JPA (Java Persistence API):

What is it?

JPA is a specification (an interface standard) from Java EE (Jakarta EE)
Think of it as *a set of rules* that says *"this is how Java objects should map to database tables."*

What does it provide?

- annotations like `@Entity`, `@Id`
- standard `EntityManager` interface
- defines the *contract*, but not the implementation.

Key point
JPA **does not implement** anything by itself — it just defines *how* persistence should work.

Hibernate:

What is it?
Hibernate is a **popular implementation of JPA**.

- It *implements* all the interfaces specified by JPA
- Adds its own powerful features on top of JPA (like caching, advanced query capabilities)

Think of it like

JPA = driving rules
Hibernate = the car that actually drives according to those rules

You can use Hibernate directly
with its own SessionFactory etc., but usually you use it as the JPA provider.

Spring Data JPA:

What is it?
A Spring project that makes working with JPA even easier.
It builds on top of JPA + Hibernate (or another provider)
and:

- auto-wires repositories
- removes boilerplate code
- allows you to define interfaces like:
- **Example:**

```
interface UserRepository extends JpaRepository<User,
Long> { }
```

- and Spring Data JPA automatically creates the implementation.

Key advantage:
You write *almost no code* for standard CRUD
It handles the dirty work for you