**Documentación Java**

**Carlos Ortiz y Diego García**

Índice

[Aplicación pedida 1](#_Toc197162402)

[Tecnologías utilizadas 1](#_Toc197162403)

[Estructura de la Aplicación 1](#_Toc197162404)

[Database 2](#_Toc197162405)

[Model 2](#_Toc197162406)

# Aplicación pedida

Se pidió el desarrollo de una aplicación de gestión de relaciones con clientes (CRM) utilizando Java y JDBC para conectarse a una base de datos relacional para gestionar comerciales, clientes y facturas, aplicando operaciones CRUD sobre cada entidad.

# Tecnologías utilizadas

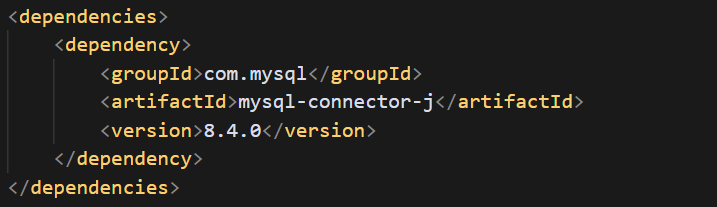
* Java
* MySQL
* JBDC
* Swing

# Estructura de la Aplicación

La aplicación consta de varios paquetes con diferentes clases o interfaces que hace que funcione como un todo, además de un Main donde se ejecuta todo. Los paquetes osn los siguientes:

## Database

Aquí se realiza la conexión a la base de datos de MySQL poniendo la bd, usuario y contraseña, además de un método para realizar la conexión. Esto se logra gracias a que el proyecto esta hecho en Maven con un pom.xml con la siguiente dependencia que nos permite conectarnos a una base de datos MySQL



## Model

Se guardan cada entidad de la base de datos con sus atributos correspondientes, con un constructor lleno y vacío para la realización de operaciones, además de los métodos tipos de getters, setters y To String.

## Repository

Se almacenan interfaces para cada entidad de la base de datos con los métodos que se desarrollaran, siendo findAll(mostrar todo de una entidad), findById(Busca un elemento específico de la entidad por su id), save(para añadir un actualizar un elemento de una entidad) y delate(eliminar un elemento de una entidad)

Cada método es de un tipo distinto: findAll es una List de esa entidad, findbyId es del tipo de la entidad, recibiendo como parámetro un Long que es el id de esa entidad, y tanto save como delate son void, aunque reciben parámetros distintos (save la entidad y delete un Long id).

## Controller

Se guardan clases para cada entidad que implementan su interfaz Repository específica para desarrollar sus métodos.

Para cada clase, primero se realiza un constructor con la conexión a la base de datos, y luego un método create el cual recibe como parámetro un Resultset(contiene resultados de una secuencia SQL), en el que se crea una nueva entidad y se le asignan los valores de sus atributos con los datos del Resultset(cabe destacar que si la entidad tiene atributos los cuales son foreing keys de otra entidad, en el propio método se debe crear y asignar los valores de los atributos de dicha otra entidad también (Ej. Categoría en Producto)