

Documentación


Enunciado: [Enunciado](#)

EJ1:

Seguir [Tutorial](#) de Spring Boot.

Instalo MongoDB en mi equipo

- Me descargo un proyecto pre iniciado de [spring initializr](#).

 **spring initializr**

Project
☐ Gradle - Groovy ☐ Gradle - Kotlin ☒ **Maven**

Language
☒ **Java** ☐ Kotlin ☐ Groovy

Spring Boot
☐ 4.0.0 (SNAPSHOT) ☐ 3.5.1 (SNAPSHOT) ☒ **3.5.0** ☐ 3.4.7 (SNAPSHOT)
☐ 3.4.6 ☐ 3.3.13 (SNAPSHOT) ☐ 3.3.12

Project Metadata

Group

com.example

Artifact

accessing-data-mongodb

Name

accessing-data-mongodb

Description

Demo project for Spring Boot

Package name

com.example.accessing-data-mongodb

Packaging

☒ **Jar** ☐ War

Java

☐ 24 ☐ 21 ☒ **17**

Dependencies

ADD

GENERATE

EXPLORE

...

it.spring.io

- inicio mongodb

```
C:\Users\rober>mongosh
Current Mongosh Log ID: 684193947640bda22c50eb66
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.2
Using MongoDB:      8.0.10
Using Mongosh:       2.5.2

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

-----
  The server generated these startup warnings when booting
  2025-06-05T14:53:30.479+02:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----

test>
```

```
C:\Users\rober>mongosh
Current Mongosh Log ID: 684193947640bda22c50eb66
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.2
Using MongoDB:      8.0.10
Using Mongosh:       2.5.2

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

-----
  The server generated these startup warnings when booting
  2025-06-05T14:53:30.479+02:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----

test>
```

- Creo una clase clientes

```
package com.example.accessing_data_mongodb;

import org.springframework.data.annotation.Id;

public class Cliente { no usages

    @Id 1 usage
    public String id;

    public String nombre; 2 usages
    public String apellido; 2 usages

    public Cliente() {} no usages

    public Cliente(String nombre, String apellido) { no usages
        this.nombre = nombre;
        this.apellido = apellido;
    }

    @Override
    public String toString() {
        return String.format(
            "Cliente[id=%s, nombre='%s', apellido='%s']",
            id, nombre, apellido);
    }
}
```

- Creo el repositorio para el cliente.

```
import java.util.List;
import org.springframework.data.mongodb.repository.MongoRepository;

public interface ClienteRepositorio extends MongoRepository<Cliente, String> {
    Cliente findByNombre(String nombre); no usages
    List<Cliente> findByApellido(String apellido); no usages
}
```

- Creo la clase principal para insertar y consultar datos.

```
@SpringBootApplication
public class AccessingDataMongodbApplication implements CommandLineRunner {

    @Autowired    6 usages
    private ClienteRepositorio repositorio;

    public static void main(String[] args) {
        SpringApplication.run(AccessingDataMongodbApplication.class, args);
    }

    @Override    no usages
    public void run(String... args) throws Exception {
        repositorio.deleteAll();

        // Guarda algunos clientes
        repositorio.save(new Cliente( nombre: "Ana",  apellido: "García"));
        repositorio.save(new Cliente( nombre: "Luis",  apellido: "García"));

        // Obtener todos los clientes
        System.out.println("Clientes encontrados con findAll():");
        for (Cliente cliente : repositorio.findAll()) {
            System.out.println(cliente);
        }

        // Buscar cliente por nombre
        System.out.println("\nCliente encontrado con findByNombre('Ana'):");
        System.out.println(repositorio.findByNombre("Ana"));

        // Buscar clientes por apellido
        System.out.println("\nClientes encontrados con findByApellido('García'):");
        for (Cliente cliente : repositorio.findByApellido("García")) {
            System.out.println(cliente);
        }
    }
}
```

- inicio la aplicación, y recibo la respuesta esperada.

```
Clientes encontrados con findAll():
Cliente[id=684197413e49eac27a28cfbc, nombre='Ana', apellido='García']
Cliente[id=684197413e49eac27a28cfbd, nombre='Luis', apellido='García']

Cliente encontrado con findByNombre('Ana'):
Cliente[id=684197413e49eac27a28cfbc, nombre='Ana', apellido='García']

Clientes encontrados con findByApellido('García'):
Cliente[id=684197413e49eac27a28cfbc, nombre='Ana', apellido='García']
Cliente[id=684197413e49eac27a28cfbd, nombre='Luis', apellido='García']
```

EJ2:

Elige una de las bases de datos NoSQL mencionadas en clase (como Cassandra, Redis, Neo4j...) y haz un documento **resumen de 2 páginas** con:

- Características principales de esa base de datos.
- Diferencias respecto a las bases de datos SQL tradicionales.

[Redis](#)

EJ3:

Haz una prueba de acceso a otra base de datos NoSQL distinta de MongoDB desde Java o Spring Boot. Puedes usar:

- **Redis (clave-valor en memoria):**
 - [Tutorial Redis 1](#)
 - [Tutorial Redis 2 \(Baeldung\)](#)