Database Access with Hibernate/JPA CRUD

What is hibernate?

* A framework for persisting or saving and retrieving Java objects (data) in a database.
* Often used for saving and retrieving data in a database

A blue square with white text

Description automatically generated with medium confidence

* Benefits:
  + Handles all the low-level SQL code.
  + Minimizes the amount of JDBC code you must develop.
  + Provides the Object-to-Relational Mapping (ORM)
* Object-to-Relational Mapping (ORM)
  + The dev defines the mapping between the Java class and the database table. (The dev tells hibernate how your java class or object maps to the data in the database using a table).
  + The mapping can be setup via a config file, XML, or using Java annotations.

A blue square with white text

Description automatically generated with medium confidence

What is JPA?

* The Jakarta persistence API was previously known as the Java Persistence API.
* Standard API for Object-to-Relational Mapping (ORM)
* It’s only a specification that defines a set of interfaces and requires an implementation to be usable.
* Hibernate is the most popular and is also the default vendor implementation for JPA used in spring boot.
* What are the benefits of using JPA?
  + By having a standard API you’re not locked into the vendor’s implementation.
  + Maintain portable, flexible code by coding to JPA spec (interfaces)
  + Theoretically switch vendor implementations
    - If vendor A stops supporting their product you can switch to vendor B without a vendor lock-in and with very few code changes just a configuration change.
* Saving a Java object with JPA
  + Use the new keyword to create an instance of an object.
  + Save the object to the database by using the JPA helper object “entityManager.persist()”
  + Pass in our object.
  + JPA will take the Java Object based on the defined mappings and store it in the appropriate table.
  + Hibernate pretty much does all the work for us.
* Retrieving a Java Object with JPA
  + Find the object in the database by using the JPA helper object “entityManager.find()”
  + Pass in the object and its id or pk.
  + JPA will then look at the given table name and the pk given and find the object.
* How does Hibernate / JPA relate to JDBC?
  + Hibernate/JPA actually uses JDBC for all database communications so Hibernate/JPA is just another layer of abstraction on top of JDBC.