

BASE DE DATOS NoSQL

BASE DE DATOS II

Fecha de entrega[dd/mm/aaaa]

Elaborado por: Oscar Albuja

Carrera: ingeniería en Software

Dirigido a: Jorge Torres Rueda

OBJETIVO PROPUESTO DE ESTUDIO:

Demostrar el nivel de conocimientos para la gestión de Bases de Datos, Colecciones y Documentos en la Datos NoSQL – Mongo DB, desde una aplicación.

Ejercicios de desarrollo:

Enunciado:
Usando la base de datos Coop_Taxi (Tarea “ Bases de datos no SQL ”): <ol style="list-style-type: none">1. Evidenciar la conexión a la base de datos MONGO DB, desde una aplicación, con el lenguaje de programación de su preferencia2. Realizar un proceso o función dependiendo del lenguaje de programación seleccionado para:<ol style="list-style-type: none">a. Conexión a la base de datos MongoDBb. Inserción de documentos (uno o varios a la vez) en las colecciones indicadas.c. Actualización de datos de un documento, en las colecciones indicadasd. Actualización de datos de un campo, de un documento, en las colecciones indicadase. Eliminar documentos de las colecciones indicadas3. Se considerará un plus, la creación de un entorno grafico para la ejecución y visualización, de los procedimiento o funciones solicitados (no es mandatorio)
Argumentación:
<i>Justifica el uso del lenguaje de programación, utilizado para realizar el enunciado propuesto</i> <i>Explicación de:</i> <ol style="list-style-type: none">1. Elementos, variables o métodos para realizar la conexión y gestión de Inserción, Actualización y Eliminación de documentos en las colecciones, acorde al lenguaje seleccionado

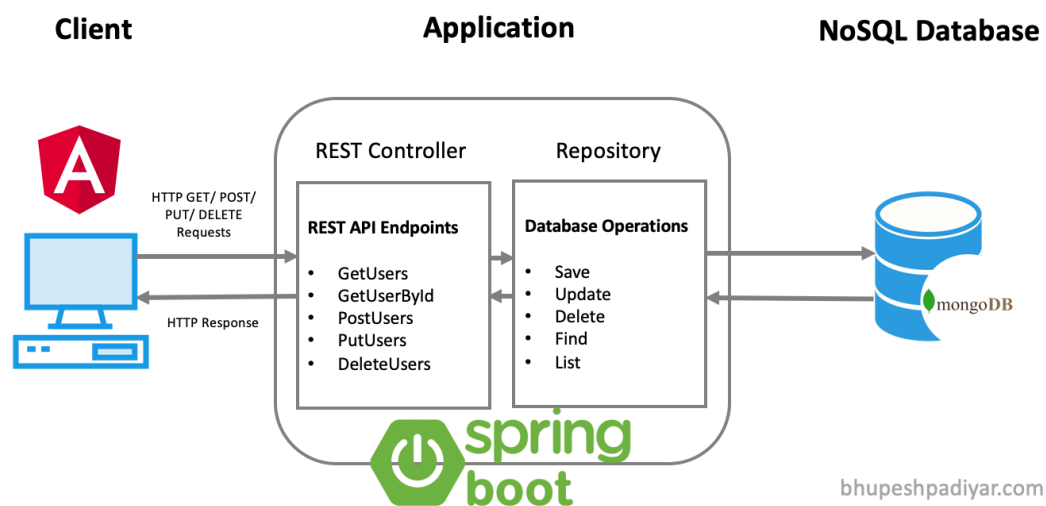
2. Las librerías, plugins, extensiones, objetos, etc, usados para la implementación y ejecución, de lo solicitado en el enunciado.

Scripts:

Introducción

La presente documentación describe el proceso de conexión entre una API REST implementada en Java utilizando el framework Spring Boot y una base de datos MongoDB. Se eligió Java con Spring Boot por su robusta gestión de dependencias y su capacidad para facilitar la creación de conexiones de bases de datos de forma sencilla y eficaz. Spring Data MongoDB, una parte integral de Spring Boot, proporciona la funcionalidad necesaria para interactuar con MongoDB de manera intuitiva, aprovechando la abstracción de datos y la inyección de dependencias.

Arquitectura



La imagen proporciona un esquema gráfico de la arquitectura de software utilizada en el proyecto, mostrando la sinergia entre el cliente, la aplicación y la base de datos. En el extremo del cliente, se representa Angular, un framework de desarrollo para construir interfaces de usuario dinámicas y de una sola página. El cliente interactúa con la aplicación a través de solicitudes HTTP tales como GET, POST, PUT y DELETE.

En el núcleo de la aplicación, Spring Boot actúa como servidor y gestiona las solicitudes entrantes a través de un conjunto de controladores REST API. Estos controladores definen los endpoints necesarios para realizar operaciones como obtener, publicar, actualizar y eliminar usuarios. La lógica de negocio se procesa aquí y luego se comunica con la capa de persistencia para operaciones de base de datos.

El lado derecho de la imagen muestra la base de datos MongoDB, una solución NoSQL que almacena datos en un formato similar a JSON, lo que la hace idónea para trabajar con aplicaciones basadas en JavaScript como Angular. El repositorio en la aplicación Spring Boot define la conexión con la base de datos y permite realizar operaciones de base de datos, como guardar, actualizar, borrar y listar registros.

Este diagrama refleja una arquitectura de aplicación completa y bien estructurada, donde el Frontend es responsable de la presentación y la interacción con el usuario, mientras que el Backend maneja la lógica de negocio, la seguridad y la persistencia de datos, operando como el enlace entre el cliente y la base de datos.

Contexto del Proyecto:

En el contexto del desarrollo de este proyecto, se tomó la decisión estratégica de reciclar y adaptar un proyecto previo. Esto se hizo con el fin de optimizar el tiempo y recursos disponibles, dada la complejidad inherente a la creación de una aplicación con un Backend en Spring Boot y un Frontend en Angular. La reutilización de código, patrones y estructuras ya probados y familiarizados permite acelerar el proceso de desarrollo sin sacrificar la calidad y robustez del producto final.

El enfoque de reciclaje no solo es una práctica eficiente en términos de tiempo y esfuerzo, sino que también contribuye a la consistencia y estabilidad del software, aprovechando patrones de diseño y soluciones que ya han sido validadas en un entorno de producción. Además, permite a los desarrolladores centrarse en la implementación de nuevas características y la mejora de la funcionalidad existente, en lugar de reconstruir componentes básicos desde cero.

Interfaz de Usuario para la Gestión de Socios Taxistas

Visión General de la Aplicación

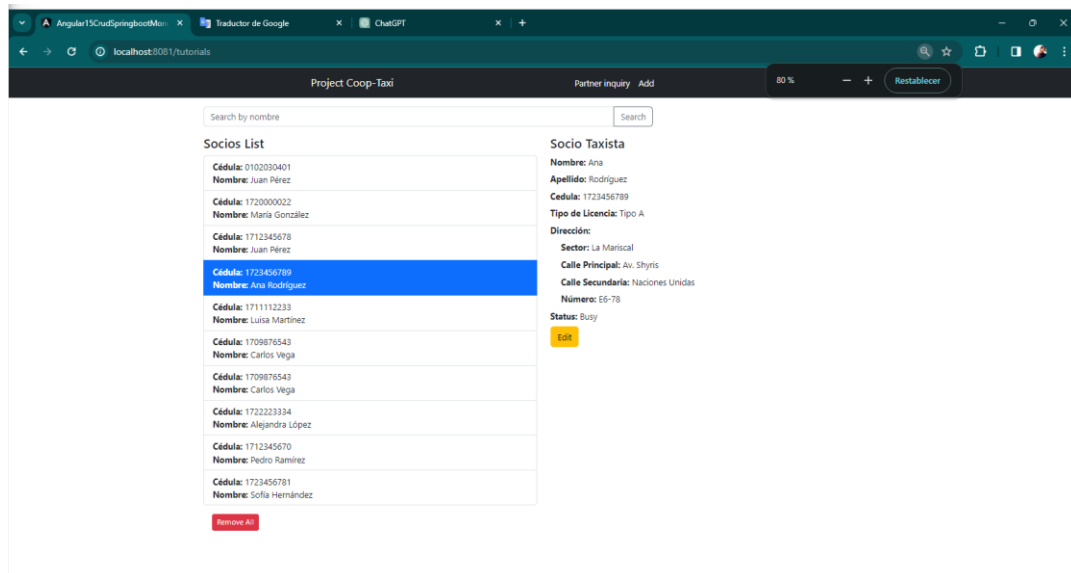


Ilustración 1 Ejecución del proyecto en el navegador

La aplicación diseñada proporciona una interfaz de usuario avanzada para la gestión integral de socios taxistas, facilitando operaciones de inserción, actualización, consulta y eliminación de documentos en una base de datos MongoDB. La armoniosa integración entre un Frontend desarrollado con el marco de trabajo Angular y un Backend construido utilizando Spring Boot, establece una comunicación fluida y efectiva, permitiendo una administración eficiente de los datos.

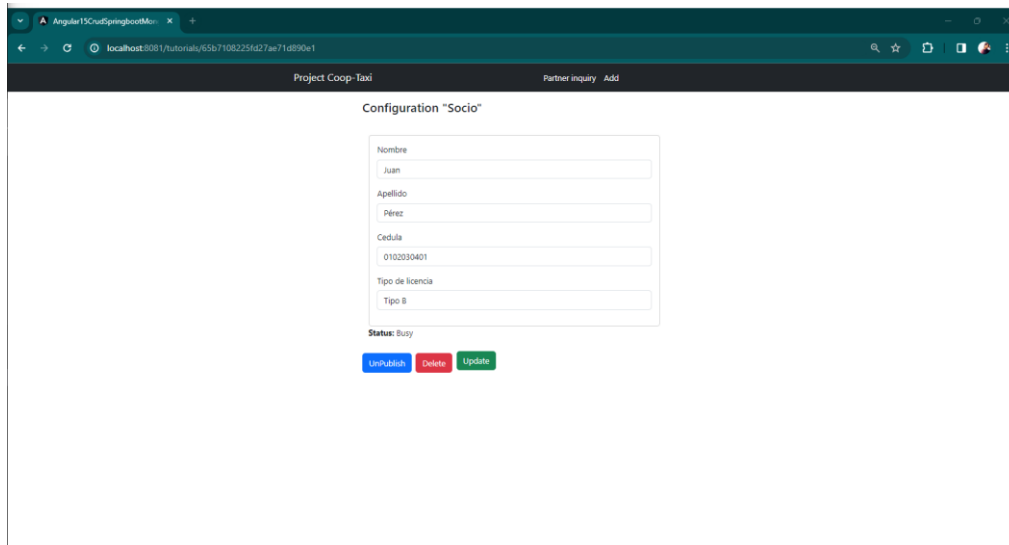
Listado y Detalles de Socios

La pantalla principal de la aplicación presenta un listado completo de socios. La interacción con cualquiera de los nombres listados despliega información detallada del socio seleccionado, como el nombre, apellido, cédula, tipo de licencia y detalles específicos de la dirección. Este mecanismo de expansión de información asegura que los usuarios puedan acceder rápidamente a los datos completos de cada socio sin sobrecargar la vista principal.

Indicador de Disponibilidad del Vehículo

En la descripción detallada del socio, se destaca un elemento crucial de la operativa diaria: un indicador de disponibilidad del vehículo. Un campo booleano en la base de datos controla este indicador, reflejando el estado actual del taxi asociado al socio, ya sea disponible o ocupado. Este detalle es de vital importancia para la coordinación operacional y la asignación eficiente de servicios.

Edición y Eliminación de Socios



The screenshot shows a web browser window with the URL `localhost:3001/tutoriales/55b7108225d27ae71d890e1`. The page title is "Project Coop-Taxi". The main content area is titled "Configuration 'Socio'". It contains a form with the following fields:

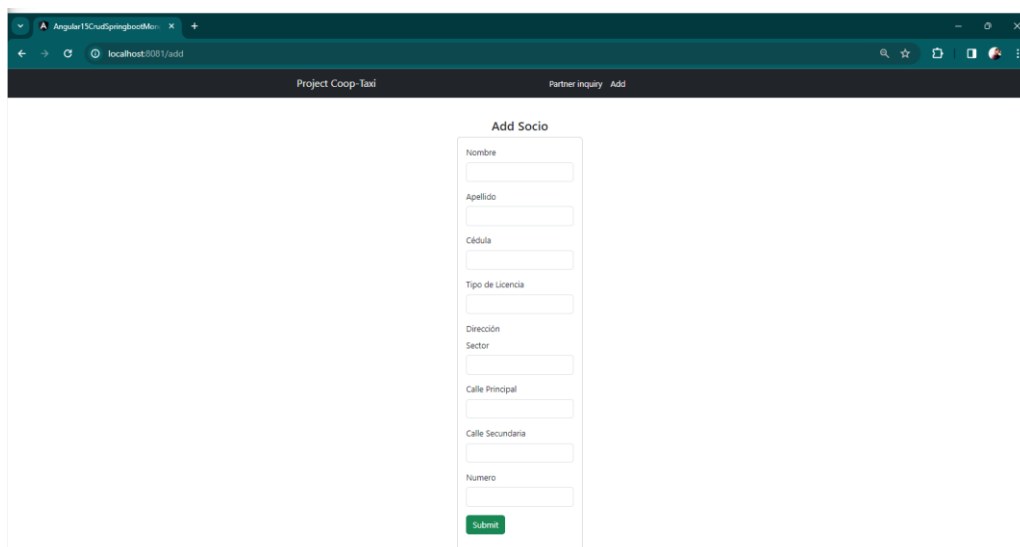
- Nombre: Juan
- Apellido: Pérez
- Cédula: 0102030401
- Tipo de licencia: Tipo B

Below the form, the status is "Status: Busy". At the bottom, there are three buttons: "UnPublish" (blue), "Delete" (red), and "Update" (green).

Ilustración 2 Configuration (CRUD) Socio Taxista

Cada perfil de socio en la interfaz detallada incluye un botón "Edit" que permite al usuario acceder a un formulario de edición. En esta sección, el usuario puede modificar la información del socio, excepto la dirección, la cual permanece como solo lectura por motivos de seguridad y protección de la privacidad. Además, un botón "Delete" posibilita la eliminación puntual del registro del socio, mientras que un botón "Remove All" en la pantalla principal ofrece una forma rápida de limpiar la lista de socios en su totalidad.

Adición de Nuevos Socios



The screenshot shows a web browser window with the URL `localhost:3001/add`. The page title is "Project Coop-Taxi". The main content area is titled "Add Socio". It contains a form with the following fields:

- Nombre
- Apellido
- Cédula
- Tipo de Licencia
- Dirección
- Sector
- Calle Principal
- Calle Secundaria
- Numero

At the bottom of the form is a green "Submit" button.

Ilustración 3 Agregar Socio Taxista

La funcionalidad de agregar nuevos socios es accesible mediante un botón "Add" situado en la barra de navegación. Este botón lleva al usuario a un formulario donde se deben ingresar todos los datos requeridos del socio, incluyendo la dirección completa. El botón "Submit" al pie del formulario captura y envía la información para su procesamiento y almacenamiento en la base de datos.

Notificaciones de Confirmación

La aplicación está configurada para presentar notificaciones post-operación, las cuales informan al usuario sobre el resultado de las acciones realizadas. Estos mensajes son un componente crítico en la retroalimentación del usuario, garantizando una interacción clara y efectiva con la aplicación.

Revisión y Consulta de Socios

La navegación a la sección "Partner inquiry" permite a los usuarios revisar y confirmar los cambios realizados. Este módulo es esencial para mantener una trazabilidad y supervisión de las operaciones CRUD efectuadas dentro de la aplicación.

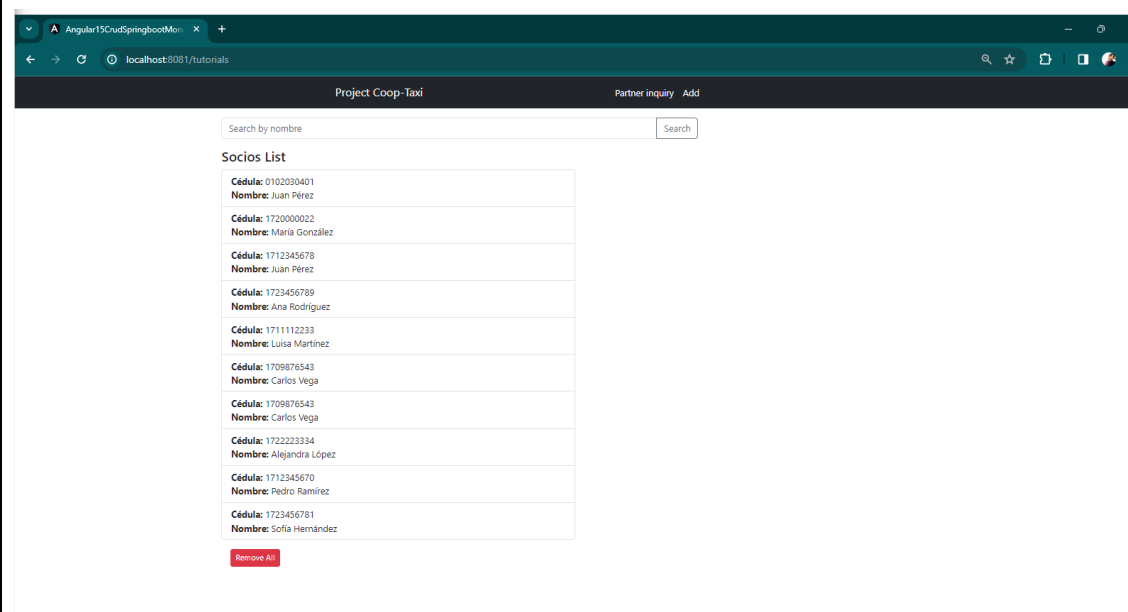


Ilustración 4 Revision cambios realizados Socio Taxista

Evidencia de la conexión a la base de datos MONGO DB, desde una aplicación, con el lenguaje de programación de su preferencia

Configuración de Conexión

La conexión a MongoDB se realiza mediante la especificación de propiedades en el archivo `application.properties` de Spring Boot. Las propiedades relevantes incluyen el host, el puerto, el

nombre de la base de datos y las credenciales si es necesario. Aquí se presenta un ejemplo de la configuración estándar:

```
spring.data.mongodb.uri=mongodb://username:password@localhost:27017/nombreBaseDatos
```

Pero como no se ha creado una instancia con usuario y contraseña la configuración de la conexión con la base de datos Coop_Taxi se la realiza de la siguiente manera:

```

application.properties X
spring-boot-server > src > main > resources > application.properties
1  spring.data.mongodb.database=Coop_Taxi
2  spring.data.mongodb.port=27017

```

Implementación de la Conexión

La API utiliza repositorios de Spring Data para manejar las operaciones de la base de datos. Estos repositorios extienden interfaces como `MongoRepository`, que proporcionan métodos CRUD para las entidades de la aplicación.

Descripción del Código (Backend):

Esta interfaz del repositorio es esencial para abstraer las operaciones de base de datos, permitiendo a los desarrolladores interactuar con la base de datos MongoDB de una manera simplificada y declarativa.

```

TutorialRepository.java X
spring-boot-server > src > main > java > com > bezkoder > spring > data > mongodb > repository > TutorialRepository.java > ...
1  package com.bezkoder.spring.data.mongodb.repository;
2
3  import java.util.List;
4
5  import org.springframework.data.mongodb.repository.MongoRepository;
6
7  import com.bezkoder.spring.data.mongodb.model.Tutorial;
8
9  public interface TutorialRepository extends MongoRepository<Tutorial, String> {
10     List<Tutorial> findByPublished(boolean published);
11     List<Tutorial> findByNombreContaining(String nombre);
12 }
13
14

```

- `MongoRepository<Tutorial, String>`: `MongoRepository` es una interfaz genérica que se especializa aquí para la entidad `Tutorial`. `Tutorial` es el tipo de la entidad y `String` es el tipo de su clave principal. Al extender `MongoRepository`, `TutorialRepository` hereda un

conjunto de operaciones CRUD básicas para manejar Tutorial entidades, incluyendo métodos como save, findAll, findById, y delete.

- *findByPublished(boolean published): Este es un método derivado de consulta que Spring Data MongoDB habilitará automáticamente. Busca y devuelve una lista de Tutorial entidades donde el campo published coincide con el parámetro booleano proporcionado. Este método es útil para filtrar tutoriales basados en su estado de publicación.*
- *findByNombreContaining(String nombre): Otro método derivado de consulta, que permite realizar búsquedas de texto parcial en el campo nombre de la entidad Tutorial. Este método devuelve una lista de tutoriales cuyo nombre contiene la cadena proporcionada, lo que es útil para la funcionalidad de búsqueda o filtro en la interfaz de usuario.*

Verificación de la Conexión

La conexión se verifica mediante el arranque exitoso de la aplicación y la ejecución de operaciones de base de datos sin errores. Los logs de Spring Boot proporcionan información detallada sobre el proceso de conexión y cualquier posible error.

CRUD desde la interfaz de usuario

Agregar

El proceso para incluir un nuevo socio comienza en la interfaz de usuario, donde se completa el formulario 'Añadir Socio' con los detalles requeridos. Una vez enviado el formulario, los datos se presentan en la sección 'Consulta de Socios', donde pueden ser revisados. Esta acción desencadena la inserción del nuevo registro en la base de datos MongoDB, lo cual se refleja en tiempo real en la interfaz 'Consulta de Socios', evidenciando la adición exitosa del nuevo socio en el sistema.

Project Coop-Taxi Partner inquiry Add

Add Socio

Nombre
Joaquin

Apellido
Albujá

Cédula
1721544045

Tipo de Licencia
Tipo B

Dirección
Atahualpa

Calle Principal
Gonzalo Cabezas

Calle Secundaria
Michelena

Numero
0995757367

Submit

Ilustración 5 Creando nuevo socio

Project Coop-Taxi Partner inquiry Add

Search by nombre Search

Socios List

Cédula: 0102030401	Nombre: Juan Pérez
Cédula: 1720000022	Nombre: Maria González
Cédula: 1712345678	Nombre: Juan Pérez
Cédula: 1723456789	Nombre: Ana Rodriguez
Cédula: 1711112233	Nombre: Luisa Martínez
Cédula: 1709876543	Nombre: Carlos Vega
Cédula: 1709876543	Nombre: Carlos Vega
Cédula: 1722233334	Nombre: Alejandra López
Cédula: 1712345670	Nombre: Pedro Ramirez
Cédula: 1723456781	Nombre: Sofia Hernández
Cédula: 1721544045	Nombre: Joaquin Albujá

Ilustración 6 Verificando la Creacion nuevo usuario un IU

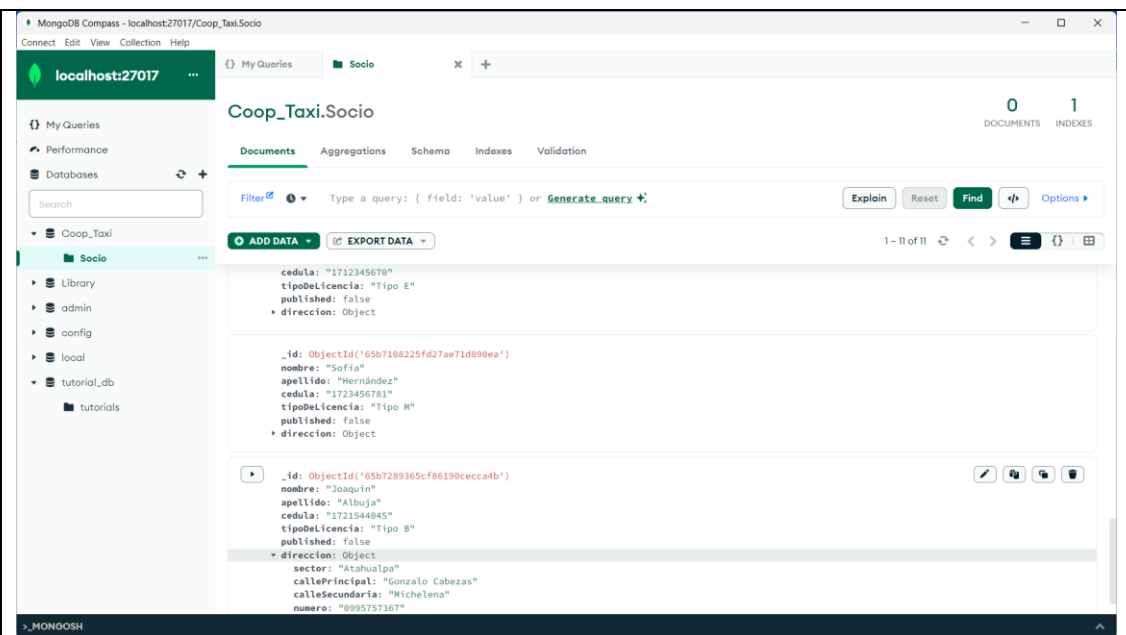


Ilustración 7 Verificando la creación en Mongo Compass

Editar

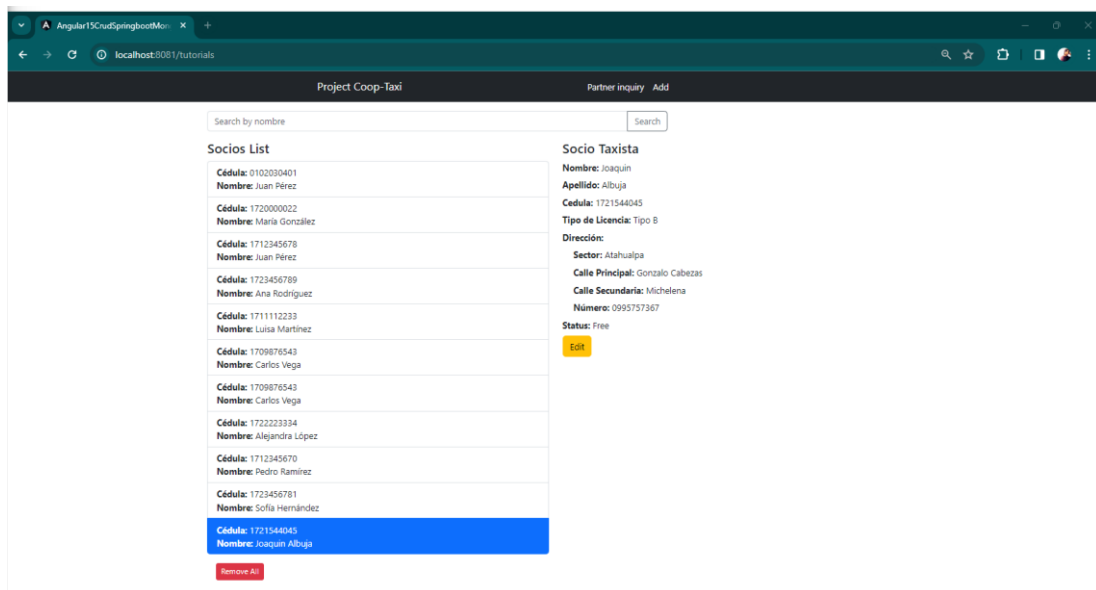


Ilustración 8 Estado actual antes de la edición del socio

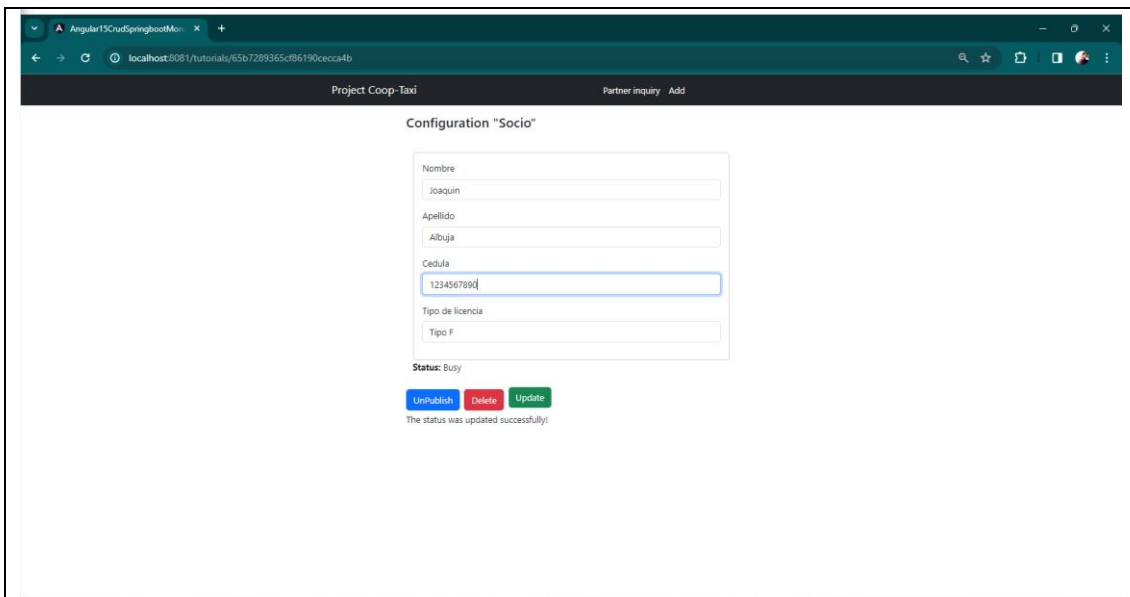


Ilustración 9 Editando Campos del socio

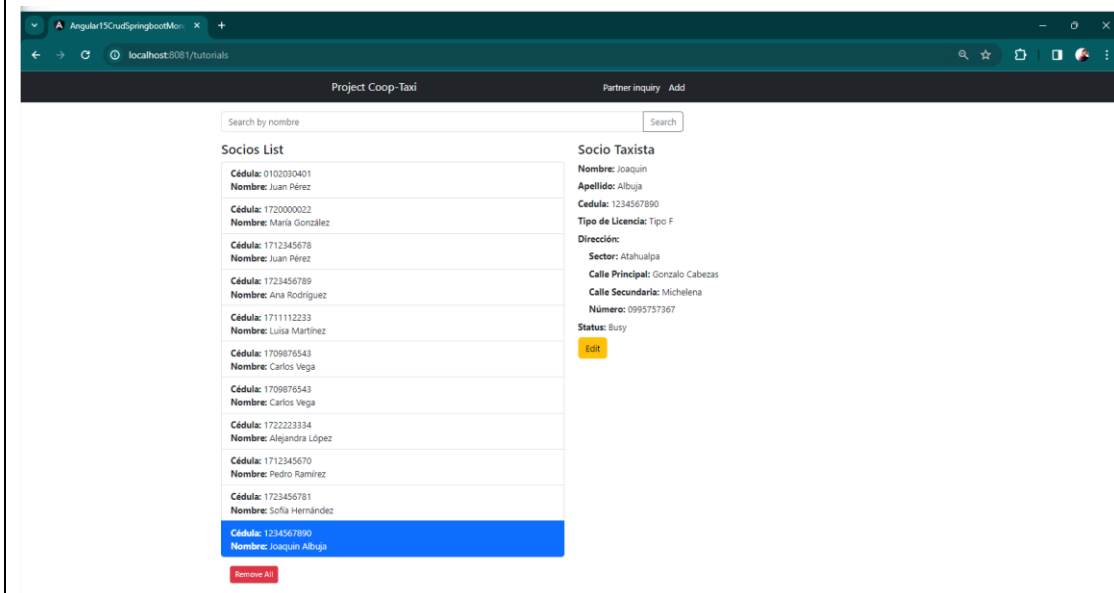


Ilustración 10 Socio Editado

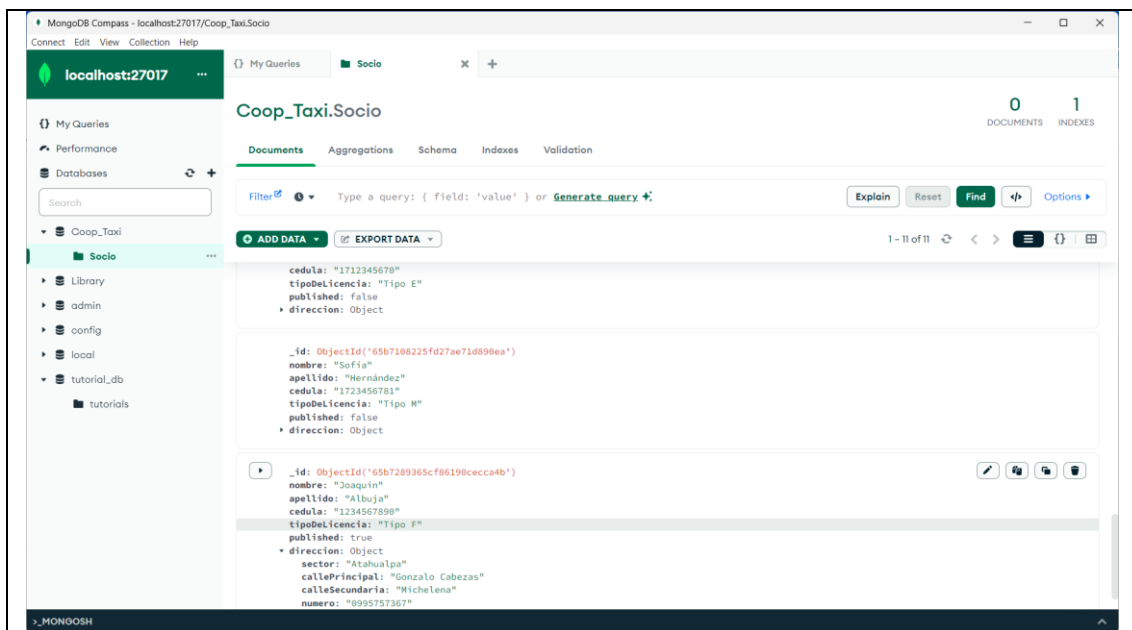


Ilustración 11 Mostrando al socio editado en Mongo Compass

Eliminar

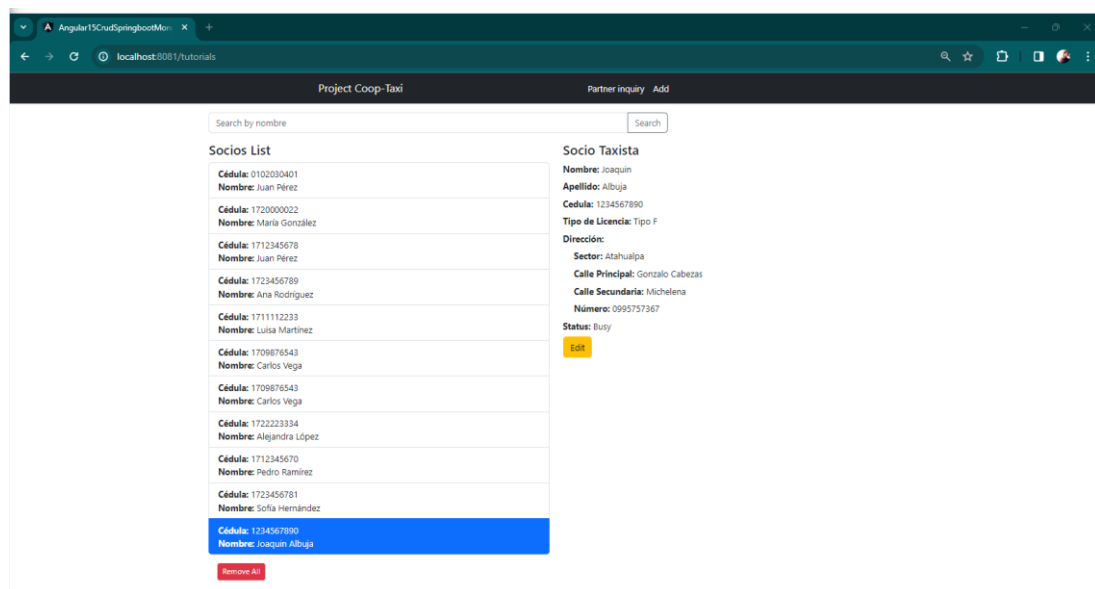
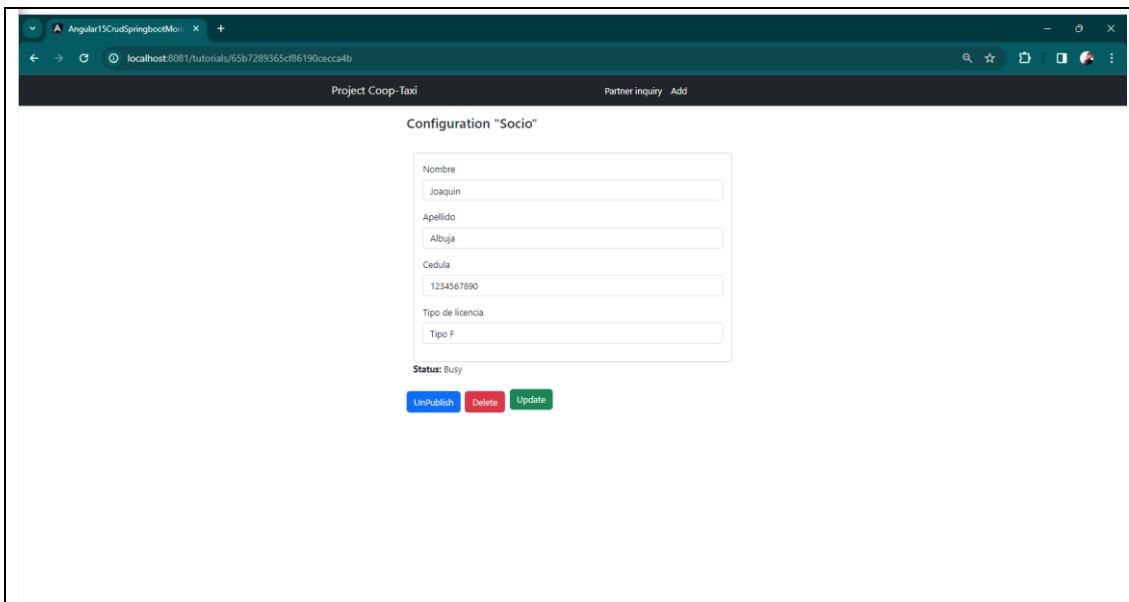


Ilustración 12 Estado Actual antes de eliminar



Project Coop-Taxi Partner inquiry Add

Configuration "Socio"

Nombre
Joaquín

Apellido
Albuja

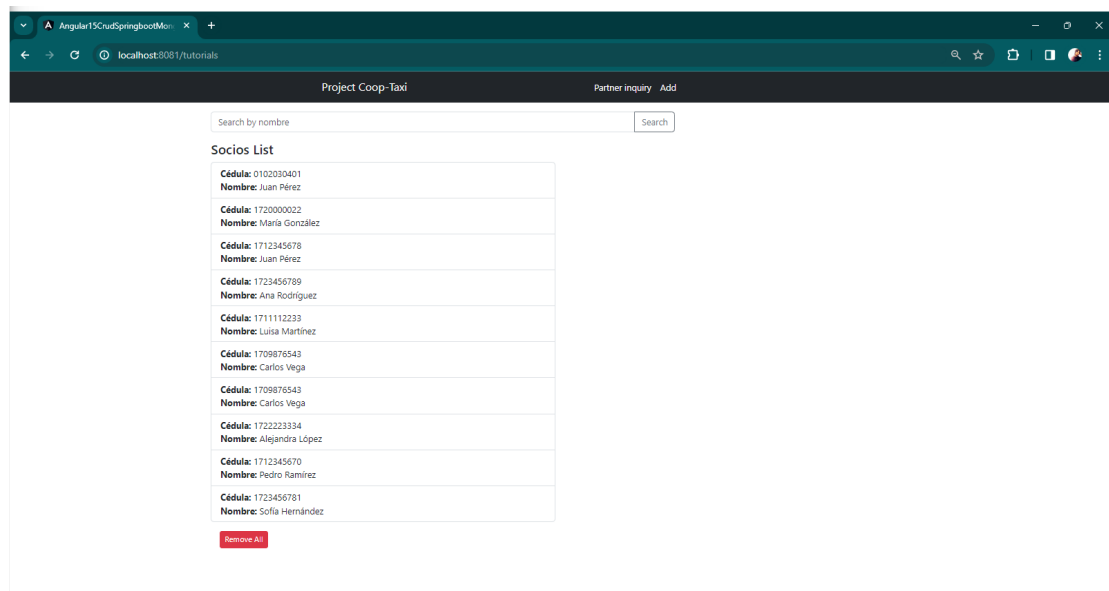
Cedula
1234567890

Tipo de licencia
Tipo F

Status: Busy

UnPublish Delete Update

Ilustración 13 Proceso de Eliminación



Search by nombre Search

Socios List

Cédula: 0102030401	Nombre: Juan Pérez
Cédula: 1720000022	Nombre: María González
Cédula: 1712345678	Nombre: Juan Pérez
Cédula: 1723456789	Nombre: Ana Rodríguez
Cédula: 1711112233	Nombre: Luisa Martínez
Cédula: 1709876543	Nombre: Carlos Vega
Cédula: 1709876543	Nombre: Carlos Vega
Cédula: 1722223334	Nombre: Alejandra López
Cédula: 1712345670	Nombre: Pedro Ramírez
Cédula: 1723456781	Nombre: Sofía Hernández

Remove All

Ilustración 14 Socio Eliminado

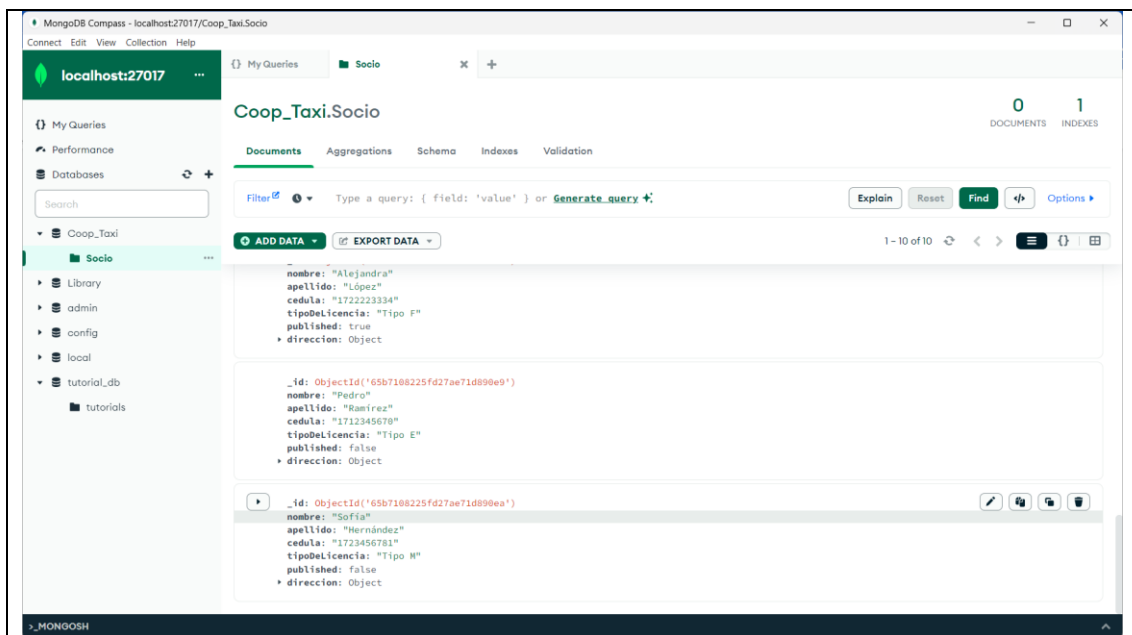


Ilustración 15 Socio Eliminado en Mongo Compass

Justificación del Uso de Java con Spring Boot y Angular

La elección de Java junto con el framework Spring Boot para el backend y Angular para el frontend se fundamenta en la robustez, escalabilidad y madurez de estas tecnologías. Java es un lenguaje de programación versátil y de tipado estático, ampliamente utilizado para construir aplicaciones empresariales seguras y de alto rendimiento. Spring Boot simplifica el proceso de configuración y despliegue de aplicaciones Java, proporcionando una amplia gama de herramientas para el desarrollo rápido y eficiente.

1. Conexión y Gestión de Documentos en MongoDB

Elementos y Métodos: Spring Boot utiliza Spring Data MongoDB, una librería que facilita la integración con bases de datos MongoDB. Los repositorios de Spring Data, como `MongoRepository`, proporcionan métodos predefinidos para realizar operaciones CRUD sin necesidad de escribir consultas complejas.

Variables: Se utilizan propiedades de configuración en el archivo `application.properties` de Spring Boot para establecer la URI de conexión a MongoDB, que incluye el host, puerto y credenciales de acceso.

2. Librerías y Herramientas Utilizadas

Spring Data MongoDB: Esta extensión de Spring Boot permite la mapeo automático de objetos Java a documentos MongoDB y viceversa, simplificando así las operaciones de inserción, actualización y eliminación de documentos.

Angular HTTP Client: En el frontend, el módulo HTTP Client de Angular es usado para realizar solicitudes HTTP asincrónicas al servidor. Permite una fácil integración con los endpoints REST definidos en el backend de Spring Boot.

Maven: Como sistema de gestión de proyectos y dependencias, Maven es utilizado para manejar las librerías de Spring Boot, facilitando la compilación y el empaquetado de la aplicación.

Node.js y npm: Estas herramientas son esenciales para la gestión de paquetes y el entorno de ejecución de Angular, respectivamente.

La integración de estas herramientas y tecnologías proporciona una solución completa para el manejo eficaz de datos en tiempo real, ofreciendo una experiencia de usuario fluida y coherente. La arquitectura de microservicios que promueve Spring Boot, junto con la estructura de componentes de Angular, asegura que la aplicación sea mantenible y extensible, adaptándose a las necesidades cambiantes de los usuarios y del negocio.

Link GitHub

<https://github.com/OAlbuja/ProyectoFinalMongoDB.git>

Resultado por acierto:

Captura la o las imágenes que del resultado del procedimiento realizado

Capturas de la consola de Mongo DB o del IDE Mongo Compass, una antes de ejecutar y otra después de realizar una inserción, actualización o eliminación