**CONNECTION OF JAVA BACKEND WITH DB (MongoDB)**

ORM: Object-Relational Mapping

* ORM is a technique used to map Java object to database tables.
* It allows developers to work with databases using Object-Oriented Programming concepts, making it easier to interact with relational databases.
* (Operation will be performed in Class and Database will be updated behind the scenes!)
* Consider a JAVA class User and database table *users*. ORM frameworks like Hibernate can map the fields in the User class to columns in the *user* table, making it easier to insert, update, retrieve, and delete records.

JPA: Java Persistence API ***(ONLY USED IN RELATIONAL DATABASES AND NOT IN MONGODB)***

* A way to achieve ORM, includes interfaces and annotations that you use in your Java classes, requires a persistence provider (ORM tools) for implementation.

Persistence Provider / ORM Tools:

* To use JPA, you need a persistence provider. A persistence provider is a specific implementation of the JPA specification. Examples of JPA persistence providers include Hibernate, EclipseLink, and OpenJPA. These providers implement the JPA interfaces and provide the underlying functionality to interact with databases.

Spring Data JPA:

* It is built on top of JPA (Java Persistence API) specification, but it is not a JPA implementation itself. Instead, it simplifies working with JPA by providing higher-level abstractions and utilities. However, to use Spring Data JPA effectively, you still need a JPA implementation, such as Hibernate, EclipseLink, or another JPA-compliant provider, to handle those actual database interactions.

JPA is primarily designed for working with relational databases, where data is stored in tables with a predefined schema. MongoDB, on the other hand, is a NoSQL database that uses a different data model, typically based on collections of documents, which are schema-less or have flexible schemas. This fundamental difference in data models and storage structure is why JPA IS NOT USED WITH MONGODB!

In the case of MongoDB, you don’t have a traditional JPA persistence provider. MongoDB is a NoSQL database, and **Spring Data MongoDB** serves as the “persistence provider” for MongoDB. It provides the necessary abstractions and implementations to work with MongoDB in a spring applications. **(DEPENDENCY)**

**Query Method DSL** and **Criteria API** are two different ways to interact with a database when using Spring Data JPA for relational databases and Spring Data MongoDB for MongoDB databases.

Spring Data JPA is a part of the Spring Framework that simplifies data access in Java applications, while Spring Data MongoDB provides similar functionality for MongoDB.

**Query Method DSL:** is a simple and convenient way to create queries based on method naming conventions, while the **CRITERIA API** offers a more dynamic and programmatic approach for building complex and custom queries.