DGTIC UNAM
DIRECCIÓN GENERAL DE CÓMPUTO Y DE TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN

REDEC UNAM

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
DIRECCIÓN GENERAL DE CÓMPUTO Y DE TECNOLOGÍAS
DE INFORMACIÓN Y COMUNICACIÓN

DIPLOMADO

Desarrollo de sistemas con tecnología Java

Módulo 8

Persistencia con Spring Data

Dr. Omar Mendoza González

omarmendoza564@aragon.unam.mx

MongoRepository

Es una interfaz de Spring Data que proporciona métodos CRUD y operaciones personalizadas para trabajar con MongoDB de manera sencilla y eficiente.

Extiende las funcionalidades de **CrudRepository** y **PagingAndSortingRepository**.

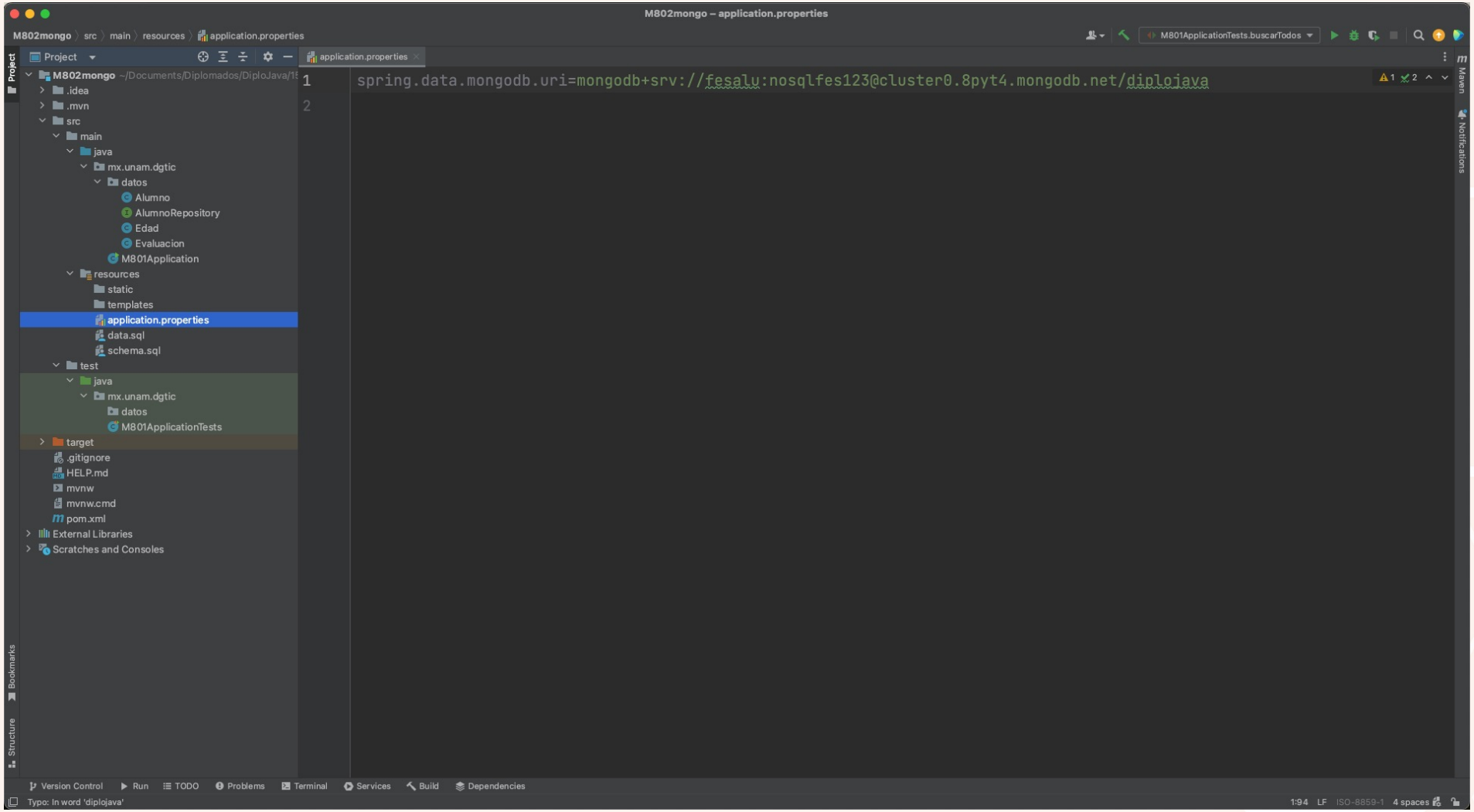
Para usar MongoRepository solo es necesario definir una interfaz que extiende de **MongoRepository**:

MongoRepository

- Para conectarse a MongoDB en una aplicación Spring Boot, la conexión se define en el archivo **application.properties** utilizando la propiedad **spring.data.mongodb.uri**

- spring.data.mongodb.uri=mongodb+srv://usuario:passwd@cluster.mongodb.net/basededatos

MongoRepository



MongoRepository

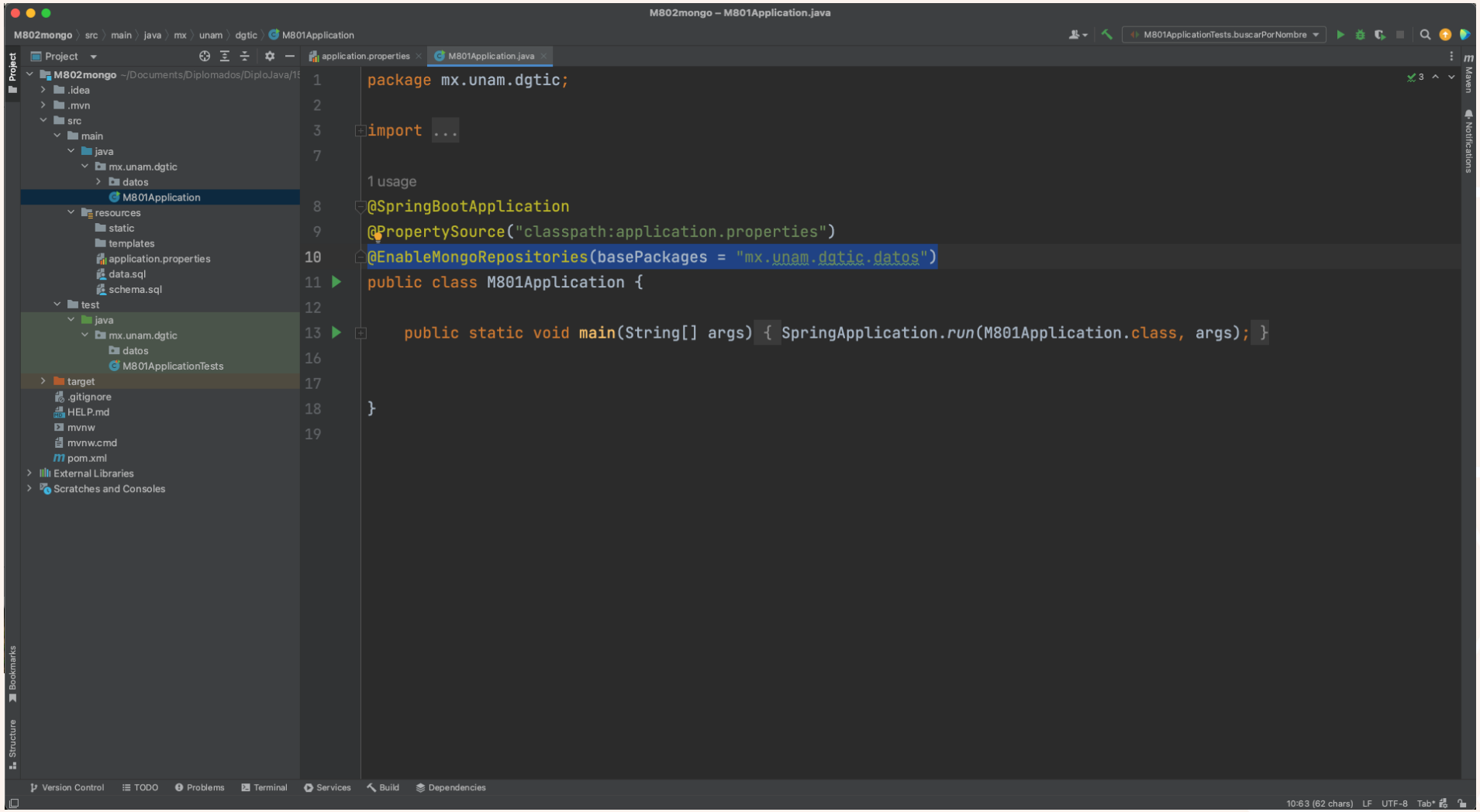
- Spring Data MongoDB facilita la interacción con MongoDB utilizando repositorios basados en interfaces, como MongoRepository.
- No es necesario escribir código para la conexión manual, ya que Spring Boot crea automáticamente el **MongoTemplate** o **MongoRepository** con base en la configuración proporcionada.

@EnableMongoRepositories

Es una anotación de Spring Data que habilita la creación de repositorios de MongoDB dentro de un proyecto Spring Boot.

Esta anotación permite a Spring escanear un paquete específico en busca de interfaces que extienden **MongoRepository** u otras interfaces de repositorio relacionadas con MongoDB.

@EnableMongoRepositories



@Document

Define que una clase Java debe ser persistida en una colección de MongoDB.

Al aplicar **@Document** a una clase, la clase se convierte en un documento en MongoDB, y cada instancia de la clase será un registro en esa colección.

Si no se especifica un nombre, Spring usará el nombre de la clase como el nombre de la colección en MongoDB.

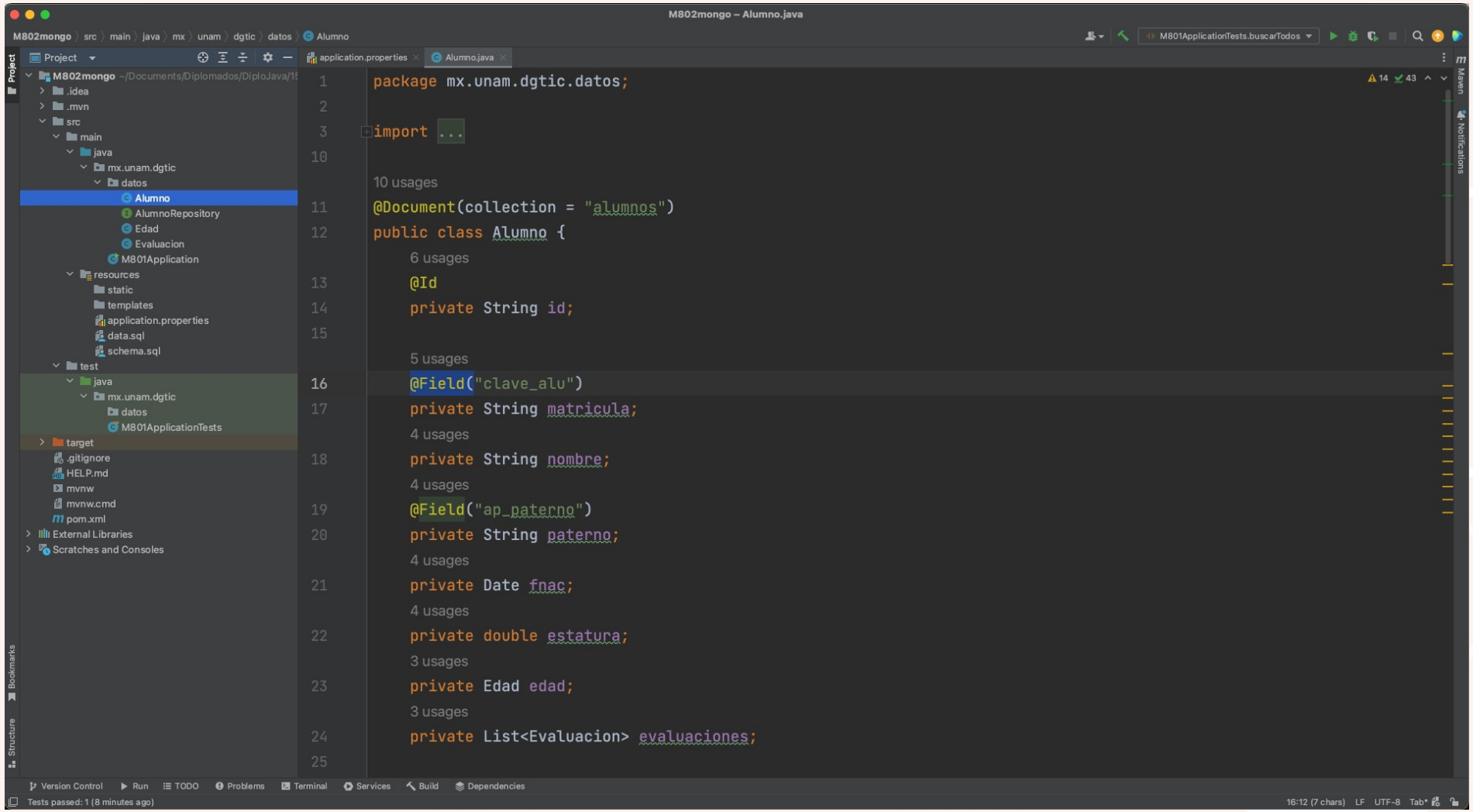
@Field

Se utiliza para mapear un campo de una clase Java a un campo de un documento en MongoDB, es útil cuando el nombre del atributo en la clase difiere del nombre del campo en el documento de MongoDB.

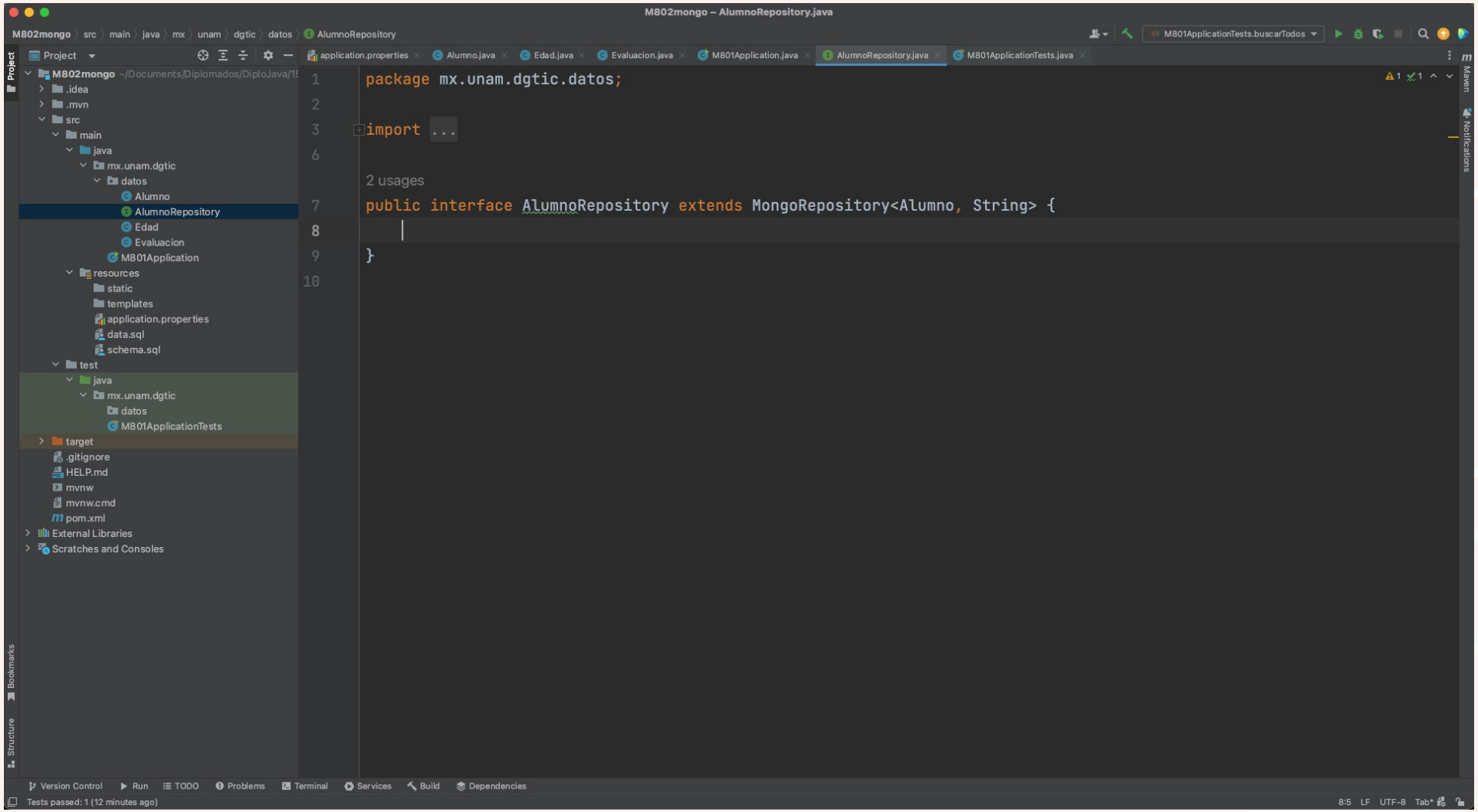
Si no se utiliza la anotación @Field, Spring Data MongoDB asume que el nombre del atributo en la clase Java es igual al nombre del campo en MongoDB.

Es útil para establecer un nombre de campo diferente en el documento de MongoDB o para manejar casos especiales de serialización.

Alumno


M802mongo – Alumno.java
M802mongo src main java mx unam dgtic datos Alumno
M801ApplicationTests.buscarTodos
Project
application.properties
Alumno.java
M802mongo ~/Documents/Diplomados/DiploJava/1
.idea
.mvn
src
main
java
mx.unam.dgtic
datos
Alumno
AlumnoRepository
Edad
Evaluacion
M801Application
resources
static
templates
application.properties
data.sql
schema.sql
test
java
mx.unam.dgtic
datos
M801ApplicationTests
target
.gitignore
HELP.md
mvnw
mvnw.cmd
pom.xml
External Libraries
Scratches and Consoles
1 package mx.unam.dgtic.datos;
2
3 import ...
10
10 usages
11 @Document(collection = "alumnos")
12 public class Alumno {
6 usages
13 @Id
14 private String id;
15
5 usages
16 @Field("clave_alu")
17 private String matricula;
4 usages
18 private String nombre;
4 usages
19 @Field("ap_paterno")
20 private String paterno;
4 usages
21 private Date fnac;
4 usages
22 private double estatura;
3 usages
23 private Edad edad;
3 usages
24 private List<Evaluacion> evaluaciones;
25
Version Control
Run
TODO
Problems
Terminal
Services
Build
Dependencies
Tests passed: 1 (8 minutes ago)
16:12 (7 chars)
LF
UTF-8
Tab*
Maven
Notifications
Bookmarks
Structure

MongoRepository



MongoTemplate

Es una clase central de **Spring Data MongoDB** que proporciona una API de alto nivel para interactuar con MongoDB.

Es la implementación principal para las operaciones de persistencia en MongoDB y actúa como una plantilla para realizar operaciones como inserción, consulta, actualización y eliminación de documentos en MongoDB.

MongoTemplate

- **Operaciones CRUD**
- **Consultas avanzadas**
  - Soporta la creación de consultas personalizadas utilizando **Query** y **Criteria** para filtrar documentos.
- **Manejo de colecciones**
  - Puedes crear, eliminar o verificar la existencia de colecciones en la base de datos.
- **Soporte para agregaciones**
  - Puedes realizar consultas de agregación en MongoDB.
- **Manejo de transacciones**
  - A partir de MongoDB 4.x, soporta transacciones multi-documento en bases de datos replicadas.

MongoTemplate vs MongoRepository

Característica	**MongoTemplate**	**MongoRepository**
Nivel de abstracción	Bajo nivel (más control)	Alto nivel (más simplificado)
Uso principal	Proporciona un API programática para operaciones personalizadas y complejas en MongoDB.	Se utiliza para operaciones CRUD básicas y consultas personalizadas con métodos de repositorio.
Operaciones CRUD	Se requiere escribir el código explícito para las operaciones CRUD utilizando métodos como find(), insert(), update(), remove().	CRUD se maneja automáticamente con métodos como save(), findAll(), deleteById().
Consultas personalizadas	Ofrece un control total sobre las consultas mediante **Query** y **Criteria**. Ideal para consultas complejas o dinámicas.	Soporta consultas personalizadas utilizando convenciones de nombres o anotaciones como @Query.
Transacciones	Soporta transacciones multi-documento a partir de MongoDB 4.x.	Generalmente no maneja transacciones de forma explícita. Se usa principalmente para operaciones sencillas.

MongoTemplate vs MongoRepository

Característica	MongoTemplate	MongoRepository
Agregaciones	Soporta operaciones de agregación y consultas complejas utilizando métodos como aggregate().	No soporta agregaciones directamente. Para esto es necesario utilizar MongoTemplate.
Flexibilidad	Muy flexible para realizar operaciones avanzadas, como pipelines de agregación, manejo de índices, y manipulación avanzada de documentos.	Menos flexible, está más enfocado en operaciones CRUD básicas.
Código requerido	Requiere más código y configuración para realizar operaciones.	Requiere menos código, ya que muchas operaciones CRUD se generan automáticamente.
Uso en proyectos grandes	Se recomienda cuando se necesita mayor control sobre las consultas o cuando se necesitan operaciones complejas.	Ideal para proyectos más simples, donde las operaciones básicas CRUD son suficientes.
Configuración	Debe ser inyectado manualmente (por ejemplo, mediante @Autowired).	Automáticamente detectado y configurado por Spring Data si extiende MongoRepository.

Contacto

Dr. Omar Mendoza González

omarmendoza564@aragon.unam.mx