TRƯỜNG ĐẠI HỌC VĂN LANG
KHOA CÔNG NGHỆ THÔNG TIN

LẬP TRÌNH JAVA NÂNG CAO

CHƯƠNG 6: JPA & HIBERNATE

Giảng Viên Giảng Dạy:
ThS. Nguyễn Minh Tân
ThS. Đặng Đình Hòa
ThS. Trần Công Thanh

HỌC KỲ III – NĂM HỌC 2023-2024

KHÓA 27T-IT

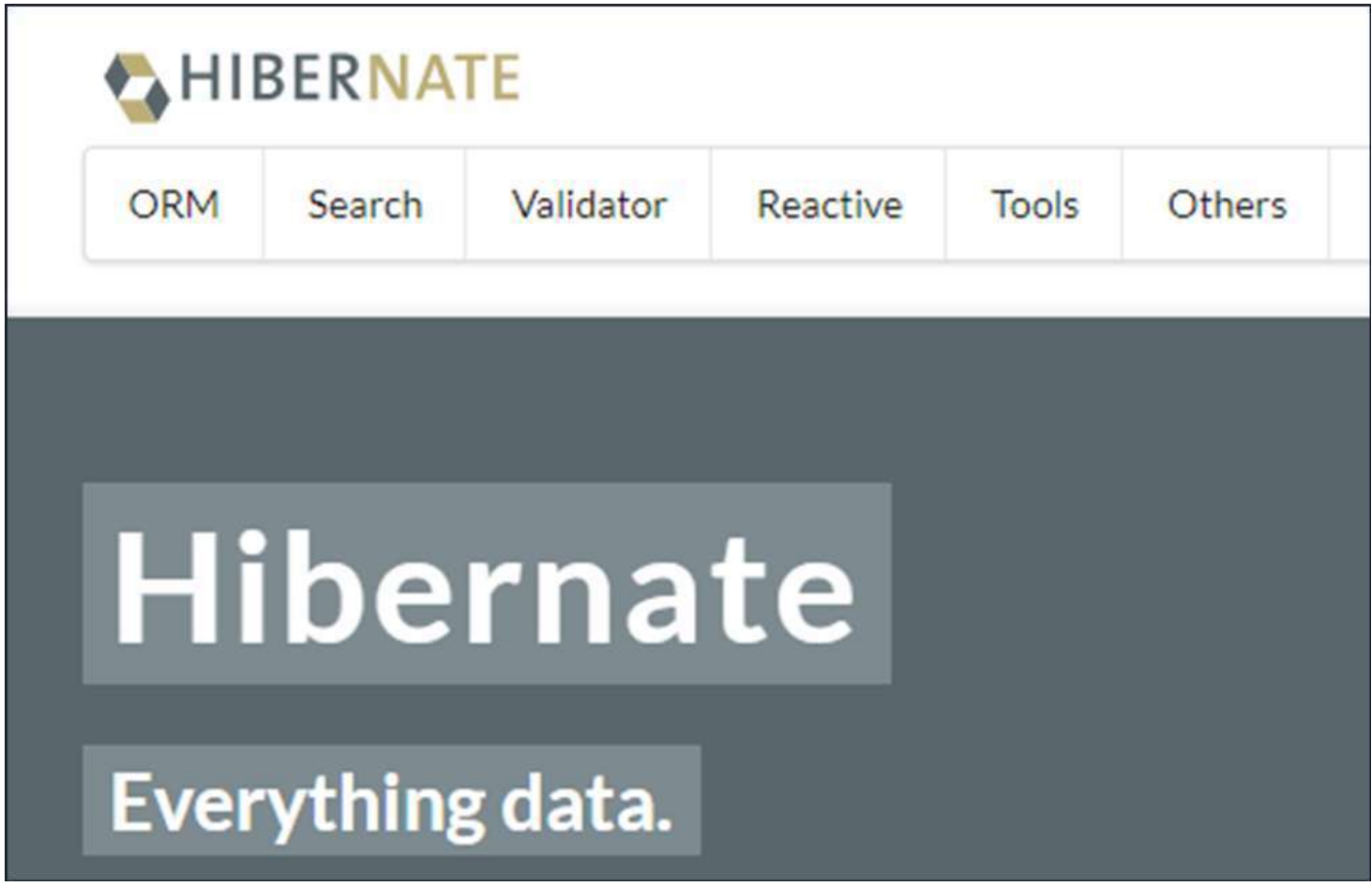
Nội dung

1. Hibernate
2. JPA
3. Một số Annotations quan trọng
4. Ví dụ minh họa
5. Q & A

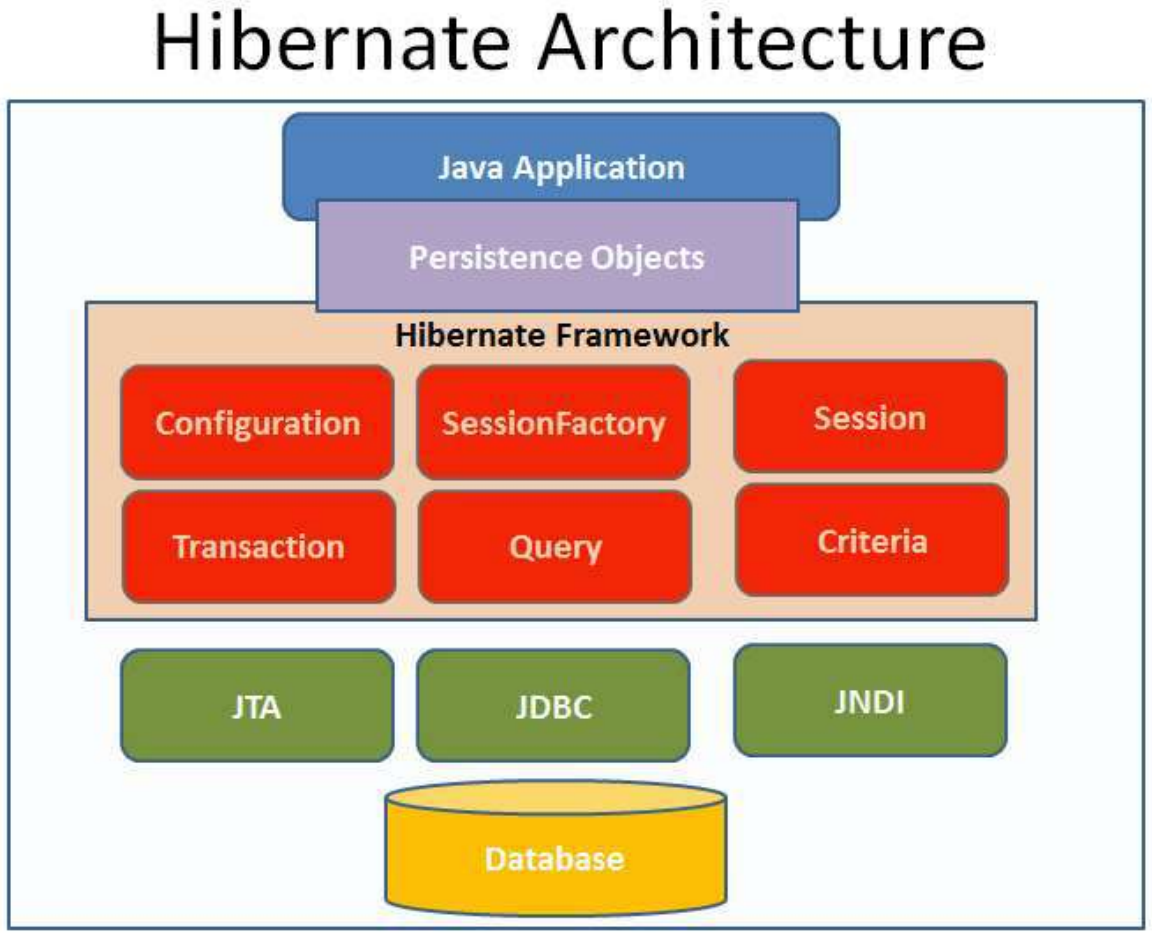
Hibernate là gì ?

https://hibernate.org/

- Hibernate là một framework phát triển ứng dụng trong lĩnh vực lập trình Java.
- Được sử dụng để giải quyết vấn đề liên quan đến lưu trữ và truy xuất dữ liệu từ cơ sở dữ liệu.



Hibernate Architecture



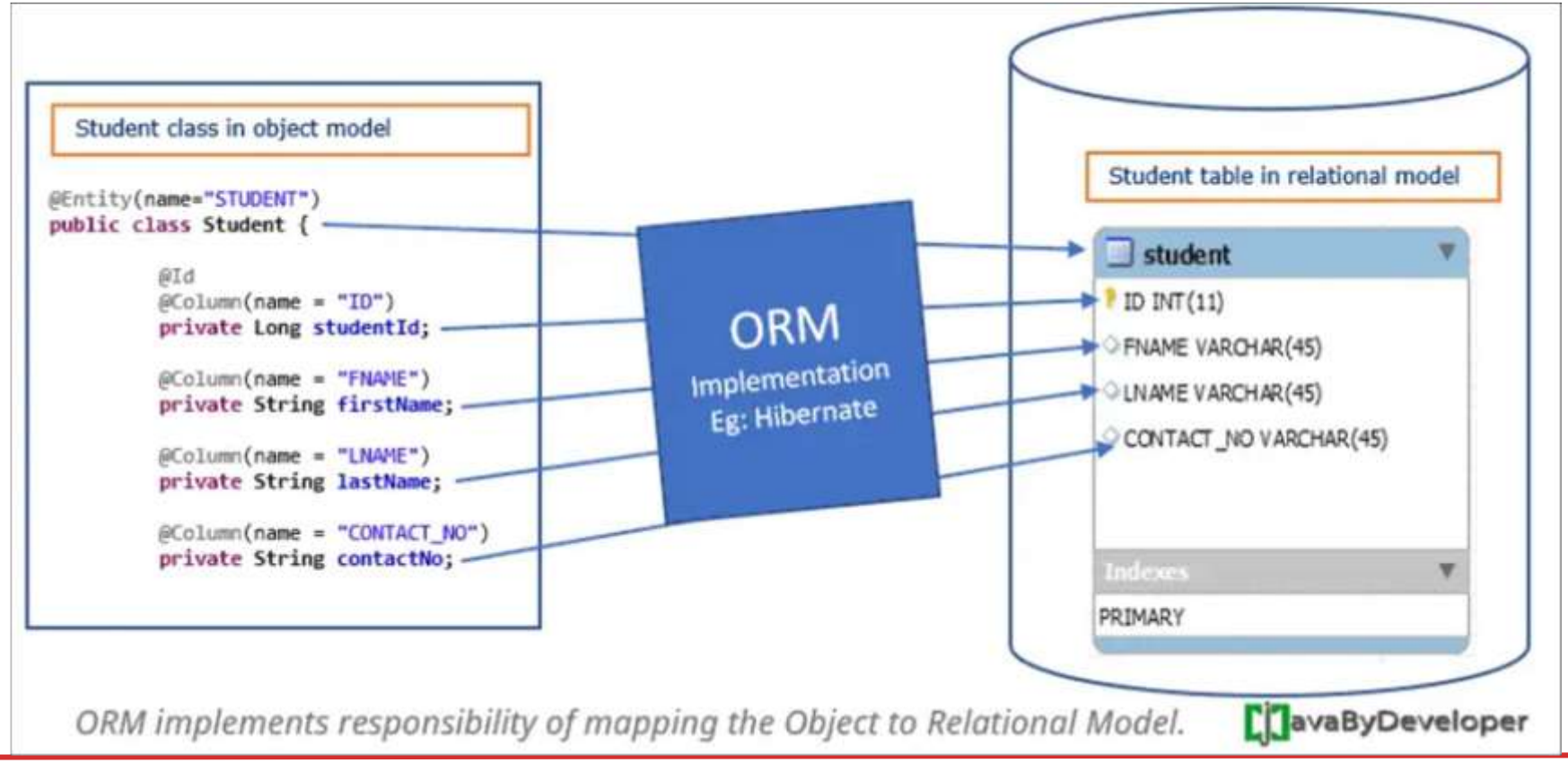
https://topdev.vn/blog/hibernate-la-gi-sao-phai-dung-no-thay-jdbc/

Vì sao nên dùng Hibernate

- Ánh xạ đối tượng quan hệ - Object Relational Mapping (ORM)
- Quản lý mối quan hệ - Relationship Management
- Tối ưu hóa truy vấn - Query Optimization

Object Relational Mapping (ORM)

https://javabydeveloper.com/orm-object-relational-mapping/



Một số annotations quan trọng

Entity Class - @Entity

- Được đánh dấu bằng @Entity
- Bắt buộc có một constructor rỗng (không có tham số) (nếu Class không có bất kỳ constructor nào, thì sẽ có một constructor rỗng mặc định)
- Class có thể có nhiều constructor

```
@Entity
public class Student {

    // fields, getters and setters

}
```

```
@Entity(name="student")
public class Student {

    // fields, getters and setters

}
```

Một số annotations quan trọng

@Table(name="xxxx")

- Được đánh dấu bên dưới các @Entity
- Quy định tên của Table trong cơ sở dữ liệu

```
@Entity
@Table(name = "ARTICLES")
public class Article {
    // ...
}
```

Một số annotations quan trọng

@Id
- Đánh dấu khóa chính

```java
@Entity
public class SinhVien {
    @Id
    private Long id;

    private String ten;
    private int tuoi;

    // Constructors, getters, setters...
}
```

```java
@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private Long id;
```

```java
@Id
@GeneratedValue(strategy = GenerationType.AUTO)
private Long id;
```

Một số annotations quan trọng

@Column

- Chỉ định tên cột, định dạng dữ liệu, độ dài, ràng buộc và các thuộc tính khác

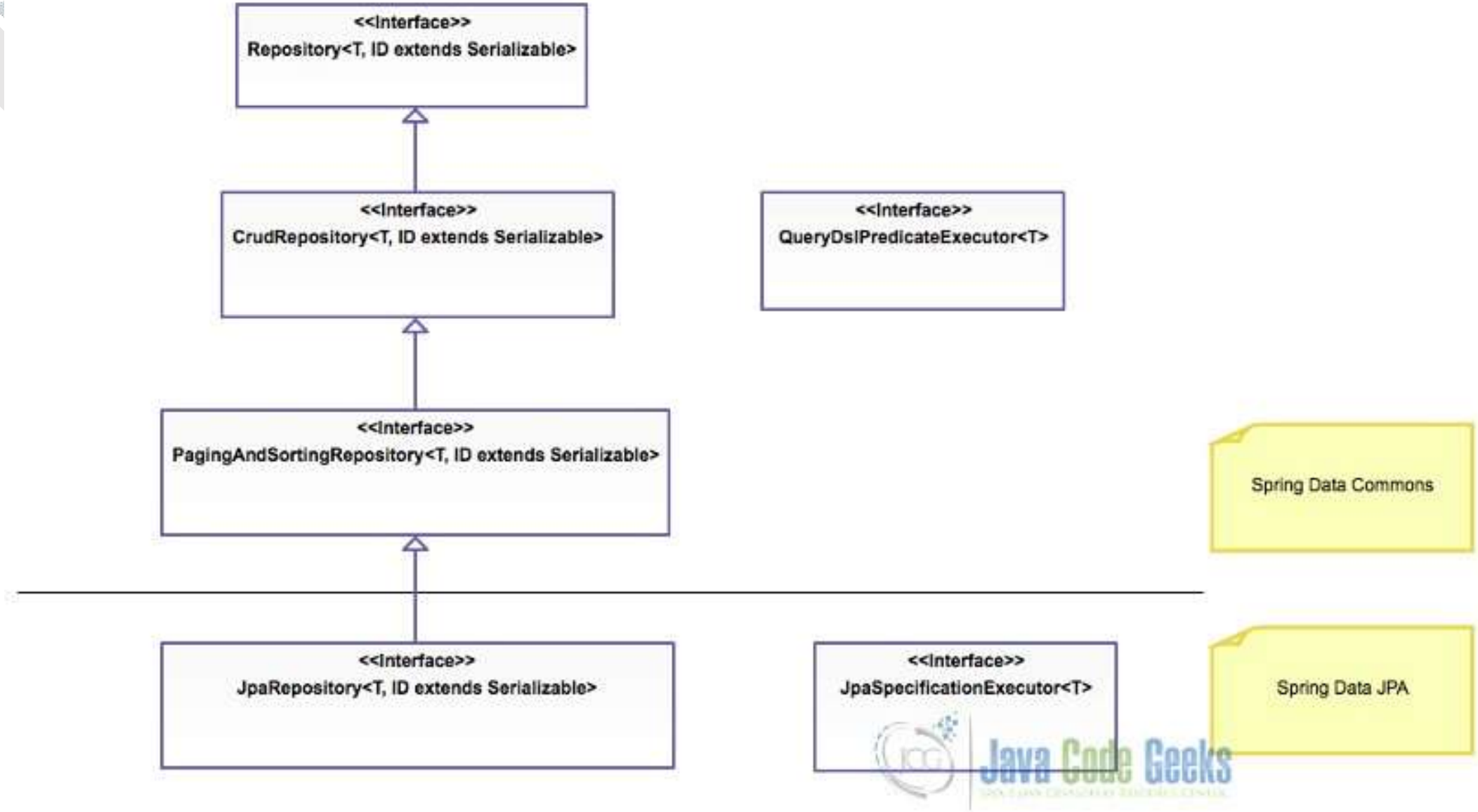
```java
@Entity
public class SinhVien {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(name = "ten_sv")
    private String ten;

    @Column(name = "tuoi_sv", nullable = false)
    private int tuoi;

    // Constructors, getters, setters...
}
```

```java
@Column(name = "ten_sv", length = 50)
private String ten;
```

Spring Data JPA là gì ?

- Spring Data JPA là một công nghệ trong Spring Boot cung cấp cho chúng ta các tính năng để dễ dàng tương tác với cơ sở dữ liệu. Nó là một phần mở rộng của JPA (Java Persistence API) và sử dụng Hibernate để thực hiện các thao tác trên cơ sở dữ liệu.

Cấu trúc Spring Data JPA



Lý do sử dụng Spring Data JPA là gì ?

- Với Spring Data JPA, chúng ta không cần phải viết các truy vấn SQL trực tiếp mà có thể sử dụng các phương thức được tạo tự động từ các Repository Interface để thực hiện các truy vấn đó. Ngoài ra, Spring Data JPA cũng cung cấp cho chúng ta các tính năng như Paging and Sorting, Query By Example, Specification, Native Queries, ... giúp tăng tính linh hoạt và hiệu suất trong ứng dụng của chúng ta.
- Với Spring Boot, việc tích hợp Spring Data JPA trở nên rất dễ dàng bằng cách chỉ định các phụ thuộc trong file pom.xml và sử dụng các annotations để đánh dấu các Entity và Repository Interface. Sau đó, chúng ta có thể sử dụng các phương thức được tạo tự động từ Repository Interface để thực hiện các truy vấn dữ liệu một cách dễ dàng và tiện lợi.

Ưu điểm Spring Data JPA

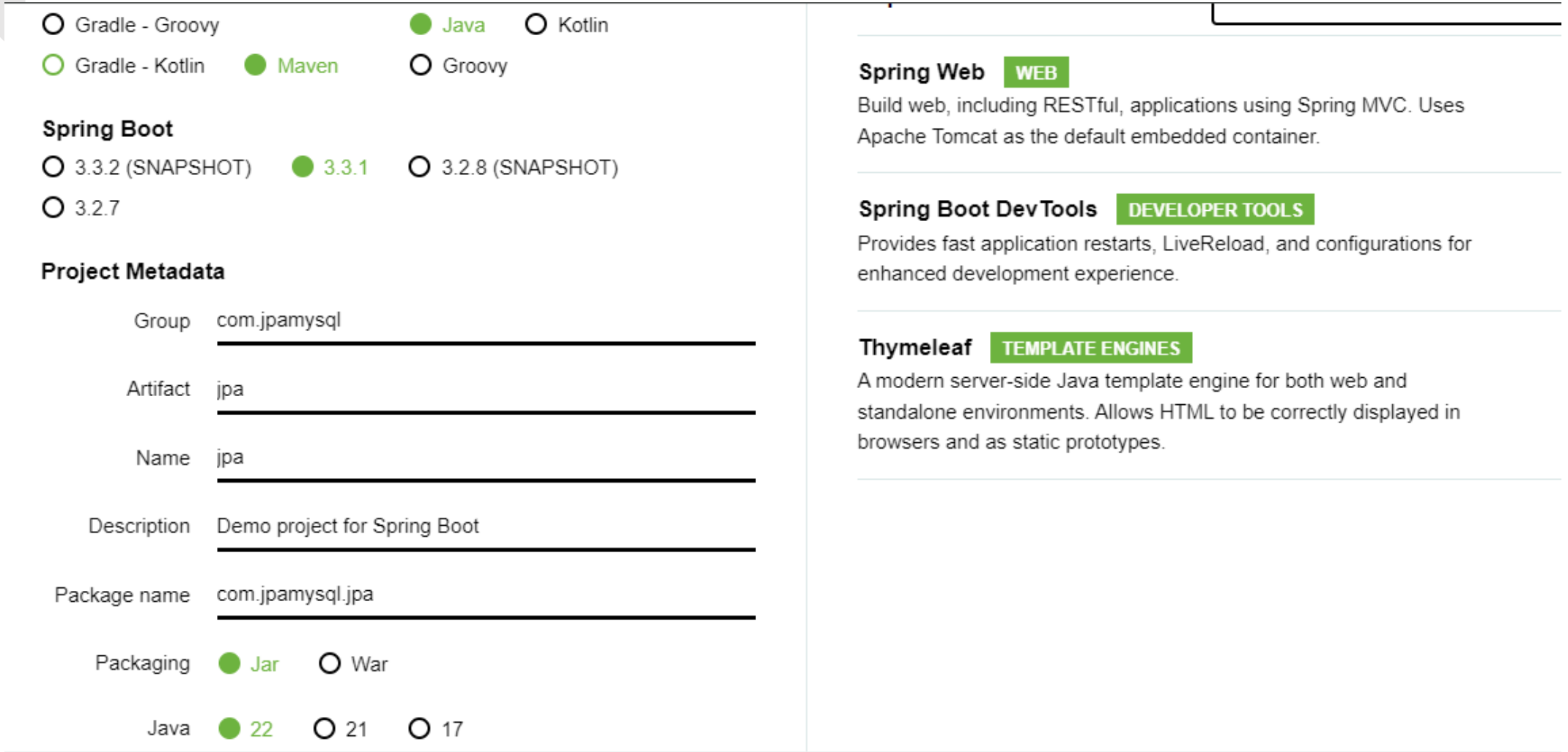
- Tính linh hoạt cao
- Giảm thiểu sự lặp lại của mã
- Tích hợp tốt với Spring Boot
- Tính tương thích cao
- Được hỗ trợ bởi cộng đồng lớn
- Tính bảo mật cao

Nhược điểm Spring Data JPA

- Khó khăn khi truy vấn nhiều bảng:
- Yêu cầu kiến thức về JPA:
- Hiệu suất có thể bị giảm:
- Khó khăn trong việc xử lý các truy vấn phức tạp:
- Thời gian học và hiểu quả phụ thuộc vào kinh nghiệm:

Cài đặt & Cấu hình Spring Data JPA

- Tạo dự án Spring boot tại spring.io



Mysql Database

- Trong phần này chúng ta sử dụng MySQL Database
- Chúng cần cần chuẩn bị
  - MySQL Database Server (có thể dung XAMPP hoặc cài trực tiếp)
  - Các công cụ để kết nối và tương tác với Database (HeidiSQL, MySQL Workbench, Dbeaver, …..)

Ôn tập một số câu lệnh truy vấn

- Lấy danh sách sinh viên có họ đệm bắt đầu bằng chữ "Nguyễn
- Lấy số lượng sinh viên có email kết thúc bằng "@gmail.com".
- Lấy danh sách sinh viên được sắp xếp theo tên và họ đệm theo thứ tự giảm dần
- Cập nhật email từ vani@gmail.com thành ivan@gmail.com
- Xóa sinh viên có id lớn nhất.
  - DELETE FROM sinhvien
  - WHERE id = (SELECT MAX(id) FROM sinhvien);

Cài đặt & Cấu hình Spring Data JPA

- **Bước 1**: Thêm dependency vào file pom.xml:
- Đầu tiên bạn muốn sử dụng được Spring Data JPA thì bạn cần phải thêm dependency của Spring Data JPA vào file pom.xml

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
```

Cài đặt & Cấu hình Spring Data JPA

Bước 1: Thêm dependency vào file pom.xml:

Đầu tiên bạn muốn sử dụng được Spring Data JPA thì bạn cần phải thêm dependency của Spring Data JPA vào file pom.xml

Dependency spring-boot-starter-data-jpa cung cấp một số dependency liên quan đến JPA, bao gồm Hibernate, JDBC và các dependency khác.

Lưu ý rằng bạn cũng cần phải thêm các dependency liên quan đến cơ sở dữ liệu mà bạn đang sử dụng (ví ss như MySQL connector).

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
```

Cài đặt & Cấu hình Spring Data JPA

Bước 2: Cấu hình kết nối cơ sở dữ liệu trong file application.properties.

Spring Boot cung cấp một cách để cấu hình database thông qua file application.properties

Bạn có thể cấu hình database bằng cách thêm các thuộc tính trong file này. Ví dụ:

```
spring.datasource.url=jdbc:mysql://localhost:3306/db_name
spring.datasource.username=db_username
spring.datasource.password=db_password
#spring boot tự động phát hiện ra driver-class
#spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
```

Cài đặt & Cấu hình Spring Data JPA

- Bước 3: Tạo các Entity để ánh xạ đến các bảng trong cơ sở dữ liệu.
- Một entity class là một lớp Java đại diện cho một bảng trong database. Trong entity class, các trường của lớp tương ứng với các cột trong bảng. Bạn có thể sử dụng annotation để chỉ định các thuộc tính của entity class.
- Ví dụ, để tạo một entity class để ánh xạ với bảng users trong database, bạn có thể sử dụng đoạn mã sau:

Cài đặt & Cấu hình Spring Data JPA

```java
@Entity
@Table(name = "employees")
public class Employee {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(name = "name")
    private String name;

    @Column(name = "age")
    private int age;

    // Getters and setters
}
```

Cài đặt & Cấu hình Spring Data JPA

- Bước 4: Tạo Repository interface để thao tác với database.
- Repository interface là một interface trong Spring Data JPA được sử dụng để thao tác với database. Spring Data JPA sẽ tạo ra các method để thao tác với database dựa trên tên method và kiểu dữ liệu của tham số truyền vào.
- Ví dụ, để tạo một repository để thao tác với entity class User, bạn có thể sử dụng đoạn mã sau:

Cài đặt & Cấu hình Spring Data JPA

```java
@Repository
public interface UserRepository extends JpaRepository<User, Long> {

    List<User> findByLastName(String lastName);

    User findByEmail(String email);
}
```

Cài đặt & Cấu hình Spring Data JPA

- Bước 5: Sử dụng Repository để thao tác với database.
- Sau khi đã tạo Repository interface, bạn có thể sử dụng nó để thao tác với database trong ứng dụng. Ví dụ, để tạo một user mới, bạn có thể sử dụng đoạn mã sau:

```
@Service
public class UserService {

    @Autowired
    private UserRepository userRepository;

    public User createUser(User user) {
        return userRepository.save(user);
    }
}
```

Cài đặt & Cấu hình Spring Data JPA

Bước 6: Sử dụng các phương thức được cung cấp bởi Spring Data JPA để thực hiện các thao tác CRUD với cơ sở dữ liệu:

Tham khảo ví dụ:

Cài đặt & Cấu hình Spring Data JPA

```java
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class UserService {

    @Autowired
    private UserRepository userRepository;

    public List<User> getAllUsers() {
        return userRepository.findAll();
    }

    public User createUser(User newUser) {
        return userRepository.save(newUser);
    }

    public User updateUser(User userToUpdate) {
        return userRepository.save(userToUpdate);
    }

    public void deleteUser(Long userId) {
        userRepository.deleteById(userId);
    }
}
```

Các tính năng nâng cao của Spring Data JPA

Spring Data JPA cung cấp các tính năng nâng cao như lọc dữ liệu, phân trang, đặt tên các truy vấn tùy chỉnh và sử dụng các annotation để định nghĩa các mối quan hệ giữa các đối tượng.

TRƯỜNG ĐẠI HỌC
VĂN LANG

KHOA CÔNG NGHỆ THÔNG TIN