# BOOKSJPA AND BOOKSDATAJPA

EVER HINOJOSA AGUIRRE

# BOOKSJPA

- A BASIC CRUDS SHOWING THE USAGE  OF THE JPA USING A DAO WHICH BRINGS MORE FLEXIBILITY WHEN IMPLEMENTING MY QUERIES IN THE BOAIMPL

BookController.java | Book.java | BookDAO.java

BooksJPA ▸ src/main/java ▸ com.academia.BooksJPA.dao ▸ BookDAO ▸

```java
package com.academia.BooksJPA.dao;

import com.academia.BooksJPA.entity.Book;

public interface BookDAO {

    List<Book> findAll();

    List<Book> findByTitle(String theTitle);

    Book findById(Integer theId);

    List<Book> findByAuthor(String theAuthor);

    List<Book> findByGenre(String theGenre);

    List<Book> findByPages(Integer thePages);

    Book save(Book theBook);

    void deleteById(Integer theId);
}
```

BookController.java | Book.java | BookDAO.java | BookRepository.java | BookDAOImpl.java

BooksJPA ▸ src/main/java ▸ com.academia.BooksJPA.dao ▸ BookDAOImpl ▸

```java
package com.academia.BooksJPA.dao;

import com.academia.BooksJPA.entity.Book;

@Repository
public class BookDAOImpl implements BookDAO {

    private final EntityManager entityManager;

    @Autowired
    public BookDAOImpl(EntityManager entityManager) {
        this.entityManager = entityManager;
    }

    @Override
    public List<Book> findAll() {
        TypedQuery<Book> query = entityManager.createQuery("FROM Book", Book.class);
        return query.getResultList();
    }

    @Override
    public List<Book> findByTitle(String theTitle) {
        TypedQuery<Book> query = entityManager.createQuery("SELECT b FROM Book b WHERE b.title = :
        query.setParameter("title", theTitle);
        return query.getResultList();
    }

    @Override
    public Book findById(Integer theId) {
        return entityManager.find(Book.class, theId);
    }
```

# BOOKSDATAJPA

- MADE BASED ON BOOKSJPA IT USES THE SPRING DATA JPA TO TRADE THAT FLEXIBLITY FOR SIMPLICITY BASED ON THE METHODS IN THE INTERFACE

BookController.java | Book.java | BookDAO.java | BookRepository.java

BooksDataJPA ▸ src/main/java ▸ com.academia.BooksDataJPA.dao ▸ BookRepository ▸

```java
package com.academia.BooksDataJPA.dao;

import com.academia.BooksDataJPA.entity.Book;


public interface BookRepository extends JpaRepository<Book, Integer> {

    List<Book> findAll();

    List<Book> findByTitle(String theTitle);

    Book findById(Long theId);

    List<Book> findByAuthor(String theAuthor);

    List<Book> findByGenre(String theGenre);

    List<Book> findByPages(Integer thePages);
}

```

# BOOKS

- THE BOOKS ENTITY WHICH POINTS TO THE BOOKS TABLE, HAS AN ID, A TITLE, AN AUTOR, A GENRE AND A NUMBER OF PAGES. ALSO THE CONSTRUCTORS, GETTERS AND SETTERS AND THE TO STRING METHOD.

```
4
5  @Entity
6  @Table(name = "books")
7  public class Book {
8
9      @Id
10     @GeneratedValue(strategy = GenerationType.IDENTITY)
11     @Column(name = "id")
12     private Integer id;
13
14     @Column(name = "title", nullable = false)
15     private String title;
16
17     @Column(name = "author", nullable = false)
18     private String author;
19
20     @Column(name = "genre")
21     private String genre;
22
23     @Column(name = "pages")
24     private Integer pages;
25
```

# POSTMAN TESTS

THE SAME METHODS WILL WORK IN BOTH PROYECTS AS THEY USE THE SAME CONTROLLER.

- GET: HTTP://LOCALHOST:8080/LIBRARY/BOOKS

  HTTP://LOCALHOST:8080/LIBRARY/BOOKS/{BOOKID}

  HTTP://LOCALHOST:8080/LIBRARY/BOOKS/AUTHOR/{AUTHOR}

# POST METHOD

POST:HTTP: //LOCALHOST:8080/LIBRARY/BOOKS

- JSON: {"TITLE": “POST TEST", "AUTHOR": "EVER", "GENRE": "GENRE", "PAGES": 123 }

PUT: HTTP://LOCALHOST:8080/LIBRARY/BOOKS/{BOOKID}

- JSON: {"TITLE": “PUT  TEST", "AUTHOR": "EVER", "GENRE": "GENRE", "PAGES": 123 }

PUT: HTTP://LOCALHOST:8080/LIBRARY/BOOKS/{BOOKID}

- JSON: {"TITLE": “PUT  TEST", "AUTHOR": "EVER", "GENRE": "GENRE", "PAGES": 123 }

DELETE: HTTP://LOCALHOST:8080/LIBRARY/BOOKS/{BOOKID}