

Asignatura: Desarrollo de Aplicaciones Web

## Modelación de la Arquitectura del sistema

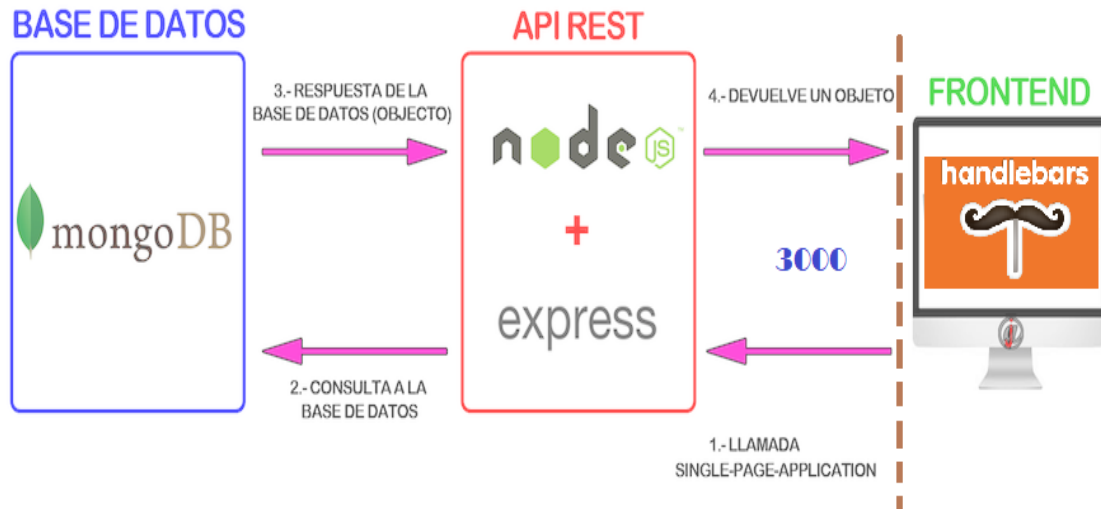


Fig. Arquitectura del Sistema

## Creación de base en Mongo Atlas

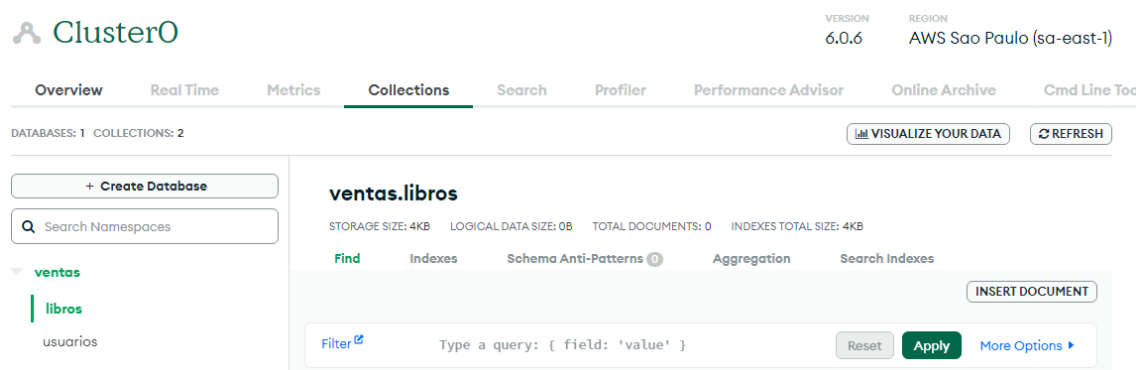


Fig. Base de Datos Ventas-libros

Integración de los miembros del equipo Team 2 a la base



## Asignatura: Desarrollo de Aplicaciones Web

Manage access to this project for users, teams, and API keys.













Users					
Teams					
API Keys					
Find a user					
Display Name	Email Address	Project Role	Created	Last Login	
ANGIE ELIZABETH DIAZ PEÑAFIEL	aediaz8@espe.edu.ec	Project Read Only, Project Data Access Admin, Project Cluster Manager	06/05/23 - 07:33:34 PM	06/21/23 - 10:53:15 PM	 
CAMILO ALEJANDRO ACOSTA MORENO	caacosta4@espe.edu.ec	Project Data Access Admin, Project Cluster Manager	06/02/23 - 01:12:10 PM	06/22/23 - 09:35:11 PM	 
LUIS MAURICIO BENAVIDES MINA	lmbenavides3@espe.edu.ec	Project Owner	06/04/23 - 08:30:25 PM	06/24/23 - 06:05:00 PM	 
NICOLAS ANDRE ALMEIDA LOGROÑO	naalmeida4@espe.edu.ec	Project Read Only, Project Data Access Admin, Project Cluster Manager	06/05/23 - 06:47:48 PM	06/21/23 - 11:11:18 PM	 
PAOLA NATALIA MORENO CARDOZO	pnmoreno@espe.edu.ec	Project Read Only	06/22/23 - 08:04:57 AM	06/22/23 - 08:04:58 AM	 
WENDY PIEDAD MORENO CARDOZO	wpmoreno1@espe.edu.ec	Project Data Access Admin, Project Cluster Manager	06/21/23 - 11:06:11 PM	06/24/23 - 03:29:51 PM	 

Fig. Usuarios que tienen acceso a la base

## Pruebas del Sistema

### Elaboración Frontend

#### Desarrollo Código Html, Css,

- Para el lado del cliente se utilizó Handlebars que es un sistema de plantillas Javascript basado en Mustache Templates. Handlebars sirve para generar HTML a partir de objetos con datos en formato JSON.
- Bootstrap que es un framework CSS que se utiliza para el desarrollo de aplicaciones front-end que se pueden adaptar a cualquier tipo de dispositivo.

```
src > views > layouts > main.hbs > html > body > ul > li > a
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Team2 Libreria</title>
8   <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.4.1/dist/css/bootstrap.min.css" int
9   <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.5/font/bootstrap-icons.css
10  <link rel="stylesheet" href="/css/main.css">
11 </head>
12 <body>
13   <ul>
14     <li><a href="/">Inicio</a></li>
15     <li><a href="/update">Venta</a></li>
16   </ul>
17   {{> navbar}}
18
19
20
21   {{{ body }}}
22 </body>
23 </html>
```

Fig. Acceso a bootstrap e Iconos



## Asignatura: Desarrollo de Aplicaciones Web

### Desarrollo código de las interfaces HTML

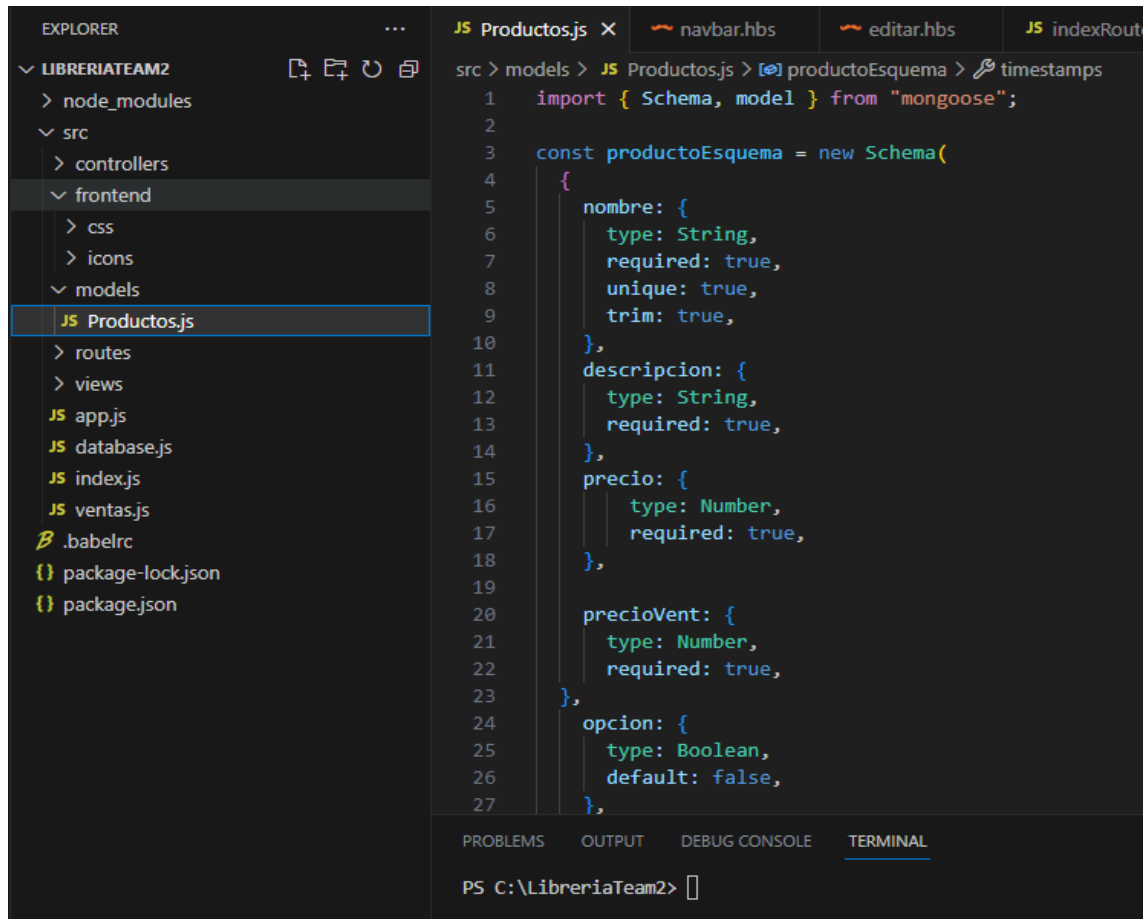
```
1 <div class="card card-body p-4 rounded-0 bg-dark text-white">
2
3 <h2>Agregar Libro <i class="bi bi-book"></i></h2>
4 <form action="/libros/agregar" method="post">
5   <div>
6     <label for="nombre">Escribe nombre del libro</label>
7     <input type="text" name="nombre" placeholder="Name book" id="nombre"
8       class="form-control bg-dark text-white"/>
9   </div>
10
11   <div>
12     <label for="descripcion">Descripción del libro</label>
13     <textarea name="descripcion" cols="25" rows="5" placeholder="Description book"
14       class="form-control bg-dark text-white"></textarea>
15   </div>
16
17   <div>
18     <label for="nombre">Escribe Precio Compra</label>
19     <input type="text" name="precio" placeholder="Price book" id="precio"
20       class="form-control bg-dark text-white"/>
21   </div>
22
23   <div>
24     <label for="nombre">Escribe Precio Venta</label>
25     <input type="text" name="precioVent" placeholder="Price book" id="precioVent"
26       class="form-control bg-dark text-white"/>
27   </div>
28 </div>
```

### Elaboración del backend

Se utilizó Node y JavaScript

Node.js, es un entorno en tiempo de ejecución multiplataforma para la capa del servidor en el lado del servidor basado en JavaScript.

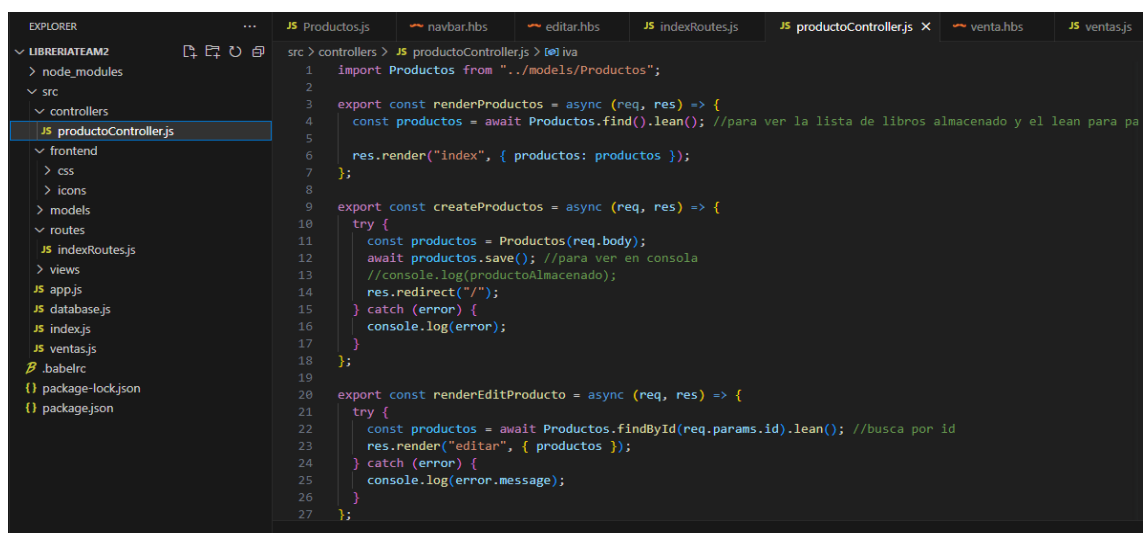
## Asignatura: Desarrollo de Aplicaciones Web



```
EXPLORER
  LIBRERIATEAM2
    node_modules
    src
      controllers
      frontend
      css
      icons
      models
        JS Productos.js
      routes
      views
      JS app.js
      JS database.js
      JS index.js
      JS ventas.js
      .babelrc
      package-lock.json
      package.json

src > models > JS Productos.js > [?] productoEsquema > timestamps
1  import { Schema, model } from "mongoose";
2
3  const productoEsquema = new Schema(
4    {
5      nombre: {
6        type: String,
7        required: true,
8        unique: true,
9        trim: true,
10     },
11     descripcion: {
12       type: String,
13       required: true,
14     },
15     precio: {
16       type: Number,
17       required: true,
18     },
19
20     precioVent: {
21       type: Number,
22       required: true,
23     },
24     opcion: {
25       type: Boolean,
26       default: false,
27     },
28   },
29   timestamps
30 );
```

Fig. Declaración de variables a implementarse en el sistema y guardarse en Mongo Atlas

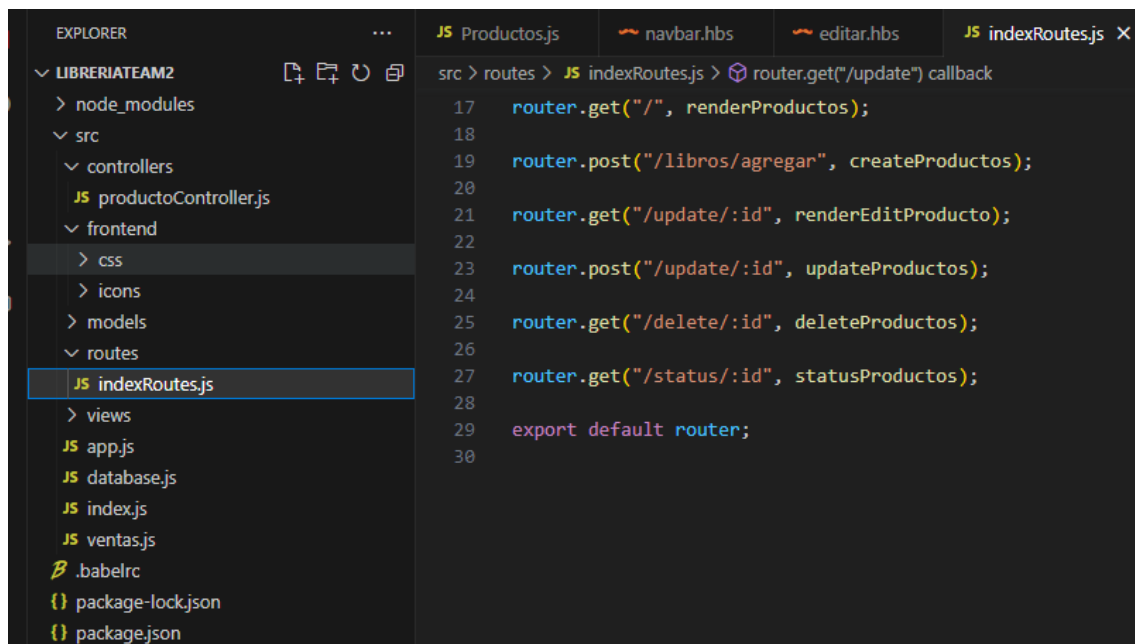


```
EXPLORER
  LIBRERIATEAM2
    node_modules
    src
      controllers
        JS productoController.js
      frontend
      css
      icons
      models
      routes
      JS indexRoutes.js
      views
      JS app.js
      JS database.js
      JS index.js
      JS ventas.js
      .babelrc
      package-lock.json
      package.json

src > controllers > JS productoController.js > [?] i va
1  import Productos from "../models/Productos";
2
3  export const renderProductos = async (req, res) => {
4    const productos = await Productos.find().lean(); //para ver la lista de libros almacenado y el lean para pa
5
6    res.render("index", { productos: productos });
7  };
8
9  export const createProductos = async (req, res) => {
10   try {
11     const productos = Productos(req.body);
12     await productos.save(); //para ver en consola
13     //console.log(productoAlmacenado);
14     res.redirect("/");
15   } catch (error) {
16     console.log(error);
17   }
18 };
19
20 export const renderEditProducto = async (req, res) => {
21   try {
22     const productos = await Productos.findById(req.params.id).lean(); //busca por id
23     res.render("editar", { productos });
24   } catch (error) {
25     console.log(error.message);
26   }
27 };
28
```

Fig. Controlador CRUD de los libros

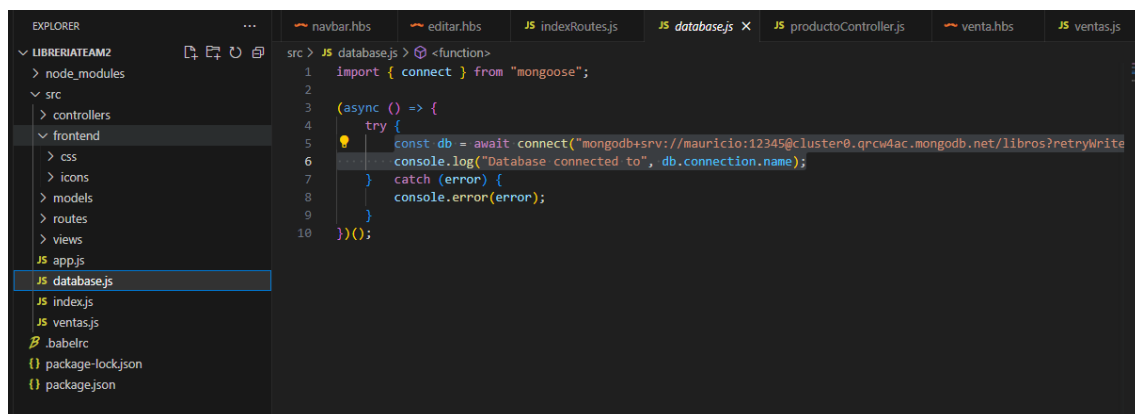
## Asignatura: Desarrollo de Aplicaciones Web



The screenshot shows the VS Code interface. The Explorer on the left displays the project structure for 'LIBRERIA TEAM2', with the 'routes' folder expanded and 'indexRoutes.js' selected. The main editor shows the content of 'indexRoutes.js', which defines several RESTful API endpoints for a CRUD application using Express.js.

```
src > routes > JS indexRoutes.js > router.get("/update") callback
17  router.get("/", renderProductos);
18
19  router.post("/libros/agregar", createProductos);
20
21  router.get("/update/:id", renderEditProducto);
22
23  router.post("/update/:id", updateProductos);
24
25  router.get("/delete/:id", deleteProductos);
26
27  router.get("/status/:id", statusProductos);
28
29  export default router;
30
```

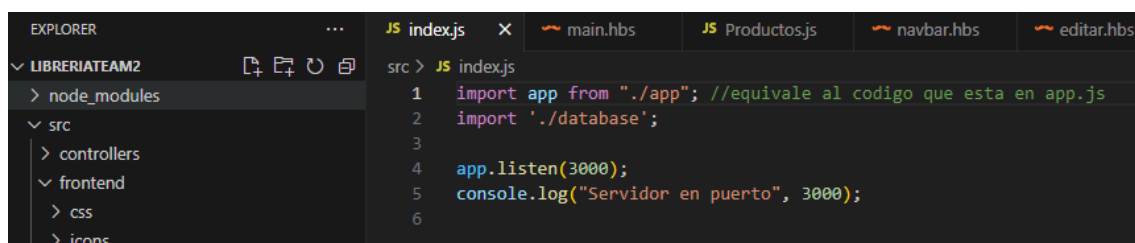
Fig. Rutas de acceso a las funciones del CRUD



The screenshot shows the VS Code interface with the 'database.js' file open in the editor. The file contains a function to establish a connection to a MongoDB database using the 'mongoose' library.

```
src > JS database.js > <function>
1  import { connect } from "mongoose";
2
3  (async () => {
4    try {
5      const db = await connect("mongodb+srv://mauricio:12345@cluster0.qrcw4ac.mongodb.net/libros?retryWrite=true");
6      console.log("Database connected to", db.connection.name);
7    } catch (error) {
8      console.error(error);
9    }
10 })();
```

Fig. Conexión a la base de Datos Atlas



The screenshot shows the VS Code interface with the 'index.js' file open in the editor. The file contains the code to start the Express.js web server, including importing the 'app' module and listening on port 3000.

```
src > JS index.js
1  import app from "../app"; //equivale al codigo que esta en app.js
2  import './database';
3
4  app.listen(3000);
5  console.log("Servidor en puerto", 3000);
6
```

Fig. Asignación de puerto a levantarse por medio de Express

**Asignatura:** Desarrollo de Aplicaciones Web

```
PS C:\LibreriaTeam2> npm run dev

> libreriateam2@1.0.0 dev
> nodemon src/index.js --exec babel-node

[nodemon] 2.0.22
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `babel-node src/index.js`
Servidor en puerto 3000
Database connected to libros
```

Fig. Ejecución del servidor y conexión a la Base de Datos Atlas

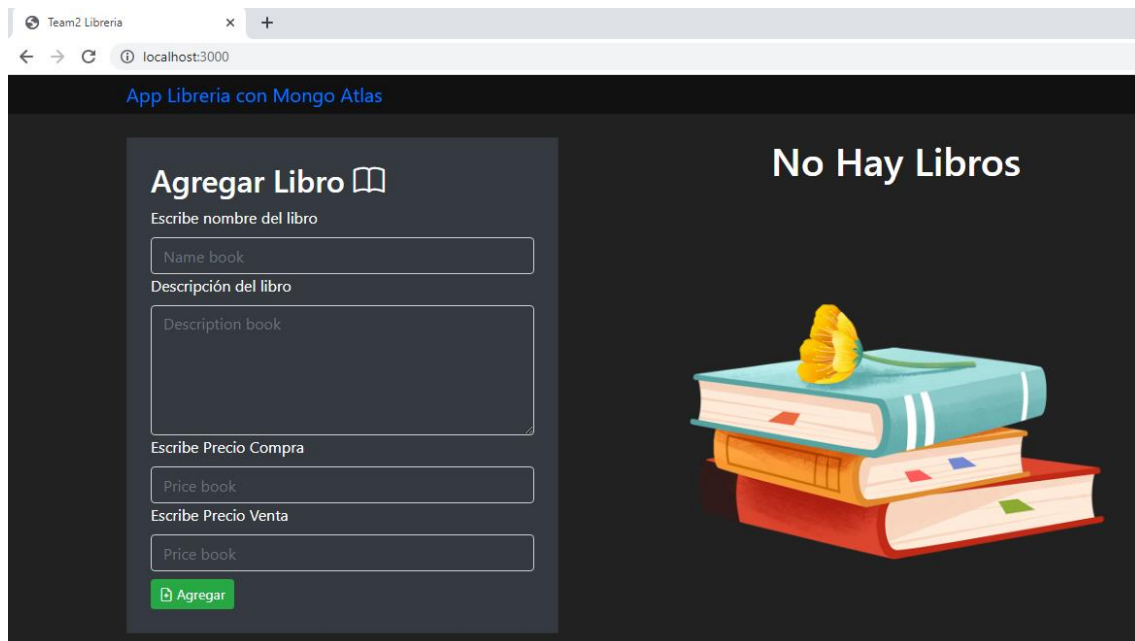


Fig. Interfaz Principal del Sistema- librería



Asignatura: Desarrollo de Aplicaciones Web

App Librería con Mongo Atlas

## Agregar Libro

Escribe nombre del libro

Descripción del libro

Matemáticas

Escribe Precio Compra

Escribe Precio Venta

 Agregar

Fig. Ingreso de Datos

App Librería con Mongo Atlas

## Agregar Libro


Escribe nombre del libro





Descripción del libro

Description book

Escribe Precio Compra

Escribe Precio Venta

 Agregar

Num. 	Nombre Libro 	Descripción 	Precio Compra \$	Precio Venta \$	Acciones 
0	Ecuaciones Diferenciales 2	Matemáticas	18	22	<div><div>Disponible</div><div>Actualizar</div><div>Eliminar</div></div>



Asignatura: Desarrollo de Aplicaciones Web

Fig. Visualización de Datos

The screenshot shows the Atlas data visualization interface. On the left, there is a sidebar with a tree view containing 'libros', 'tasks', 'users', and 'ventas'. The 'libros' item is selected. The main area displays a query result for a book. The query is: `{ field: 'value' }`. The result is a single document with the following fields: `_id` (ObjectId), `nombre` (Ecuaciones Diferenciales 2), `descripcion` (Matemáticas), `precio` (18), `precioVent` (22), `opcion` (false), `createdAt` (2023-06-25T00:32:16.201+00:00), and `updatedAt` (2023-06-25T00:32:16.201+00:00).

Fig. Registro de Datos en Atlas

## Actualizar Libro

Modificar libro:

Modificar descripción:

Modificar Precio Compra:

Modificar Precio Venta:

**Actualizar**

Fig. Actualizar Datos





Asignatura: Desarrollo de Aplicaciones Web





Num. 	Nombre Libro 	Descripción 	Precio Compra \$	Precio Venta \$	Acciones 
0	Ecuaciones Diferenciales 3	Matemáticas	22	27	<div>Disponible</div>

Fig. Disponibilidad del Libro





Num. 	Nombre Libro 	Descripción 	Precio Compra \$	Precio Venta \$	Acciones 
0	Ecuaciones Diferenciales 3	Matemáticas	22	27	<div>No Disponible</div>

Fig. No Disponibilidad del Libro