**WEEK 3**

**Spring Data JPA with Spring Boot, Hibernate**

Exercise 2: Difference between JPA, Hibernate and Spring Data JPA

**JPA (Java Persistence API)**

JPA is a Java specification that defines how Java objects are mapped to relational database tables. It is not an actual implementation but rather a set of interfaces and annotations that describe standard ways to persist, update, delete, and retrieve data from a database using Java objects. It includes important concepts like @Entity, @Id, @OneToMany, and the EntityManager interface for managing persistence operations.

Since JPA is just a specification, it requires a concrete implementation to actually perform the operations it defines. It offers portability because you can switch between implementations without rewriting your code, as long as you follow standard JPA.

**Hibernate**

Hibernate is the most popular implementation of JPA. It not only fulfills all JPA requirements but also provides additional features that go beyond the JPA specification. These include lazy loading strategies, caching, advanced mapping techniques, custom SQL support, and more.

Developers often use Hibernate directly (with its native APIs like Session and SessionFactory) or through JPA interfaces. Hibernate handles the actual interaction with the database, translating the JPA annotations and operations into SQL and managing connections, transactions, and object states.

**Spring Data JPA**

Spring Data JPA is a higher-level abstraction built on top of JPA, provided by the Spring Framework. Its main goal is to simplify the development of data access layers by reducing boilerplate code. Instead of writing implementations for DAO (Data Access Object) interfaces, you can extend interfaces like JpaRepository or CrudRepository and let Spring Data auto-generate the implementation at runtime.

Spring Data JPA also supports query methods by naming convention, pagination, sorting, and custom query definitions using JPQL or native SQL. It typically works under the hood with a JPA provider like Hibernate, making it easier for developers to focus on business logic instead of persistence logic.