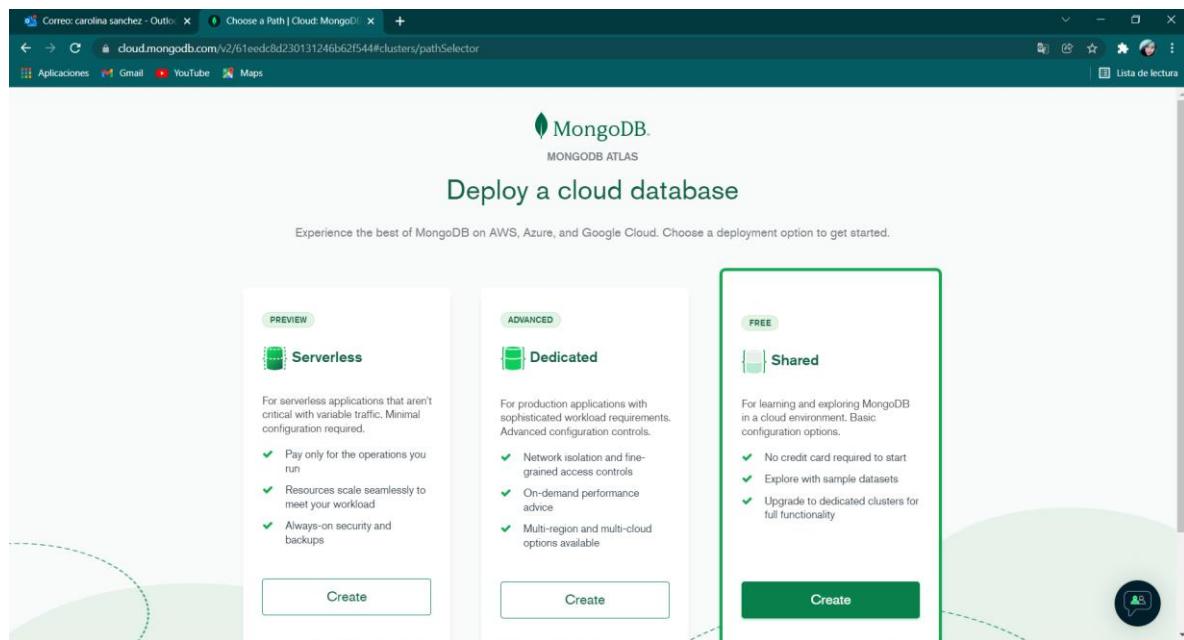
 Sign up with Google
or
[Text input field] Your Company (optional)
[Text input field] How are you using MongoDB?
[Text input field] Your Work Email
[Text input field] First Name
[Text input field] Last Name
[Text input field] Password
8 characters minimum
 I agree to the terms of service and privacy policy.
Get started free

Already have an account? [Sign in](#).

3. Contesta la en cuesta que te sale una vez que hayas verificado tu cuenta



4. Una vez finaliza selecciona la tercera opción y da clic en el botón créate, y vuelve a dar clic en créate clouster.



Correos: carolina.sanchez - Outlook | Create Deployment | Cloud: MongoDB

cloud.mongodb.com/v2/61eedc8d230131246b62f544#clusters/edit?filter=starter

Aplicaciones Gmail YouTube Maps

Carolina

CLUSTERS > CREATE A SHARED CLUSTER

Create a Shared Cluster

Welcome to MongoDB Atlas! We've recommended some of our most popular options, but feel free to customize your cluster to your needs. For more information, check our [documentation](#).

PREVIEW Serverless Dedicated **FREE** Shared

For learning and exploring MongoDB in a sandbox environment. Basic configuration controls. No credit card required to start. Upgrade to dedicated clusters for full functionality. Explore with sample datasets. Limit of one free cluster per project.

Cloud Provider & Region AWS, N. Virginia (us-east-1)

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

Back Create Cluster

Correos: carolina.sanchez - Outlook | Create Deployment | Cloud: MongoDB

https://cloud.mongodb.com/v2/61def8db403f16e51e54468#clusters/edit?filter=starter

Sao Paulo (sa-east-1) MIDDLE EAST Mumbai (ap-south-1)
Bahrain (me-south-1) ★ Osaka (ap-northeast-3) ★
Cape Town (af-south-1) ★

M0 Sandbox (Shared RAM, 512 MB Storage) Encrypted

MongoDB 4.4, No Backup

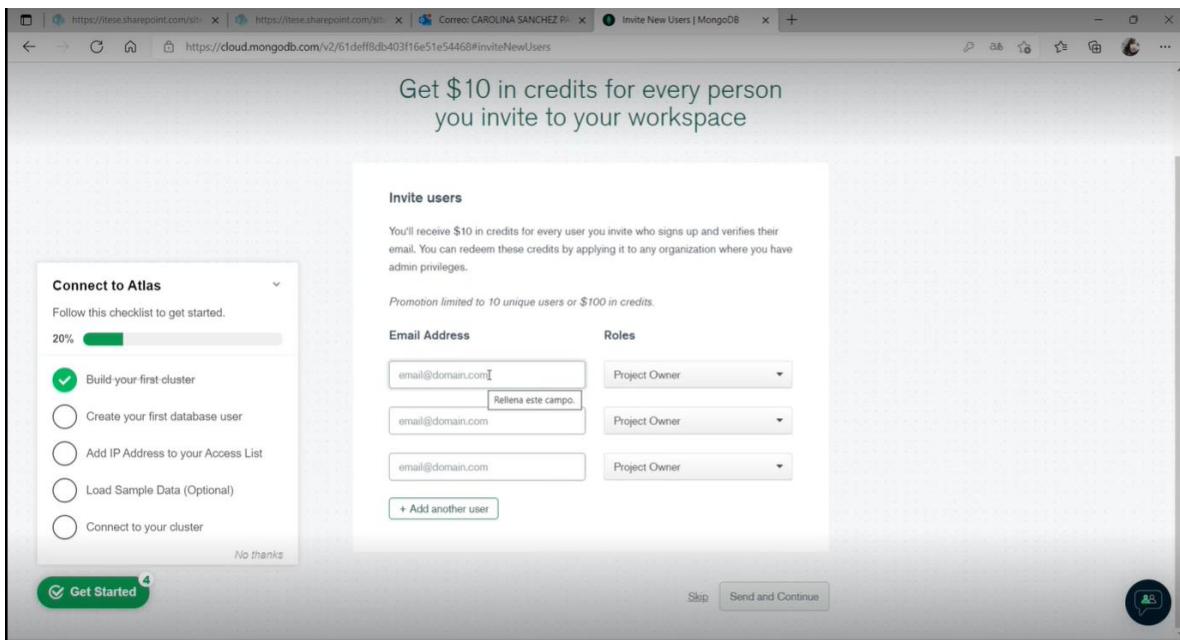
Cluster Name Clustermongo

One time only: once your cluster is created, you won't be able to change its name.

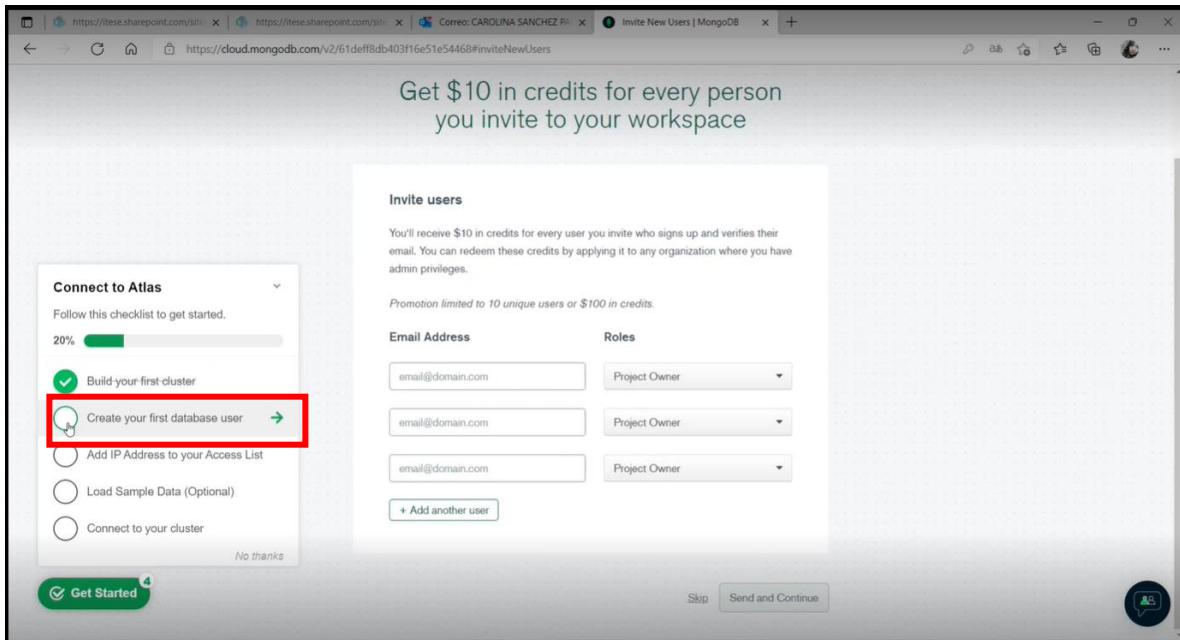
FREE Free forever! Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

Back Create Cluster

5. completar los pasos que se muestran del lado izquierdo de la pantalla.



6. create your first database user: da clic en el siguiente paso, te aparecerá una nota en la cual te dirá que des clic una vez hecho eso presiona la opción Add new database user, asigna un nombre y una contraseña para este usuario y da clic en agregar usuario.



The screenshot shows the MongoDB Cloud interface for a project named "Carolina's Org - 202...". The main heading is "Database Deployments" with a sub-section "Cluster mongo". A red box highlights the "Database Access" section under the "SECURITY" tab, which contains links for "Database Access" and "Network Access". Below this, a message says "Click here to manage your project's database users". To the right, a progress bar indicates "Your cluster is being created." with a note: "New clusters take between 1-3 minutes to provision." At the bottom, there is a table with cluster details: VERSION 4.4.11, REGION AWS / N. Virginia (us-east-1), CLUSTER TIER M0 Sandbox (General), TYPE Replica Set - 3 nodes, BACKUPS Inactive, LINKED REALM APP None Linked, and ATLAS SEARCH Create Index.

The screenshot shows the MongoDB Cloud interface for a project named "Carolina's Org - 202...". The main heading is "Database Access" with a sub-section "Database Users". A red box highlights the "Add New Database User" button. The page includes a "Create a Database User" section with the instruction: "Set up database users, permissions, and authentication credentials in order to connect to your clusters." Below this is a "Learn more" link.

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)

MongoDB uses [SCRAM](#) as its default authentication method.

Password Authentication

SHOW

[Autogenerate Secure Password](#) [Copy](#)

Database User Privileges

For advanced role based access control, you can apply your own custom roles or add specific privileges to this user. You can combine these role types with a built-in role. 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.](#)

Database User Privileges

For advanced role based access control, you can apply your own custom roles or add specific privileges to this user. You can combine these role types with a built-in role. 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
Select one built-in role for this user.

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.

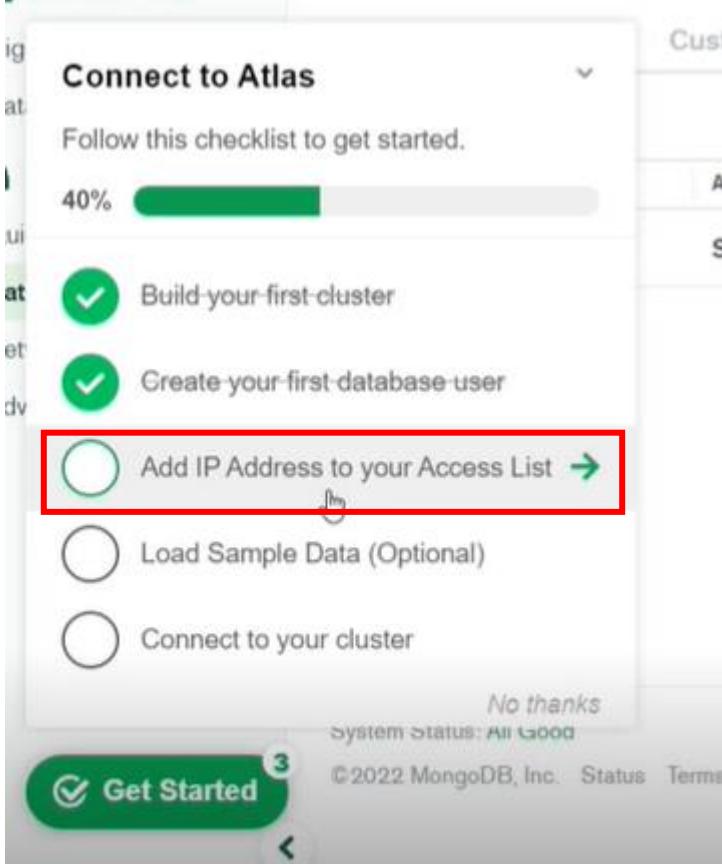
Restrict Access to Specific Clusters/Data Lakes
Enable to specify the resources this user can access. By default, all resources in this project are accessible.

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

The screenshot shows the MongoDB Atlas Database Access interface. On the left sidebar, under 'DATA SERVICES', 'Database Access' is selected. The main area displays a table for 'Database Users'. One row is visible for 'Carolina', which has the 'readWriteAnyDatabase@admin' role assigned. A blue banner at the top states: 'We are deploying your changes (current action: creating a plan)'. A tooltip on the right says: 'Guarda tus contraseñas de forma más segura en Microsoft Edge'. At the bottom, there's a 'Get Started' button.

7. Una vez terminado el paso anterior te volverá a aparecer el cuadro de los pasos a seguir da clic en Add IP Address, a continuación en Network, add IP, y por ultimo llena las casillas con la información y da clic en confirmación.

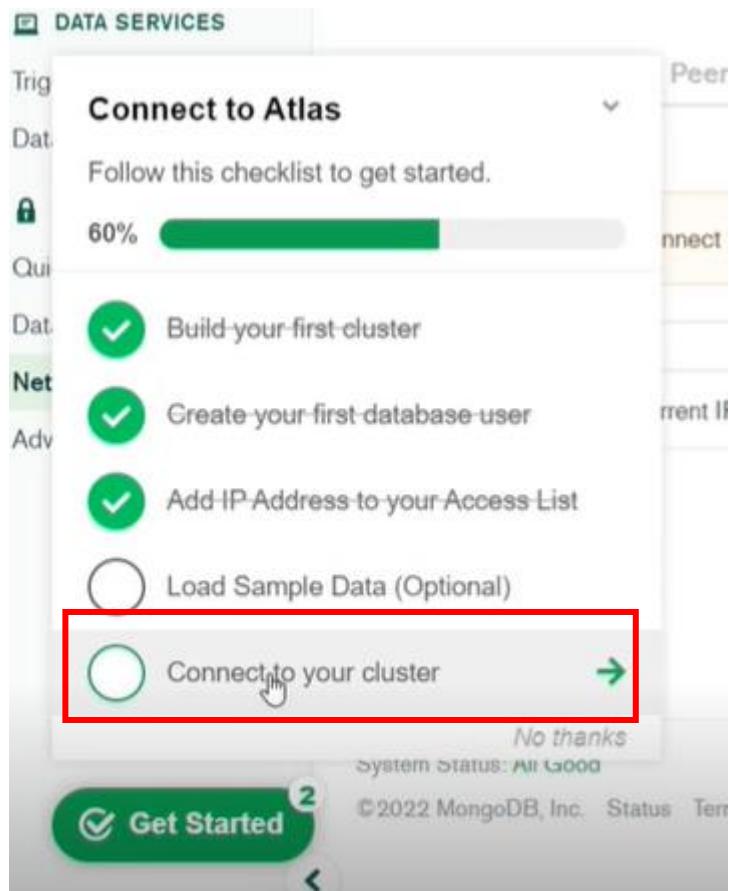


The screenshot shows the MongoDB Atlas Database Access page. On the left sidebar, under the SECURITY section, 'Database Access' is selected. In the main content area, there is a table for 'Database Users'. A red box highlights the 'Network Access' link in the table header. Below the table, a callout box says 'Click here to manage IP address configuration and VPC peering'. At the bottom of the page, there is a 'Get Started' button.

The screenshot shows the MongoDB Atlas Network Access page. On the left sidebar, 'Network Access' is selected. In the main content area, there is a section titled 'Add an IP address' with the sub-instruction 'Configure which IP addresses can access your cluster.' A red box highlights the 'Add IP Address' button. At the bottom of the page, there is a 'Get Started' button.

This is a detailed view of the 'Add IP Access List Entry' dialog box. It contains fields for 'Access List Entry' (set to 0.0.0.0) and 'Comment' (set to desarrollo). There is a toggle switch for 'This entry is temporary and will be deleted in' with a dropdown set to '6 hours'. At the bottom right, there are 'Cancel' and 'Confirm' buttons, with 'Confirm' highlighted by a red box.

8. Da clic en el último paso Connect to your cluster, espera que se termine de crear nuestro clouster, da clic en el botón de connect esto arrojara una ventana la cual daremos clic en la última opción en esta parte tendremos que copiar el URL que nos aparece.



The screenshot shows the MongoDB Atlas Database Deployments page with the following details:

- Deployment status: We are deploying your changes: 3 of 3 servers complete (current action: configuring MongoDB)
- Project: CAROLINA'S O&O - 2022-01-12 > PROJECT 0
- Database: Clustermongo
- Deployment Type: Connect
- Message: Your cluster is being created..
- Table headers: VERSION, REGION, CLUSTER TIER, TYPE, BACKUPS, LINKED REALM APP, ATLAS SEARCH
- Table data:

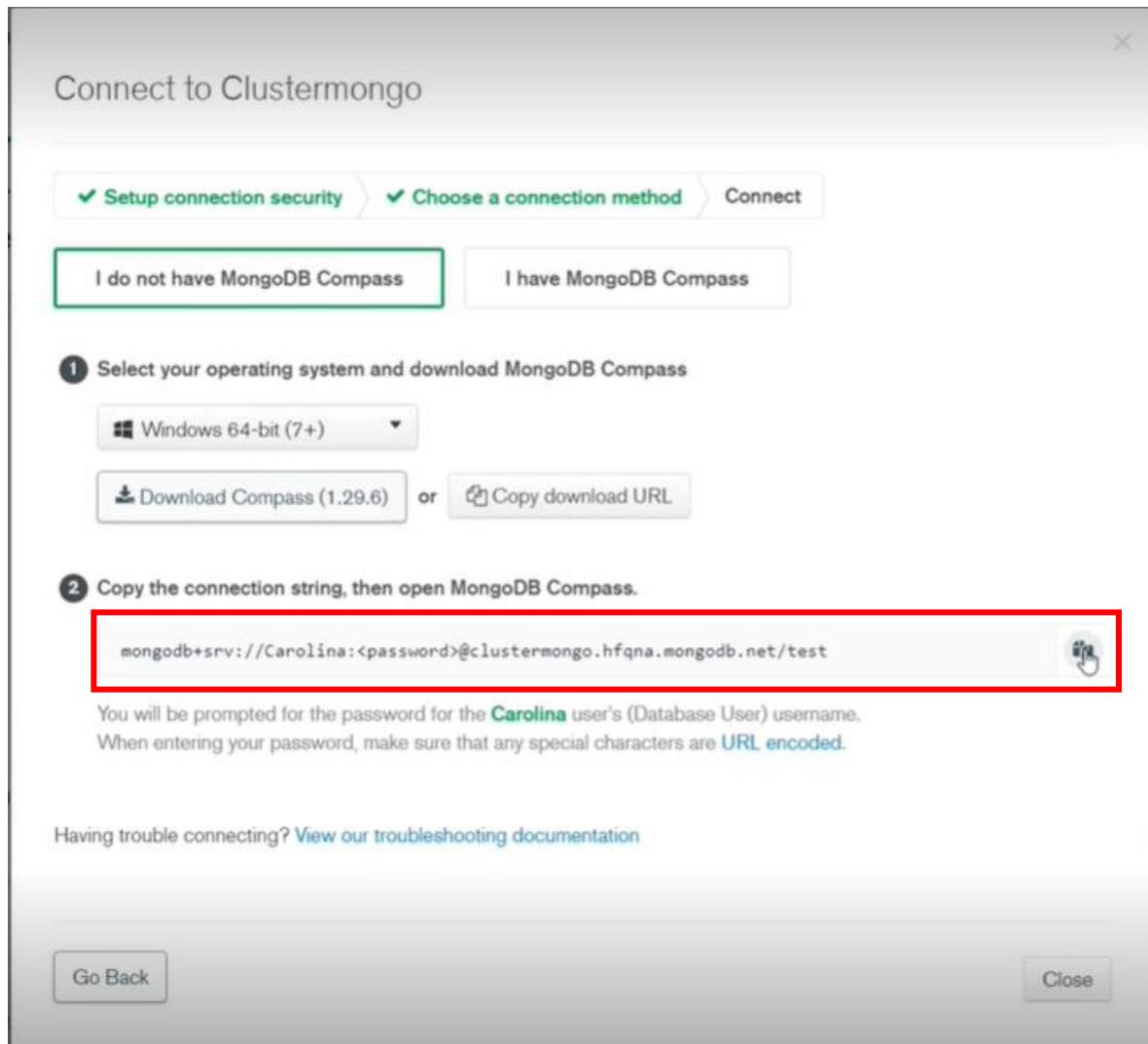
VERSION	REGION	CLUSTER TIER	TYPE	BACKUPS	LINKED REALM APP	ATLAS SEARCH
4.4.11	AWS / N. Virginia (us-east-1)	M0 Sandbox (General)	Replica Set - 3 nodes	Inactive	None Linked	Create Index

The screenshot shows the MongoDB Atlas interface. On the left, there's a sidebar with 'Project 0' selected. Under 'DEPLOYMENT', 'Databases' is highlighted. Under 'DATA SERVICES', 'Triggers' and 'Data API' are listed, with 'PREVIEW' next to Data API. Under 'SECURITY', 'Quickstart', 'Database Access', 'Network Access', and 'Advanced' are listed. In the center, under 'Atlas', it says 'CARMELA'S ORO - 2022-01-12 > PROJECT 0'. Below that is a section titled 'Database Deployments' with a search bar. A cluster named 'Clustermongo' is listed, with its status as 'FREE'. A red box highlights the 'Connect' button. Below the cluster details, there's a table with columns: VERSION, REGION, CLUSTER TIER, TYPE, BACKUPS, LINKED REALM APP, and ATLAS SEARCH. At the bottom, there's a 'Get Started' button and some footer text.

The screenshot shows a modal window titled 'Connect to Clustermongo'. It has a navigation bar at the top with steps: 'Setup connection security' (with a checkmark), 'Choose a connection method' (highlighted in blue), and 'Connect'. Below this, it says 'Choose a connection method' and provides a link to 'View documentation'. It then lists three options:

- Connect with the MongoDB Shell**: Interact with your cluster using MongoDB's interactive Javascript interface.
- Connect your application**: Connect your application to your cluster using MongoDB's native drivers.
- Connect using MongoDB Compass**: Explore, modify, and visualize your data with MongoDB's GUI.

The 'Connect using MongoDB Compass' option is highlighted with a large red box. At the bottom of the modal are 'Go Back' and 'Close' buttons.



9. Pegar el URL en nuestro proyecto de webstorm, cambiando la parte de password por nuestra contraseña, agrega el codigo

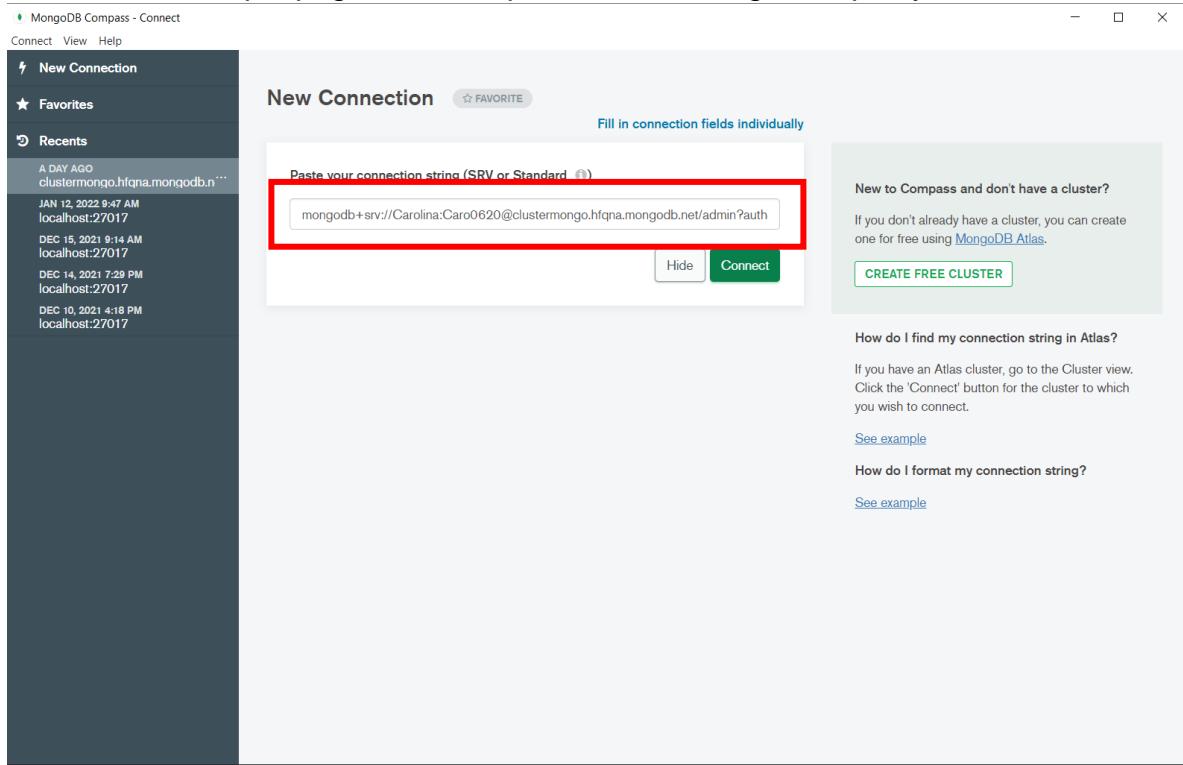
```
const mongoose = require('mongoose');
const cadena = 'mongodb+srv://Carolina:<password>@clustermongo.hfqna.mongodb.net/test';
```

```
package.json basededatos.js src\index.js routes\index.js datos.json
1 const mongoose = require('mongoose');
2 const cadena = 'mongodb+srv://Carolina:Caro0620@clustermongo.hfqna.mongodb.net/usuario';
3 mongoose.connect(cadena, options: {
4   useNewUrlParser: true,
5   useUnifiedTopology: true
6 }).then(db=>console.log('base de datos conectada'))
7 .catch(e=>console.log(e));
```

10. Agrega la siguiente línea a tu código

```
package.json basededatos.js src\index.js routes\index.js datos.json
1 const express = require('express');
2 const app = express();
3 const morgan = require('morgan');
//configuración
5 app.set('port',process.env.PORT || 4000);
6 app.set('json spaces', 2)
7 require('../configuraciones/basededatos');
//middleware mediador entre frontend y backend
9 app.use(morgan('dev'));
10 app.use(express.urlencoded({ extended:false }));
11 app.use(express.json());
12 //llamada de ruta archivo
13 app.use('/api/datos',require('./routes/index'));
14 app.use('/api/guardar',require('./routes/index'));
15 //inicio de servidor
16 app.listen(app.get('port'), hostname ()=>{
17   //obtención de backtick alt+96
18   console.log(`el servidor esta en el puerto ${4000}`)
19});
```

11. El URL tendrás que pegarlo en la aplicación de mongo compas y conéctalo



CODIGO MONGO DB ATLAS

usuariocontrolador.js

```
package.json × basedatos.js × usuariocontrolador.js × routes\usuarios.js × models\usuarios.js × index.js × datos.json ×
1 const user = require('../models/usuarios');
2 exports.insertarusuario = async (req,res)=> {
3     console.log('inserta usuario',req.body);
4     const {Fecha_Ingreso,Id_Emppleado,Cedula,Correo_Electronico,N_SeguroSocial,Empresa,Puesto,Id_Datos_Generales} = req.body;
5     const newuser = new user( doc:{Fecha_Ingreso,Id_Emppleado,Cedula,Correo_Electronico,N_SeguroSocial,Empresa,Puesto,Id_Datos_Generales});
6     await newuser.save();
7     res.json(newuser);
8 }
9 exports.consultausuario = async (req,res)=>{
10     try {
11         const consultarusuario = await user.find();
12         res.json(consultarusuario);
13     }
14     catch (e){res.status(500).json('error de consulta',e);}
15 }
16 exports.consultausuarioid = async (req,res)=>{
17     try {
18         const {Fecha_Ingreso,Id_Emppleado,Cedula,Correo_Electronico,N_SeguroSocial,Empresa, Puesto,Id_Datos_Generales} = req.body;
19         console.log('id',req.params.id);
20         let usuarioid = await user.findById(req.params.id);
21         console.log('Los datos del usuario son:', usuarioid);
22         res.json(usuarioid);
23     }
24     catch (e){console.log(e);
25         res.status(500).json({msg:'error de consulta de id'});
26     }
27 }
28 exports.actualizaruser = async (req,res)=>{
29     try {
30         const {Fecha_Ingreso,Id_Emppleado,Cedula,Correo_Electronico,N_SeguroSocial,Empresa,Puesto,Id_Datos_Generales} = req.body;
```

```

30     console.log('id',req.params._id);
31     let usuarioid = await user.findById(req.params.id);
32     //console.log('Los datos del usuario son:', usuarioid);
33     res.json(usuarioid);
34     if(!usuarioid)
35     {
36       res.status(404).json(usuarioid);
37     }
38     else
39     {
40       usuarioid.Fecha_Ingreso = Fecha_Ingreso;
41       usuarioid.Id_Empleado = Id_Empleado;
42       usuarioid.Cedula = Cedula;
43       usuarioid.Correo_Electronico = Correo_Electronico ;
44       usuarioid.N_SeguroSocial = N_SeguroSocial;
45       usuarioid.Empresa = Empresa;
46       usuarioid.Puesto = Puesto;
47       usuarioid.Id_Datos_Generales = Id_Datos_Generales;
48       usuarioid = await user.findOneAndUpdate( filter: {
49         _id:req.params.id
50       },
51       usuarioid, options: {new:true}
52     );
53     res.json(usuarioid);
54   }
55 }
56 catch (e){res.status(500).send('error de consulta de id');}
57 }

58 exports.eliminarusuario = async (req,res)=>{
59   try {
60     //const {Fecha_Ingreso,Id_Empleado,Cedula,Correo_Electronico,N_SeguroSocial,Empresa,Puesto,Id_Datos_Generales} = req.body;
61     //console.log(req.params.id);
62     let usuarioid = await user.findById(req.params.id);
63     //console.log('Los datos del usuario son:', usuarioid);
64     res.json(usuarioid);
65     if(!usuarioid)
66     {
67       res.status(500).json('no existe usuario');
68     }
69     else
70     {
71       await user.findOneAndDelete( filter: {_id:req.params.id});
72       res.json('usuario eliminado');
73     }
74   }
75   catch (e){
76     console.log(e);
77     res.status(500).send('error de consulta de id');
78   }
79 }

```

routes\usuarios.js

```

package.json × basededatos.js × usuariocontrolador.js × routes\usuarios.js × models\usuarios.js × index.js × datos.json ×
1 const Router = require('express');
2 const router = Router();
3 //const user = require('../models/usuarios');
4 const usuariocontrolador = require('../controladores/usuariocontrolador');
5 router.post( path: '/',usuariocontrolador.insertarusuario);
6 router.get( '/ ',usuariocontrolador.consultausuario);
7 router.put( path: '/:id',usuariocontrolador.actualizaruser);
8 router.get( '/:id',usuariocontrolador.consultausuario);
9 router.delete( path: '/:id',usuariocontrolador.eliminarusuario);
10 module.exports = router;
11

```

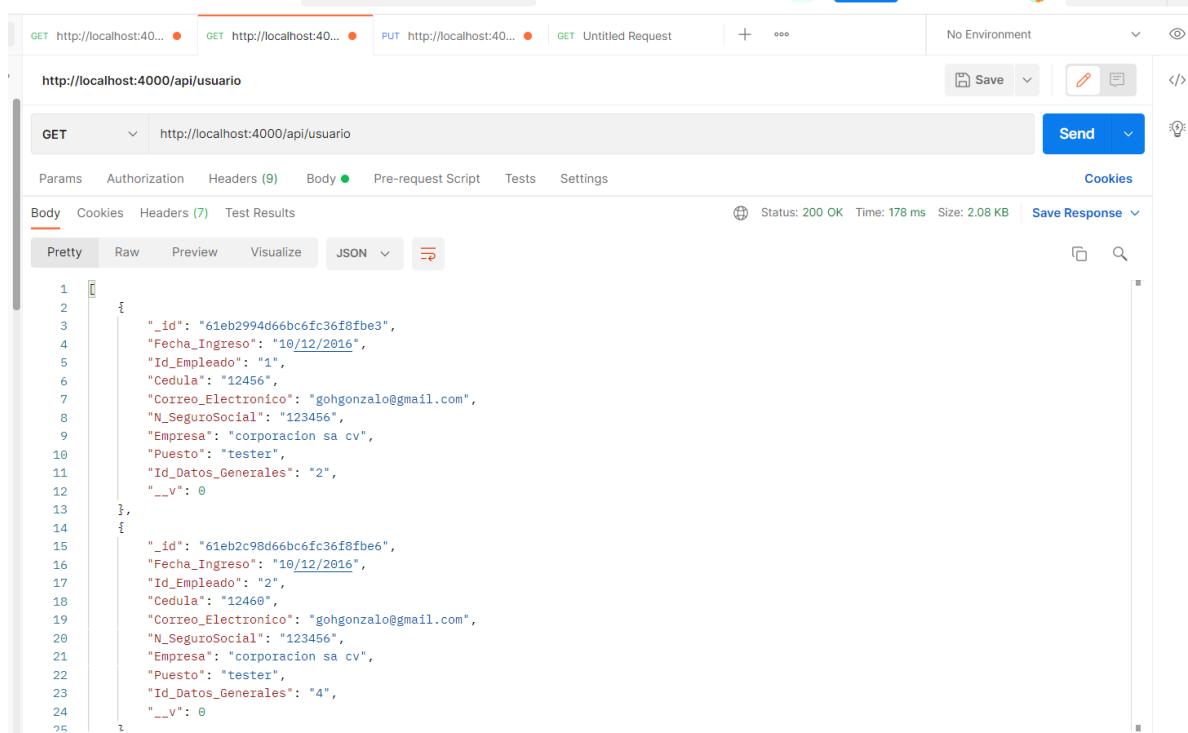
models\usuarios.js

```
1  const {Schema, model} = require('mongoose');
2  />const {date} = require("express/lib/String");
3  const esquemausuarios = new Schema( definition: {
4      'Fecha_Ingreso':{
5          prop:'Fecha_Ingreso',
6          type: String,
7          required: true
8      },
9      'Id_Empleado':{
10         prop:'Id_empleado',
11         type: String,
12         required: true
13     },
14     'Cedula':{
15         prop:'Cedula',
16         type: String,
17         required: true
18     },
19     'Correo_Electronico':{
20         prop:'Correo_Electronico',
21         type:String,
22         required: true
23     },
24     'N_SeguroSocial':{
25         prop:'N_SeguroSocial',
26         type:String,
27         required: true
28     },
29     'Empresa':{
30         'Empresa':{
31             prop:'Empresa',
32             type:String,
33             required: true
34         },
35         'Puesto':{
36             prop:'Puesto',
37             type:String,
38             required: true
39         },
40         'Id_Datos_Generales':{
41             prop:'Id_Datos_Generales',
42             type:String,
43             required: true
44         }
45     });
46 module.exports = model( name: 'usuario',esquemausuarios);
```

POSTMAN

una vez que se ha realizado el código con el que se va a manejar en la base de datos creamos nuestras pruebas en postman.

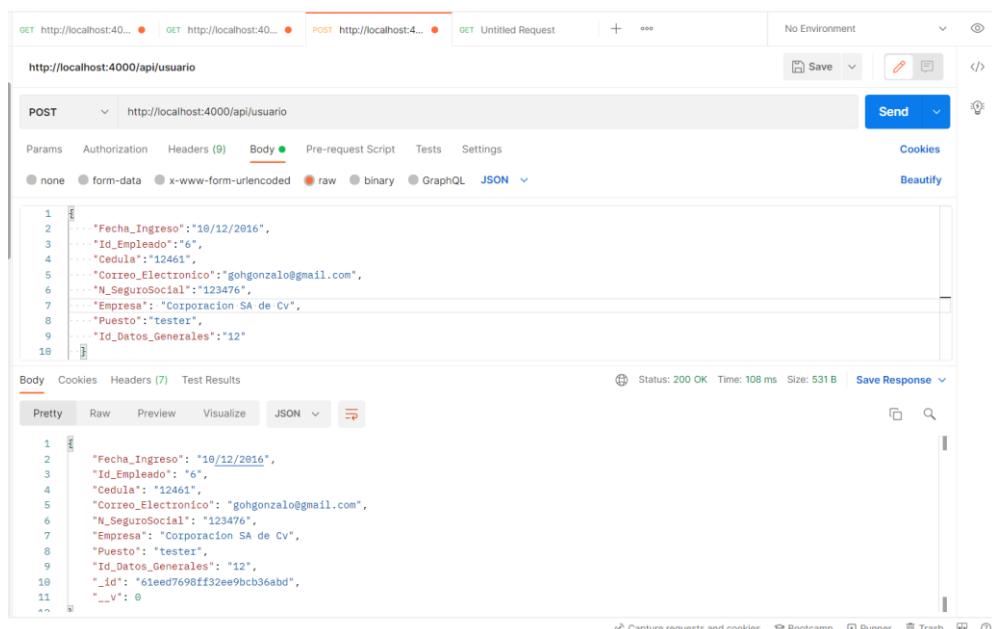
Se realizara primero una instrucción GET con la URL del puerto que se le ha asignado así como la ruta .



The screenshot shows the Postman interface with a successful GET request to `http://localhost:4000/api/usuario`. The response is a JSON array with two elements, each representing a user object:

```
1
2 [
3     {
4         "_id": "61eb2994d66bc6fc36f8fbe3",
5         "Fecha_Ingreso": "10/12/2016",
6         "Id_Epleado": "1",
7         "Cedula": "12456",
8         "Correo_Electronico": "gohgonzalo@gmail.com",
9         "N_SeguroSocial": "123456",
10        "Empresa": "corporacion sa cv",
11        "Puesto": "tester",
12        "Id_Datos_Generales": "2",
13        "__v": 0
14    },
15    {
16        "_id": "61eb2c98d66bc6fc36f8fbe6",
17        "Fecha_Ingreso": "10/12/2016",
18        "Id_Epleado": "2",
19        "Cedula": "12460",
20        "Correo_Electronico": "gohgonzalo@gmail.com",
21        "N_SeguroSocial": "123456",
22        "Empresa": "corporacion sa cv",
23        "Puesto": "tester",
24        "Id_Datos_Generales": "4",
25        "__v": 0
26    }
]
```

Se agregara nueva información esto lo haremos con la instrucción POST



The screenshot shows the Postman interface with a successful POST request to `http://localhost:4000/api/usuario`. The request body contains a JSON object representing a new user:

```
1
2 {
3     "Fecha_Ingreso": "10/12/2016",
4     "Id_Epleado": "6",
5     "Cedula": "12461",
6     "Correo_Electronico": "gohgonzalo@gmail.com",
7     "N_SeguroSocial": "123476",
8     "Empresa": "Corporacion SA de Cv",
9     "Puesto": "tester",
10    "Id_Datos_Generales": "12"
11}
```

The response is a JSON object with the newly added user and its ID:

```
1
2 {
3     "Fecha_Ingreso": "10/12/2016",
4     "Id_Epleado": "6",
5     "Cedula": "12461",
6     "Correo_Electronico": "gohgonzalo@gmail.com",
7     "N_SeguroSocial": "123476",
8     "Empresa": "Corporacion SA de Cv",
9     "Puesto": "tester",
10    "Id_Datos_Generales": "12",
11    "_id": "61eed7698ff32ee9bcb36abd",
12    "__v": 0
13}
```

Para actualizar información o modificarla usaremos la instrucción PUT junto con el id del dato que se modificará, y a continuación buscar nuevamente la información modificada con la instrucción GET y el id

The screenshot shows the Postman interface with two requests:

- PUT Request:** Targeted at `http://localhost:4000/api/usuario/61eed7698ff32ee9bcb36abd`. The Body is set to `JSON` and contains the following JSON payload:

```

1
2   ...
3     "Fecha_Ingreso": "10/12/2016",
4     "Id_Emppleado": "6",
5     "Cedula": "12461",
6     "Correo_Electronico": "gohgonzalo@gmail.com",
7     "N_SeguroSocial": "123476",
8     "Empresa": "SA de Cv",
9     "Puesto": "tester",
10    "Id_Datos_Generales": "12"

```

- GET Request:** Targeted at `http://localhost:4000/api/usuario/61eed7698ff32ee9bcb36abd`. The Body is set to `JSON` and retrieves the updated user data, which matches the PUT payload.

Para eliminar el dato se usará la instrucción DELETE junto con el id de la información a borrar

The screenshot shows two requests in the Postman interface:

DELETE Request:

- URL: `http://localhost:4000/api/usuario/61eed2071d2aa09d36cc38ba`
- Method: `DELETE`
- Body (Pretty):

```
1
2     "Fecha_Ingreso": "10/12/2016",
3     "Id_Epleado": "6",
4     "Cedula": "12461",
5     "Correo_Electronico": "gohgonzalo@gmail.com",
6     "N_SeguroSocial": "123476",
7     "Empresa": " SA de Cv",
8     "Puesto": "tester",
9     "Id_Datos_Generales": "12"
10
11
12
```
- Response Status: 200 OK

GET Request:

- URL: `http://localhost:4000/api/usuario/61eed2071d2aa09d36cc38ba`
- Method: `GET`
- Body (Pretty):

```
1
2     "_id": "61eed2071d2aa09d36cc38ba",
3     "Fecha_Ingreso": "10/12/2016",
4     "Id_Epleado": "6",
5     "Cedula": "12461",
6     "Correo_Electronico": "gohgonzalo@gmail.com",
7     "N_SeguroSocial": "123476",
8     "Empresa": " SA de Cv",
9     "Puesto": "tester",
10    "Id_Datos_Generales": "12",
11    "__v": 0
12
```
- Response Status: 200 OK

Cabe destacar que toda nuestra información está en MongoDB

The screenshot shows the MongoDB Compass application interface. On the left, the sidebar displays the cluster configuration: Local, HOSTS (clustermongo-shard-00-01, clustermongo-shard-00-02, clustermongo-shard-00-00), CLUSTER (Replica Set (atlas-118ff-s-3 Nodes)), and EDITION (MongoDB 4.4.11 Enterprise). The 'usuario.usuarios' collection is selected. The main pane shows the 'Documents' tab with 6 documents listed. A search bar at the top has the query '{ field: 'value' }'. Below it are buttons for ADD DATA, FILTER, VIEW, and various document operations. The results table includes columns for DOCUMENTS, STORAGE SIZE, AVG. SIZE, INDEXES, TOTAL SIZE, and AVG. SIZE. The documents are displayed as JSON objects:

```

[{"_id": "61e82d5add6bbc6fc36f0fbee", "Fecha_Ingreso": "10/12/2015", "Id_Empleado": "5", "Cedula": "12459", "Correo_Electronico": "gohgonzalo@gmail.com", "N_SeguroSocial": "123468", "Empresa": "corporacion sa cv", "Puesto": "tester", "Id_Datos_Generales": "8", "_v": 0}, {"_id": "61e82d5add6bbc6fc36f0fbee", "Fecha_Ingreso": "10/12/2015", "Id_Empleado": "6", "Cedula": "12460", "Correo_Electronico": "gohgonzalo@gmail.com", "N_SeguroSocial": "123472", "Empresa": "corporacion sa cv", "Puesto": "tester", "Id_Datos_Generales": "9", "_v": 0}, {"_id": "61e82d5add6bbc6fc36cc30ba", "Fecha_Ingreso": "10/12/2015", "Id_Empleado": "6", "Cedula": "12461", "Correo_Electronico": "gohgonzalo@gmail.com", "N_SeguroSocial": "123476", "Empresa": "Sa de Cv", "Puesto": "tester", "Id_Datos_Generales": "12", "_v": 0}

```

CONCLUSION

Al término de la creación del módulo podemos entender más a fondo el porqué del uso de las librerías en este tipo de proyectos, ya que gracias a ellos podemos trabajar de una forma más eficaz al implementarlo a un proyecto, en este caso que usamos la librería XLXS para leer un archivo Excel.