EnitityManagerFactory

https://www.javatpoint.com/jpa-entity-manager

https://stackoverflow.com/questions/5640778/hibernate-sessionfactory-vs-jpa-entitymanagerfactory#:~:text=Using%20EntityManagerFactory%20approach%20allows%20us,be%20found%20here%20and%20here

The `@PrePersist` and `@PostPersist` annotations are part of the Java Persistence API (JPA) and are used to intercept entity lifecycle events in JPA entities. These annotations allow you to execute custom logic before and after the persistence operation (e.g., saving an entity to the database).

1. **@PrePersist**:
   - The `@PrePersist` annotation is used to specify callback methods that are invoked before an entity is inserted into the database.
   - You can use this annotation to perform operations such as setting default values, initializing fields, or validating data before the entity is persisted.
   - Methods annotated with `@PrePersist` must be public, have no arguments, and return void.
   - Example:

     ```java
     import javax.persistence.PrePersist;

     public class Product {
         @PrePersist
         public void prePersist() {
             // Logic to execute before entity persistence
         }
     }
     ```

2. **@PostPersist**:
   - The `@PostPersist` annotation is used to specify callback methods that are invoked after an entity has been inserted into the database.
   - You can use this annotation to perform operations such as logging, sending notifications, or updating related entities after the entity has been persisted.
   - Methods annotated with `@PostPersist` must be public, have no arguments, and return void.
   - Example:

     ```java
     import javax.persistence.PostPersist;
     ```

```
public class Product {
    @PostPersist
    public void postPersist() {
        // Logic to execute after entity persistence
    }
}
```

These annotations are useful for implementing custom behavior or executing additional logic at specific points in the lifecycle of JPA entities. They provide hooks for developers to integrate application-specific functionality seamlessly with the persistence mechanism provided by JPA.