

MYSQL & JPA

Nguyễn Huy Cường nh.cuong@hutech.edu.vn

04/2023

Nội dung

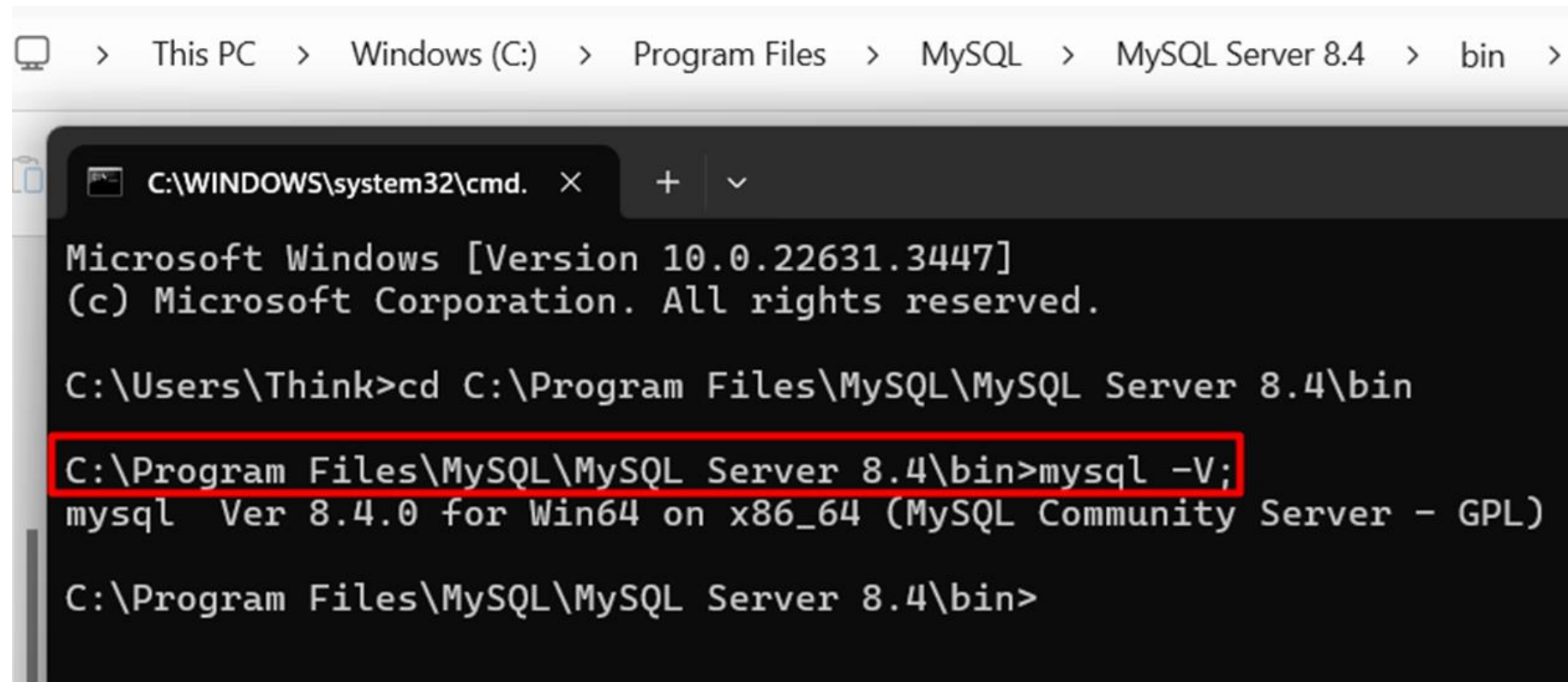
1. Cài đặt MySQL Server và GUI Tools
2. JPA Phần 1

1. Cài đặt MySQL Server

- Cài đặt bản **MySQL Community**:

<https://dev.mysql.com/downloads/mysql/>

- Kiểm tra phiên bản đã cài đặt
`mysql -V;`



```

> This PC > Windows (C:) > Program Files > MySQL > MySQL Server 8.4 > bin >

C:\WINDOWS\system32\cmd. x + v

Microsoft Windows [Version 10.0.22631.3447]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Think>cd C:\Program Files\MySQL\MySQL Server 8.4\bin

C:\Program Files\MySQL\MySQL Server 8.4\bin>mysql -V;
mysql  Ver 8.4.0 for Win64 on x86_64 (MySQL Community Server - GPL)

C:\Program Files\MySQL\MySQL Server 8.4\bin>
```

2. Cài đặt MySQL GUI Tools

Có thể sử dụng 1 trong các tools sau để quản lý:

- PHPMyadmin

Giao diện web: <http://localhost/phpmyadmin>

- HeidiSQL

<https://www.heidisql.com/download.php>

- MySQL Workbench

<https://www.mysql.com/products/workbench/>

Ngoài ra còn nhiều công cụ khác:

dbForge Studio for MySQL, Toad Edge for MySQL, SQLyog, Navicat for MySQL, Aqua Data Studio, Valentina Studio, Sequel Pro...



JAVA PERSISTENCE API (JPA)

Spring Boot làm việc với CSDL như thế nào ?

- Spring JDBC:

Phải quản lý kết nối, đóng kết nối cơ sở dữ liệu
Định nghĩa các truy vấn SQL

- **Spring data JPA:** Sử dụng Hibernate để cung cấp ORM

❑ Hibernate: Sử dụng HQL (Hibernate Query Language) để thực hiện các truy vấn, sau đó ánh xạ kết quả tới các đối tượng java (ORM).

- **MyBatis**

Dependencies

ADD DEPENDENCIES... CTRL + B

MyBatis Framework SQL

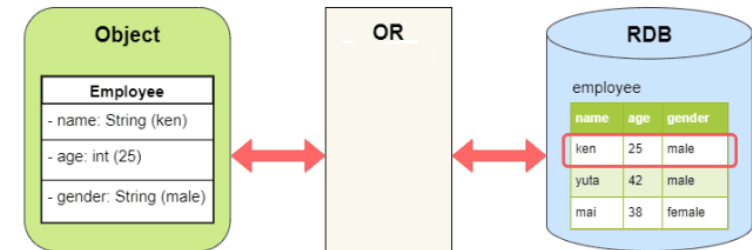
Persistence framework with support for custom SQL, stored procedures and advanced mappings. MyBatis couples objects with stored procedures or SQL statements using a XML descriptor or annotations.

Dependencies

ADD DEPENDENCIES... CTRL + B

Spring Data JDBC SQL

Persist data in SQL stores with plain JDBC using Spring Data.



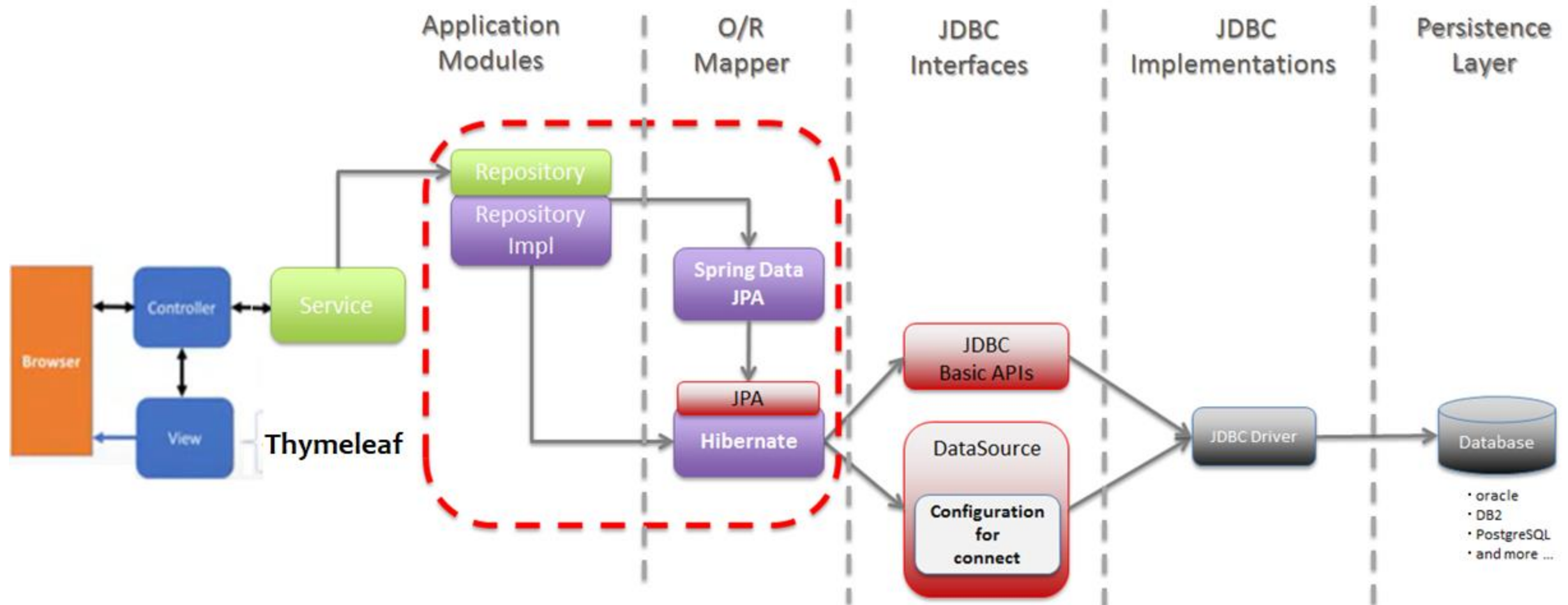
Dependencies

ADD DEPENDENCIES... CTRL + B

Spring Data JPA SQL

Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

Spring Data JPA Tổng quan



Spring Data JPA

1 – Kết nối CSDL: MySQL, cấu hình JPA

Cấu hình `application.properties`

● Kết nối với mysql

```
spring.datasource.url=jdbc:mysql://localhost:3306/{your_db}  
spring.datasource.username={your_user}  
spring.datasource.password={your_pass}
```

● Khởi tạo DDL sử dụng JPA

```
#none, create-only, create, create-drop, validate, update  
spring.jpa.hibernate.ddl-auto={update}
```

Dependencies

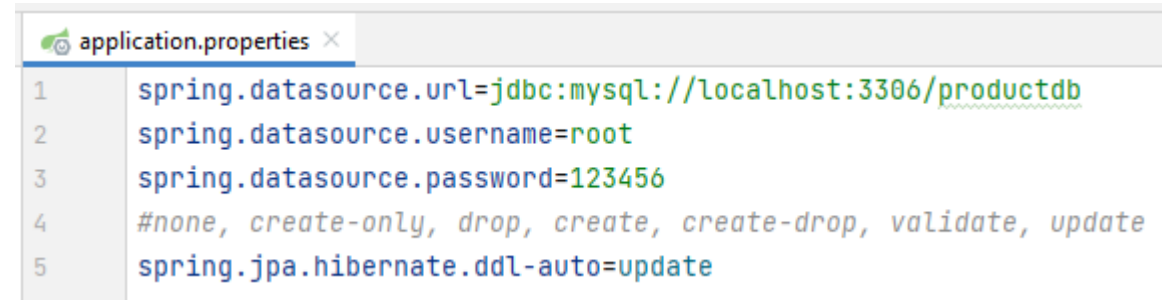
[ADD DEPENDENCIES... CTRL + B](#)

MySQL Driver SQL

MySQL JDBC driver.

Spring Data JPA SQL

Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.



The screenshot shows a code editor window titled 'application.properties'. It contains five lines of configuration code for a MySQL database connection and JPA settings. Line 1 sets the URL to 'jdbc:mysql://localhost:3306/productdb'. Line 2 sets the username to 'root'. Line 3 sets the password to '123456'. Line 4 is a comment showing various options for 'spring.jpa.hibernate.ddl-auto'. Line 5 sets 'spring.jpa.hibernate.ddl-auto' to 'update'.

```
1 spring.datasource.url=jdbc:mysql://localhost:3306/productdb  
2 spring.datasource.username=root  
3 spring.datasource.password=123456  
4 #none, create-only, drop, create, create-drop, validate, update  
5 spring.jpa.hibernate.ddl-auto=update
```


cấu hình JPA: Kiểm soát việc tạo cơ sở dữ liệu với Hibernate

`spring.jpa.hibernate.ddl-auto={ddl_type}`

Các thuộc tính DDL của Hibernate gồm có: ***none, create, update, create-drop, validate***

- ❑ ***none***: tắt việc tạo DDL (mặc định).
- ❑ ***create***: drop các bảng đang có và sau đó tạo ra các bảng mới.
- ❑ ***create-drop***: Thường được sử dụng cho unit test, tương tự như create nhưng sau khi tất cả các hoạt động đã được hoàn thành sẽ thực hiện việc drop database.
- ❑ ***update***: Dựa trên việc ánh xạ (thông qua các annotation/ XML) cập nhật lại schema với những thông tin mới. Không xóa đi các bảng hay các cột đang có.
- ❑ ***validate***: **xác thực** xem các bảng và các cột có tồn tại hay không, nếu không nó sẽ ném ra một ngoại lệ (exception).

Spring Data JPA

2- Định nghĩa JPA Entities

- **Entity:** sử dụng **@Entity** đại diện cho **table** (tên mặc định là tên class) trong CSDL
nếu cần thay đổi sử dụng thuộc tính **name** trong **@Table**
- ❑ Khóa chính: **@Id** và có 4 loại AUTO/ SEQUENCE/ TABLE / IDENTITY
@GeneratedValue(strategy=GenerationType.AUTO)
- ❑ Các cột Column: **@Column**
có nhiều thuộc tính: *name, length, nullable, unique*
- ❑ **@Transient:** thuộc tính/ phương thức này không liên quan gì tới một cột nào ở database

```

@Entity
@Table(name="product")
public class Product {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;
    @Column
    private String name;
    @Column
    private String image;
    @Column
    private long price;
}

```



Host: 127.0.0.1Database: productdbTable: productDataQuery

Columns: Add Remove Up Down

#	Name	Datatype	Length/Set	Unsign...	Allow N...	Zerofill	Default
1	id	INT		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT
2	image	VARCHAR	255	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL
3	name	VARCHAR	255	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL
4	price	BIGINT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL

3- Sử dụng JpaRepository

• Các phương thức

- Query DSL

- CRUD operations

- Paging and sorting

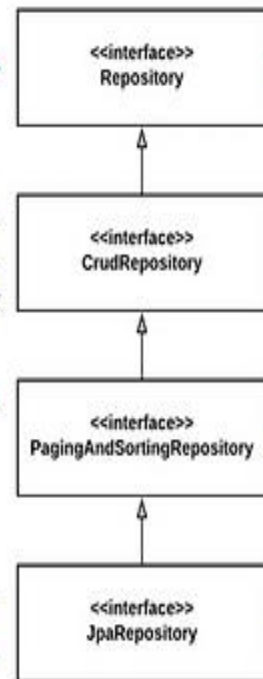
- Helpers

- count()

- existsById(ID)

- flush()

- deleteInBatch(Iterable)

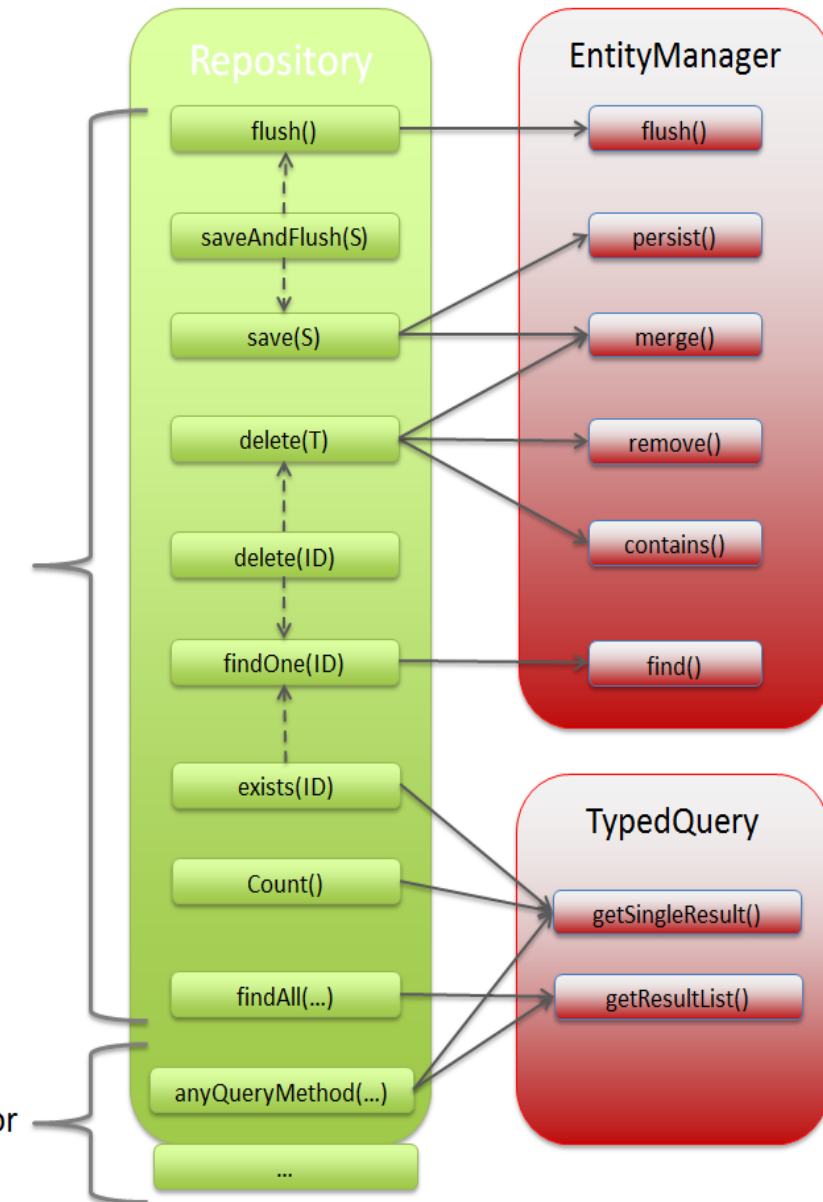


Spring Data

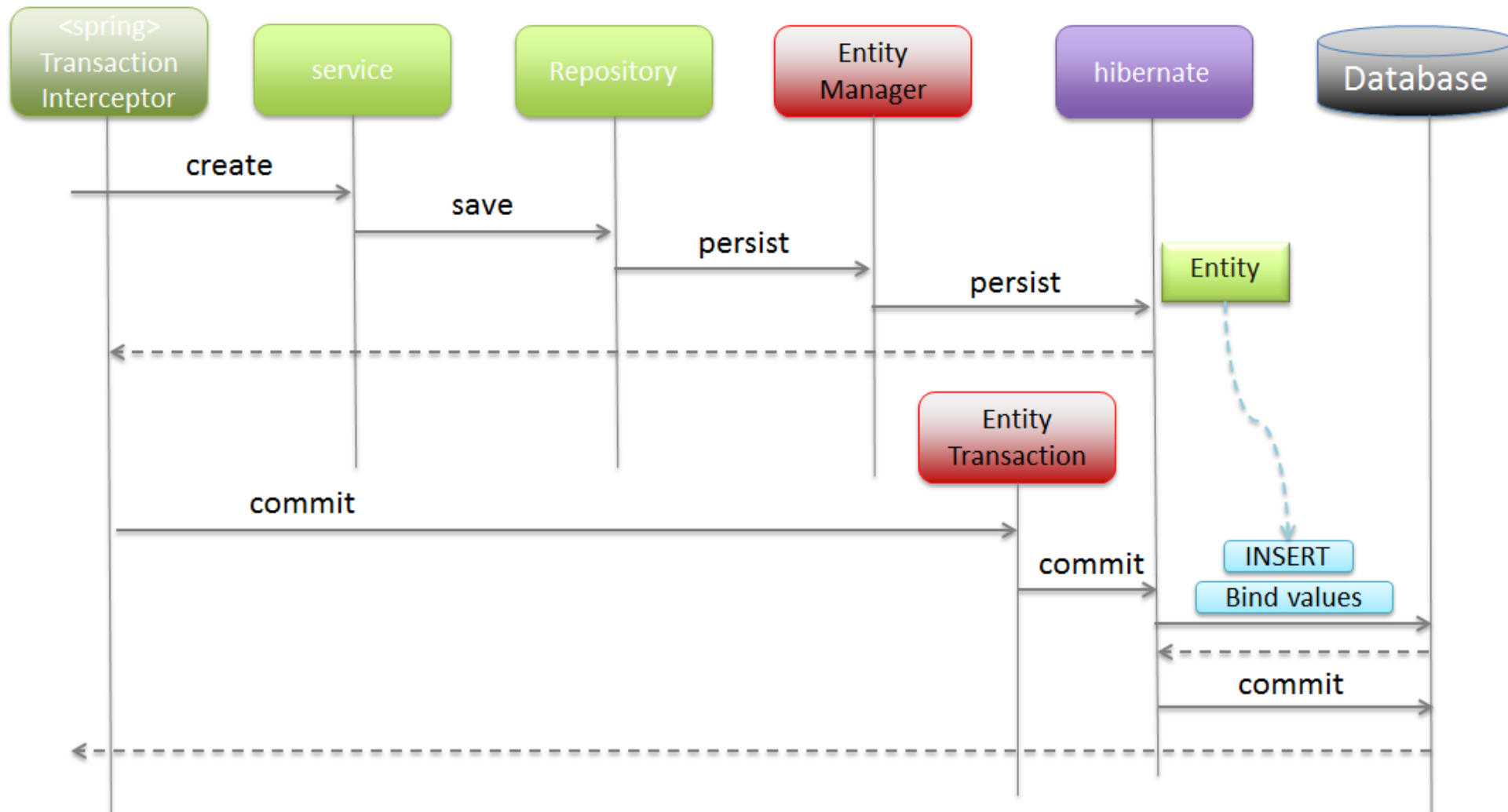
Spring Data JPA

Methods of JpaRepository Interface.

Query Methods in Repository Interface for Entity that defined by developer.



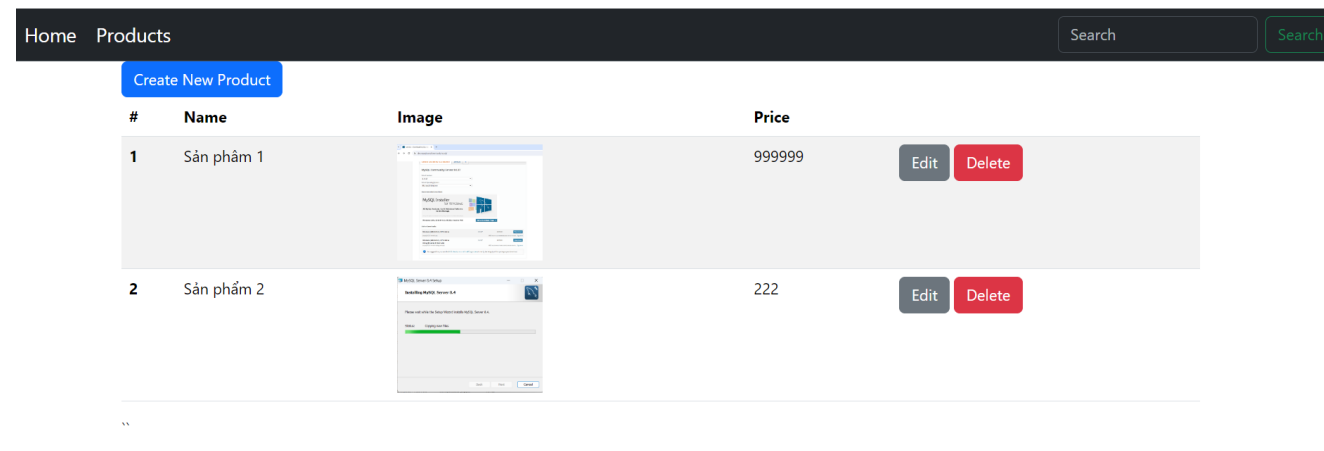
Entity manager vs Entity Transaction





Thymeleaf + JPA + MySQL

- (0) Cài đặt môi trường, dependency
- (1) Cấu hình kết nối CSDL
- (2) Tạo JPA entities
- (3) Tạo JPA repositories
- (4) Tạo Service - Tạo controller - Thymeleaf templates



Thymeleaf + JPA + MySQL

Bo: dependency cho JPA, và CSDL

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<dependency>
  <groupId>com.mysql</groupId>
  <artifactId>mysql-connector-j</artifactId>
  <scope>runtime</scope>
</dependency>
```

Dependencies

ADD DEPENDENCIES... CTRL + B

Spring Data JPA

SQL

Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

MySQL Driver

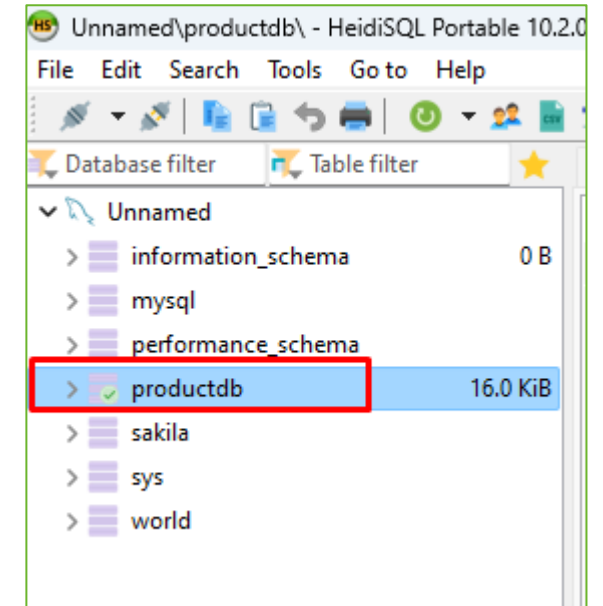
SQL

MySQL JDBC driver.

Thymeleaf + JPA + MySQL

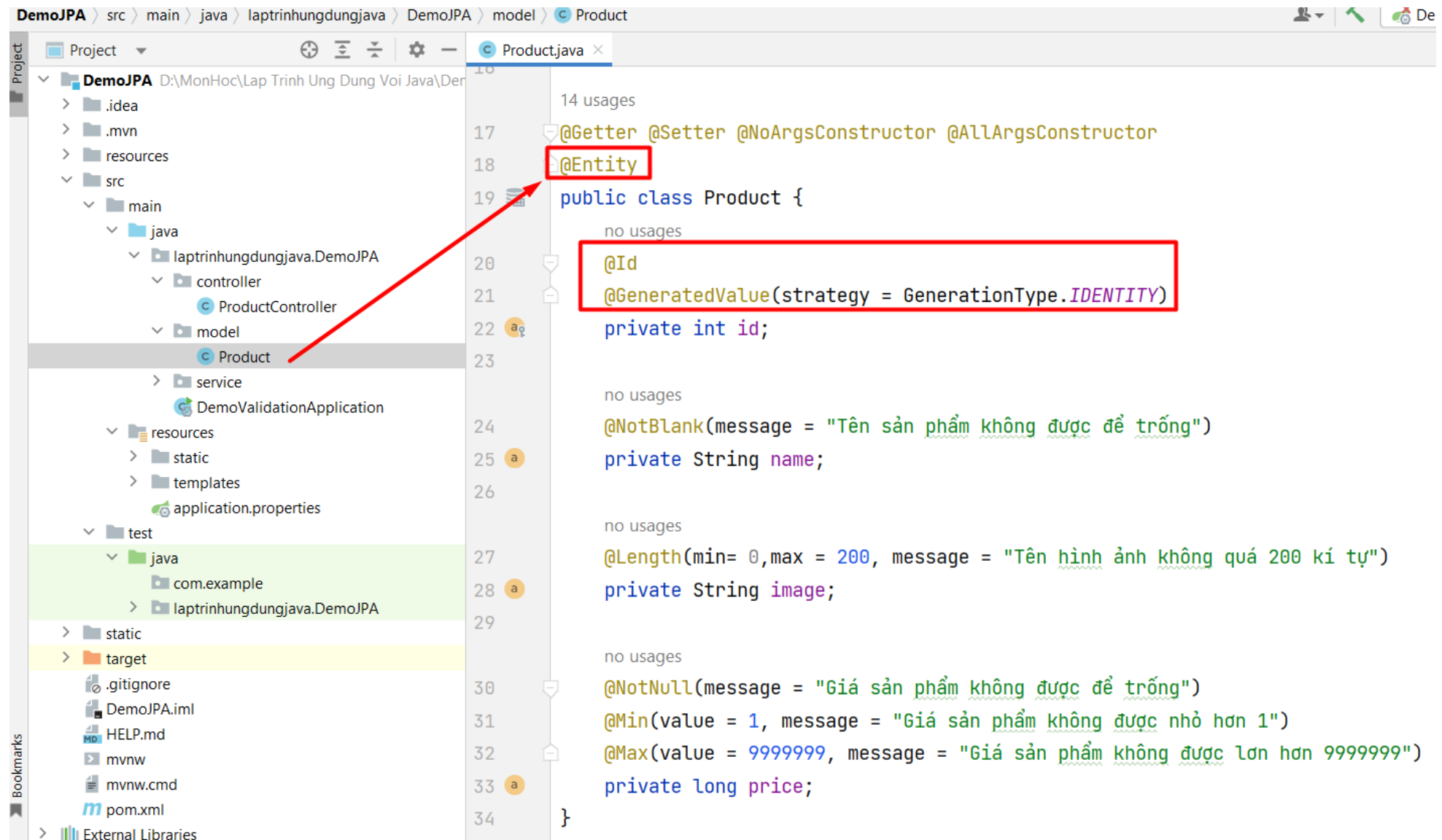
B1: Cấu hình kết nối CSDL

```
spring.datasource.url=jdbc:mysql://localhost:3306/productdb  
spring.datasource.username=root  
spring.datasource.password=123456  
#none, create-only, drop, create, create-drop, validate, update  
spring.jpa.hibernate.ddl-auto=update
```



Thymeleaf + JPA + MySQL

B2: Tạo JPA entities



The screenshot shows an IDE window with the following content:

Project Structure (Left Sidebar):

- Project: DemoJPA
- src/main/java/laptrinhungdungjava/DemoJPA/model/Product

Code Editor (Product.java):

```
10 14 usages
17 @Getter @Setter @NoArgsConstructor @AllArgsConstructor
18 @Entity
19 public class Product {
20     no usages
21     @Id
22     @GeneratedValue(strategy = GenerationType.IDENTITY)
23     private int id;
24     no usages
25     @NotBlank(message = "Tên sản phẩm không được để trống")
26     private String name;
27     no usages
28     @Length(min= 0,max = 200, message = "Tên hình ảnh không quá 200 kí tự")
29     private String image;
30     no usages
31     @NotNull(message = "Giá sản phẩm không được để trống")
32     @Min(value = 1, message = "Giá sản phẩm không được nhỏ hơn 1")
33     @Max(value = 9999999, message = "Giá sản phẩm không được lớn hơn 9999999")
34     private long price;
35 }
```

A red box highlights the `@Entity` annotation on line 18 and the `@Id` and `@GeneratedValue(strategy = GenerationType.IDENTITY)` annotations on lines 21 and 22. A red arrow points from the `Product` entity in the project structure to the `@Entity` annotation.

Thymeleaf + JPA + MySQL

B3: JPA repositories

The screenshot shows an IDE interface with the Project Explorer on the left and the ProductRepository.java file open in the editor. The Project Explorer shows the project structure: DemoJPA > src > main > java > laptrinhungdungjava.DemoJPA > repository > ProductRepository. The ProductRepository.java file contains the following code:

```
1 package laptrinhungdungjava.DemoJPA.repository;  
2  
3 import laptrinhungdungjava.DemoJPA.model.Product;  
4 import org.springframework.data.jpa.repository.JpaRepository;  
5 import org.springframework.stereotype.Repository;  
6  
7 no usages  
8 @Repository  
9 public interface ProductRepository extends JpaRepository<Product, Integer> {  
10 }
```

A red box highlights the ProductRepository in the Project Explorer, and a red arrow points from it to the @Repository annotation in the code.

B4: Service – controller – thymeleaf

```
@Service
public class ProductService {

    4 usages
    @Autowired
    private ProductRepository productRepository;

    1 usage
    public List<Product> getAll() {
        return productRepository.findAll();
    }

    public Product get(int id) {
        return productRepository.findById(id).orElse( other: null);
    }

    public void add(Product newProduct) {
        productRepository.save(newProduct);
    }

    2 usages
    public void updateImage(Product newProduct, MultipartFile imageProduct)
    {...}

    1 usage
    public void update(Product editProduct)
    {
        Product find = get(editProduct.getId());
        if(find!= null) {
            find.setImage(editProduct.getImage());
            find.setPrice(editProduct.getPrice());
            find.setName(editProduct.getName());
            productRepository.saveAndFlush(find);
        }
    }
}
```

B4: controller - thymeleaf (layout) [bài 4 – validation]

```

@Controller
@RequestMapping("/products")
public class ProductController {

    6 usages
    @Autowired
    private ProductService productService;

    no usages
    @GetMapping("/create")
    public String Create(Model model) {
        model.addAttribute(attributeName: "product", new Product());
        return "product/create";
    }

    no usages
    @PostMapping("/create")
    public String Create(@Valid Product newProduct,
        BindingResult result,
        @RequestParam MultipartFile imageProduct,
        Model model) {
        if (result.hasErrors()) {
            model.addAttribute(attributeName: "product", newProduct);
            return "product/create";
        }
        productService.updateImage(newProduct, imageProduct);
        productService.add(newProduct);
        return "redirect:/products";
    }

    no usages
    @GetMapping()
    public String Index(Model model)
    {
        model.addAttribute(attributeName: "listproduct", productService.getAll());
        return "product/products";
    }
}

```

Project: DemoJPA D:\MonHoc\Lap Trinh Ung Dung Voi Java\Der

Layout.html

```

1 <!DOCTYPE html>
2 <html lang="en"
3     xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout">
4 <head>
5     <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
6     <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
7     <meta name="description" content="BigSchool - Home page with courses and search functionality">
8     <title>Demo Products</title>
9     <!-- Bootstrap CSS -->
10    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
11          rel="stylesheet"
12          integrity="sha384-QWTKZyjpPEjISv5WaRU90FeRpok6YctnYmDr5pNlyT2bRjXh0JMhY6hW+ALEWih"
13          crossorigin="anonymous">
14 </head>
15 <body>
16     <nav class="navbar bg-dark border-bottom border-body data-bs-theme="dark">
17         <div style="padding-left: 40px">
18             <a class="navbar-brand" th:href="@{/products}">Home</a>
19             <a class="navbar-brand" th:href="@{/products/create}">Products</a>
20         </div>
21         <form class="d-flex" role="search" th:action="@{/home/search}" th:method="get">
22             <input class="form-control me-2" type="search" placeholder="Search" aria-label="Search">
23             <button class="btn btn-outline-success" type="submit">Search</button>
24         </form>
25     </nav>
26
27     <div layout:fragment="content" class="container body-content">
28
29     </div>
30     <div class="card-footer text-center">
31         <hr/>
32         <p>© Nguyễn Huy Cường - Lập trình ứng dụng với Java</p>
33     </div>
34 </body>
35 </html>

```

ASM5: BigSchool - Quản lý khóa học

Viết chương trình quản lý (xem, thêm , xóa, sửa) khóa học

Database filter Table filter

HuyCuong Database: bigschool Table: course Data Query

bigschool.course: 2 rows total (exact) Next

#	id	lecture_name	category_name	place	start_date
1	1	NGUYEN HUY CUONG	Lap trinh tren thiet bi di dong	THU DUC	2024-11-20 12:30:00.000000
2	2	NGUYEN HUY CUONG	Lap trinh tren thiet bi di dong	BINH DUONG	2025-03-08 12:30:00.000000

Yêu cầu 1: Lấy ds các khóa học (/courses)

localhost:8080/courses

Home Add Course Search

#	Lecture Name	Place	Category Name	Start Date - Time	
1	NGUYEN HUY CUONG	THU DUC	Lap trinh tren thiet bi di dong	20/11/2024 12:30	Edit Delete
2	NGUYEN HUY CUONG	BINH DUONG	Lap trinh tren thiet bi di dong	08/03/2025 12:30	Edit Delete

ASM5: BigSchool - Quản lý khóa học

Yêu cầu 2: Add Course (Thêm khóa học)

- Kiểm tra các trường bắt buộc phải nhập

❑ **Start Date** Nhập theo định dạng

dd/MM/yyyy HH:mm

Ví dụ: 10/03/2025 07:30

- **Add Course:**

+ Lưu thông tin nhập liệu vào CSDL

+ Redirect về trang home

← → ↺ ⓘ localhost:8080/courses/create 🔍 ☆ {=} 📁 | Tất cả dấu tr.

Home **Add Course** Search Search

Thông tin khóa học

Name

Lecture Name

Tên giảng viên không được để trống

Place:

Place

Nơi học không để được để trống

Start Date:

dd/MM/yyyy HH:mm

Ngày bắt đầu không để trống

Category Name

Category Name

Danh mục không được để trống

Add Course

ASM5: BigSchool - Quản lý khóa học

Yêu cầu 3: Edit Course (Thay đổi khóa học)

- Khi Edit (Từ Home) điền lại thông tin theo mã khóa học
- Kiểm tra các trường bắt buộc phải nhập

☐ **Start Date** Nhập theo định dạng

dd/MM/yyyy HH:mm

Ví dụ: 10/03/2025 07:30

- **Save Course:**

+ Thay đổi thông tin nhập liệu vào CSDL

+ Redirect về trang home

localhost:8080/courses/edit/1

[Home](#) [Add Course](#)

Thông tin khóa học

Name

Place:

Start Date:

Category Name

© Nguyễn Huy Cường - Lập trình ứng dụng với Java

ASM5: BigSchool - Quản lý khóa học

Yêu cầu 4: Delete Course (Xóa khóa học)

- Hiện thị confirm message

- OK

+ Xóa thông tin khóa học

+ Redirect về trang Home

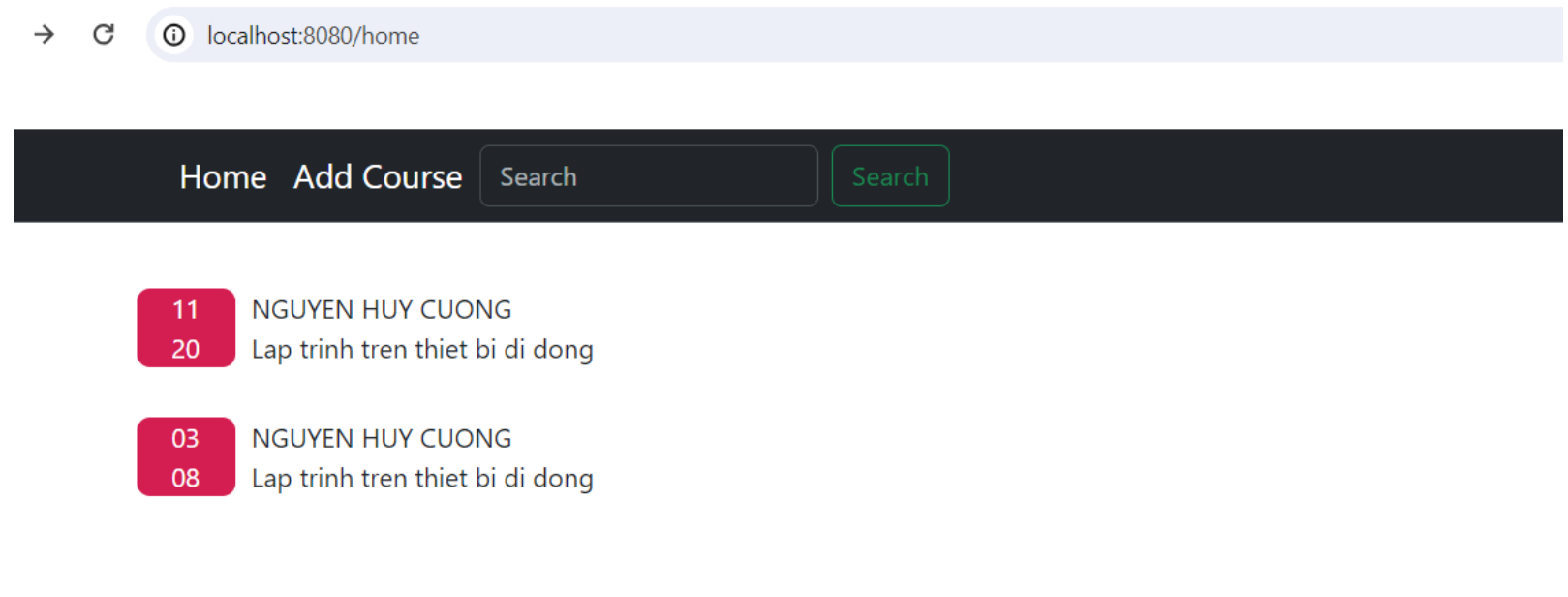
The screenshot shows a web browser at `localhost:8080/courses`. A confirmation dialog box is displayed in the center, asking "Are you sure to delete?" with "OK" and "Huỷ" (Cancel) buttons. Below the dialog is a table of courses. The table has columns: #, Lecture Name, Place, Category Name, Start Date - Time, Edit, and Delete. Two courses are listed. The "Delete" button for the second course is highlighted with a red box, and a red arrow points from it to the "OK" button in the dialog.

#	Lecture Name	Place	Category Name	Start Date - Time	Edit	Delete
1	NGUYEN HUY CUONG	THU DUC	Lap trinh tren thiet bi di dong	20/11/2024 12:30	Edit	Delete
2	NGUYEN HUY CUONG	BINH DUONG	Lap trinh tren thiet bi di dong	08/03/2025 12:30	Edit	Delete

ASM5: BigSchool - Quản lý khóa học

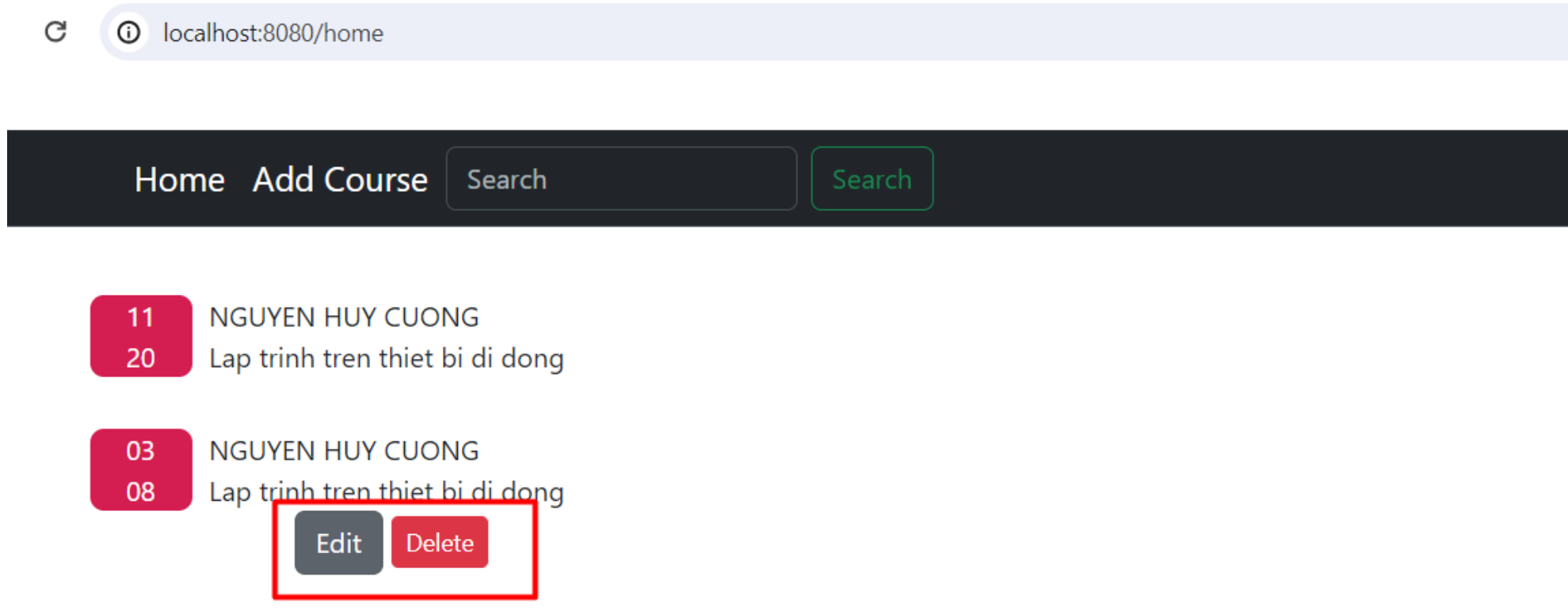
Yêu cầu 5: Home – Khóa học sắp tới

- Điều chỉnh Home: chỉ lấy các khóa học sắp tới (là khóa học có **thời gian bắt đầu > thời gian hiện tại**)
- Home có giao diện: Chứa thông tin Tháng , ngày học (sắp tới), LectureName và CategoryName



ASM5: BigSchool - Quản lý khóa học

Yêu cầu 6: **Home** – Hiển thị Edit, Delete khi hover chuột vào các item



Q&A

Thank you!