# Spring Boot

## Spring Boot

### Spring Boot là gì?

Spring Boot là một framework mạnh mẽ được xây dựng dựa trên nền của Spring Framework mà đơn giản hóa sự phát triển các ứng dụng Spring. Mục tiêu của Spring boot là **cắt giảm boilerplate code, cấu hình, thời gian thiết lập**, cho phép lập trình viên tập trung vào xây dựng chức năng chinh của ứng dụng của họ.



### Các tính năng nổi bật của Spring Boot

#### SpringApplication

Lớp **SpringApplication** là điểm khởi đầu chính cho các ứng dụng Spring Boot. Lớp này cung cấp một cách thuận tiện để bootstrap và khởi chạy một ứng dụng Spring từ phương thức main().

Khi chạy một ứng dụng Spring Boot, SpringApplication sẽ tự động thực hiện các tác vụ như tạo **ApplicationContext**, đăng ký các **CommandLineRunner** và **ApplicationRunner** (nếu có), và khởi động các **máy chủ nhúng** (như Tomcat). Điều này giúp việc khởi động ứng dụng trở nên đơn giản chỉ bằng **một dòng lệnh**.

#### Externalized Configuration

**Externalized Configuration** (cấu hình bên ngoài) là một tính năng cốt lõi của Spring Boot, cho phép bạn tách biệt cấu hình của ứng dụng khỏi mã nguồn chính. Điều này có nghĩa là bạn có thể thay đổi các thiết lập như cổng máy chủ, thông tin kết nối cơ sở dữ liệu, hoặc các tham số cụ thể của môi trường mà không cần phải biên dịch lại ứng dụng.

Spring Boot hỗ trợ nhiều nguồn cấu hình khác nhau, bao gồm:

* **application.properties** hoặc **application.yml**: Các tệp cấu hình mặc định trong thư mục **src/main/resources**.
* **Biến môi trường**: Các biến được định nghĩa ở cấp độ hệ điều hành.
* **Đối số dòng lệnh**: Các tham số được truyền vào khi chạy ứng dụng.

Ví dụ: java -jar myapp.jar --server.port=9000

* **Hệ thống quản lý cấu hình tập trung**: Như Spring Cloud Config Server (dành cho các hệ thống phức tạp hơn).

#### Profiles

**Profiles** (hồ sơ) trong Spring Boot là một cơ chế mạnh mẽ cho phép định nghĩa các **nhóm cấu hình khác nhau** và kích hoạt chúng dựa trên môi trường hiện tại. Ví dụ, ta có thể có một “profile dev” cho môi trường phát triển với cấu hình cơ sở dữ liệu cục bộ, và một “profile prod” cho môi trường sản xuất với cấu hình cơ sở dữ liệu từ xa và các thiết lập bảo mật cao hơn.

Để sử dụng Profiles, ta có thể tạo các tệp cấu hình như application-dev.properties và application-prod.properties. Khi muốn kích hoạt một profile, ta có thể thiết lập biến môi trường **spring.profiles.active** hoặc sử dụng đối số dòng lệnh **–spring.profiles.active=dev**. Tính năng này giúp quản lý cấu hình theo môi trường một cách rất linh hoạt và an toàn.

#### Logging

Spring Boot cung cấp khả năng cấu hình **Logging** (ghi log) một cách mạnh mẽ và linh hoạt. Mặc định, Spring Boot sử dụng **Logback** kết hợp với **SLF4J** cho mục đích logging, nhưng cũng hỗ trợ các hệ thống logging phổ biến khác như **Log4j2**.

Ta có thể dễ dàng cấu hình **mức độ log** (ví dụ: DEBUG, INFO, WARN, ERROR) và **định dạng đầu ra** của log thông qua file **application.properties** hoặc **application.yml**.

Ví dụ: Properties

logging.level.root=INFO

logging.level.com.interdata.demo=DEBUG

logging.file.name=my-app.log

#### Tự động cấu hình (Auto-configuration)

**Auto-configuration** là một trong những tính năng mạnh mẽ nhất của Spring Boot. Nó tự động cấu hình các bean và thiết lập ứng dụng dựa trên các thư viện (dependencies) có sẵn trong classpath.

Ví dụ: Auto-Configuration, ví dụ như “Spring Boot tự động cấu hình **JPA** khi thêm thư viện **Spring Data JPA** vào dự án, giúp tiết kiệm thời gian thiết lập cơ sở dữ liệu.

#### Starter Dependencies

**Starter Dependencies** là các **bộ sưu tập phụ thuộc** (POMs) tiện lợi mà Spring Boot cung cấp. Mỗi **“starter”** chứa tất cả các dependencies cần thiết cho một chức năng cụ thể. Ví dụ, **spring-boot-starter-web** bao gồm tất cả các thư viện cần thiết để phát triển ứng dụng web, bao gồm **Spring MVC**, **Tomcat nhúng**, **Jackson**, và các thư viện khác.

Điều này giúp giảm đáng kể số lượng dependencies cần khai báo trong file pom.xml của Maven hoặc build.gradle của Gradle, đồng thời đảm bảo tính tương thích giữa các phiên bản.

#### Máy chủ nhúng (Embedded Servers)

Spring Boot features cho phép **nhúng các máy chủ ứng dụng** phổ biến như **Tomcat**, **Jetty**, hoặc **Undertow** trực tiếp vào file **JAR** của ứng dụng. Điều này có nghĩa là ta có thể đóng gói toàn bộ ứng dụng của mình (bao gồm cả máy chủ) thành một **file JAR duy nhất** và chạy nó bằng lệnh **java -jar**.

Tính năng này đơn giản hóa đáng kể quá trình triển khai, loại bỏ nhu cầu cài đặt và cấu hình máy chủ ứng dụng riêng biệt.

#### Spring Boot Actuator

**Spring Boot Actuator** cung cấp các **điểm cuối (endpoints) HTTP** hoặc **JMX** để **giám sát và quản lý ứng dụng** trong môi trường production. Với Actuator, ta có thể **kiểm tra tình trạng ứng dụng**, xem thông tin về các **bean** được cấu hình, **đo lường hiệu suất**, và truy cập các **thông tin môi trường**. Đây là công cụ hữu ích cho DevOps và SRE để duy trì và khắc phục sự cố ứng dụng.

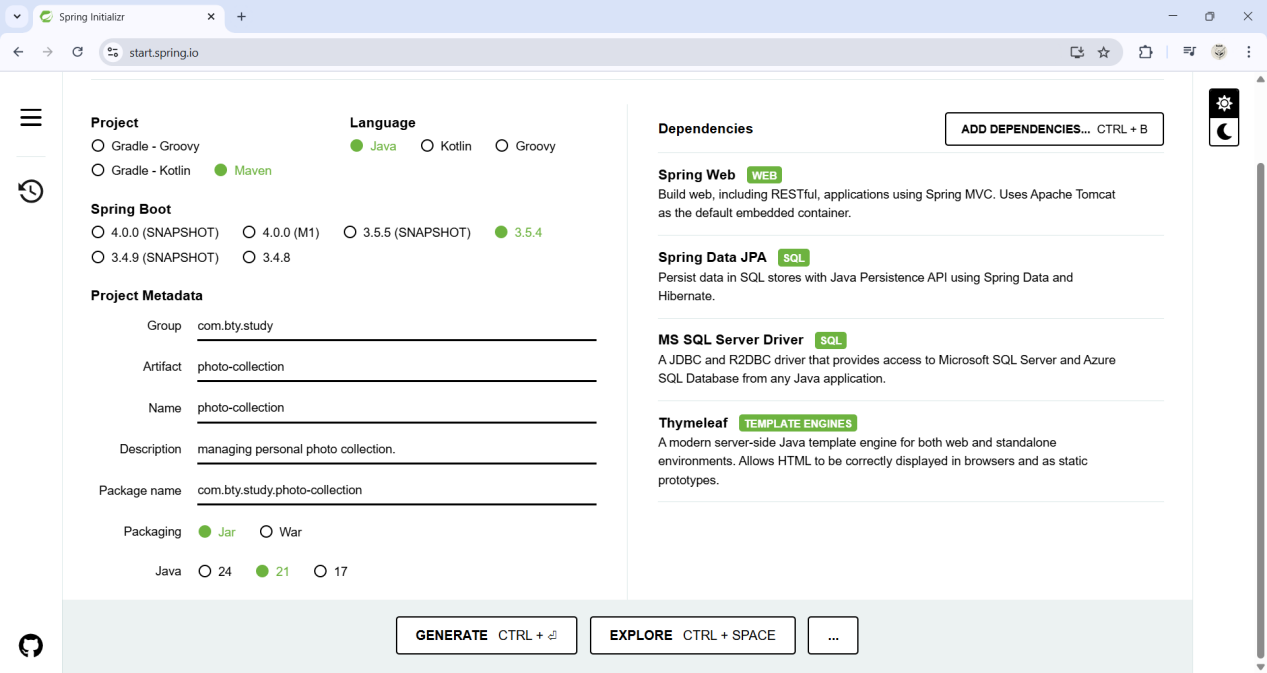
### Những lợi ích của việc sử dụng Spring Boot

* **Rapid Application Development**: Tạo và deploy ứng dụng nhanh chóng.
* **Simplified Configuration**: Làm giảm bớt những “boilerplate” cấu hình.
* **Embedded Servers**: Không cần thiết phải cài đặt và cấu hình server ở bên ngoài.
* **Cloud-native support**: Tích hợp liền mạch với các cloud platforms như AWS, Azure, và GCP.

## Spring Boot Programming Demo

Tạo Spring Boot project - **photo-collection** (Quản lý bộ sưa tập hình ảnh).

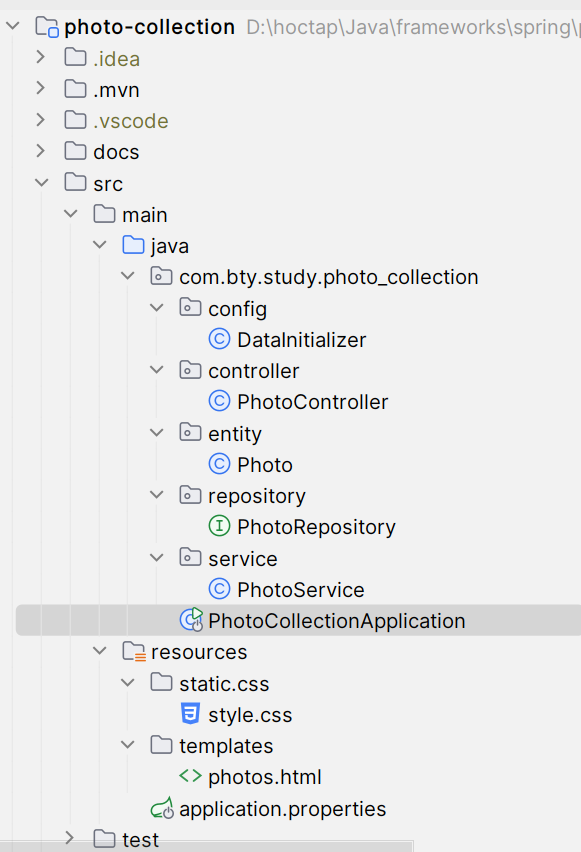
Ta sử dụng các dependencies như sau:



Cấu hình cho dự án trong file **src/main/resources/application.properties**



Tạo cấu trúc dự án như sau:



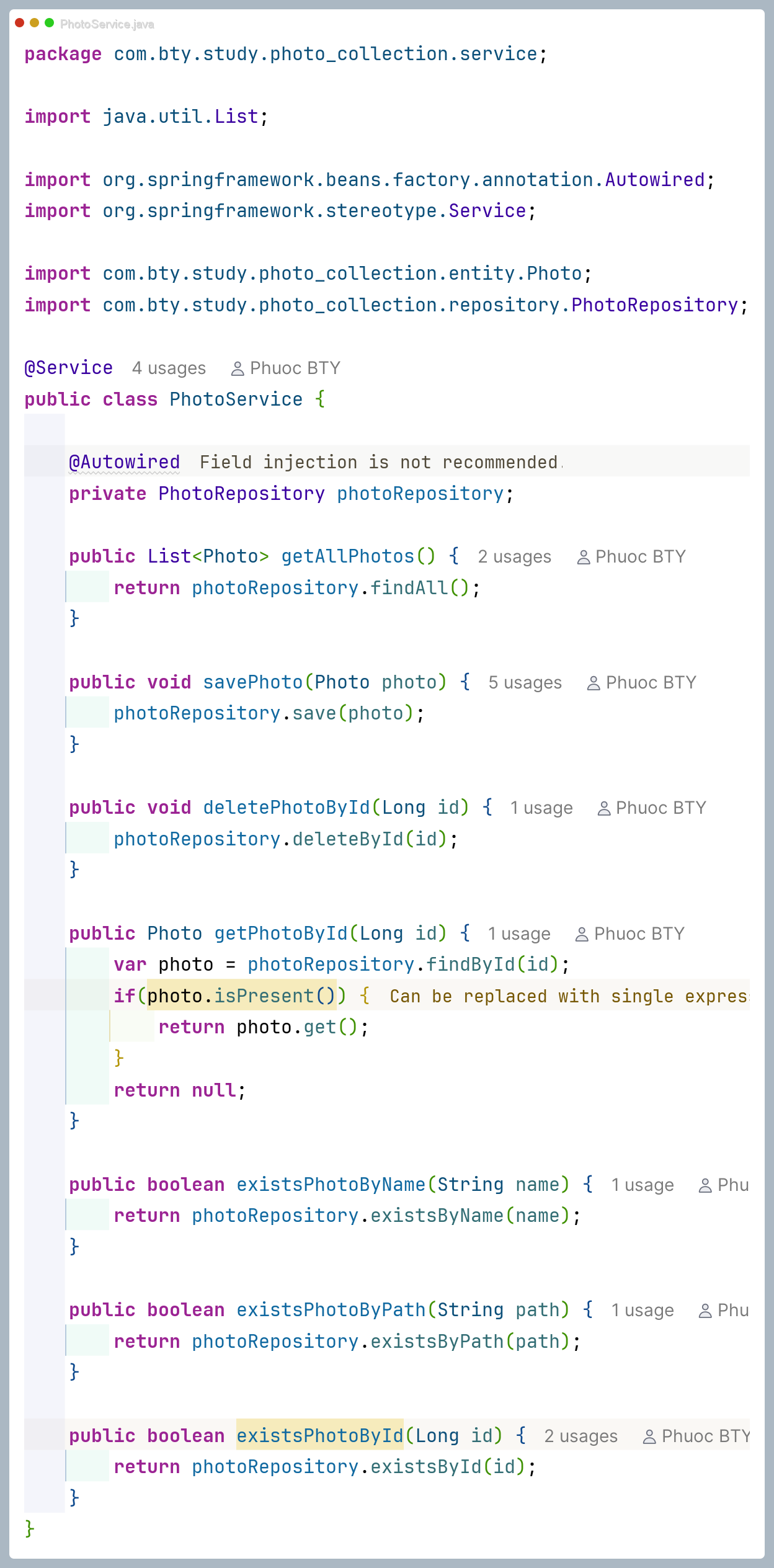
Tạo entity **Photo:**



Tạo interface **PhotoRepository**:



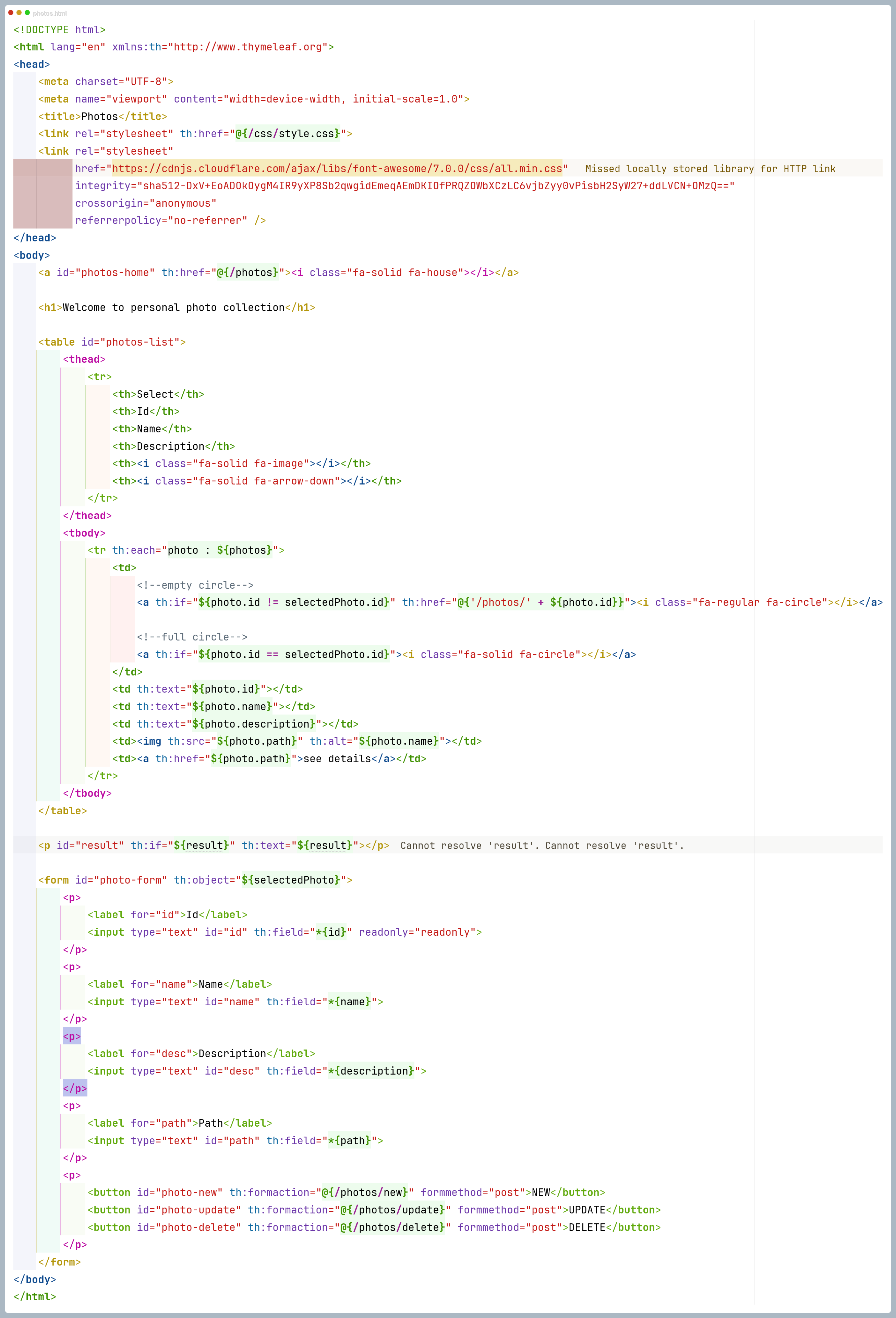
Tạo class **PhotoService**:



Tạo class **PhotoController**:



Tạo giao diện **photos.html**:



Tạo class **DataInitializer** để khởi tạo dữ liệu ban đầu cho ứng dụng:



Chạy ứng dụng spring boot:

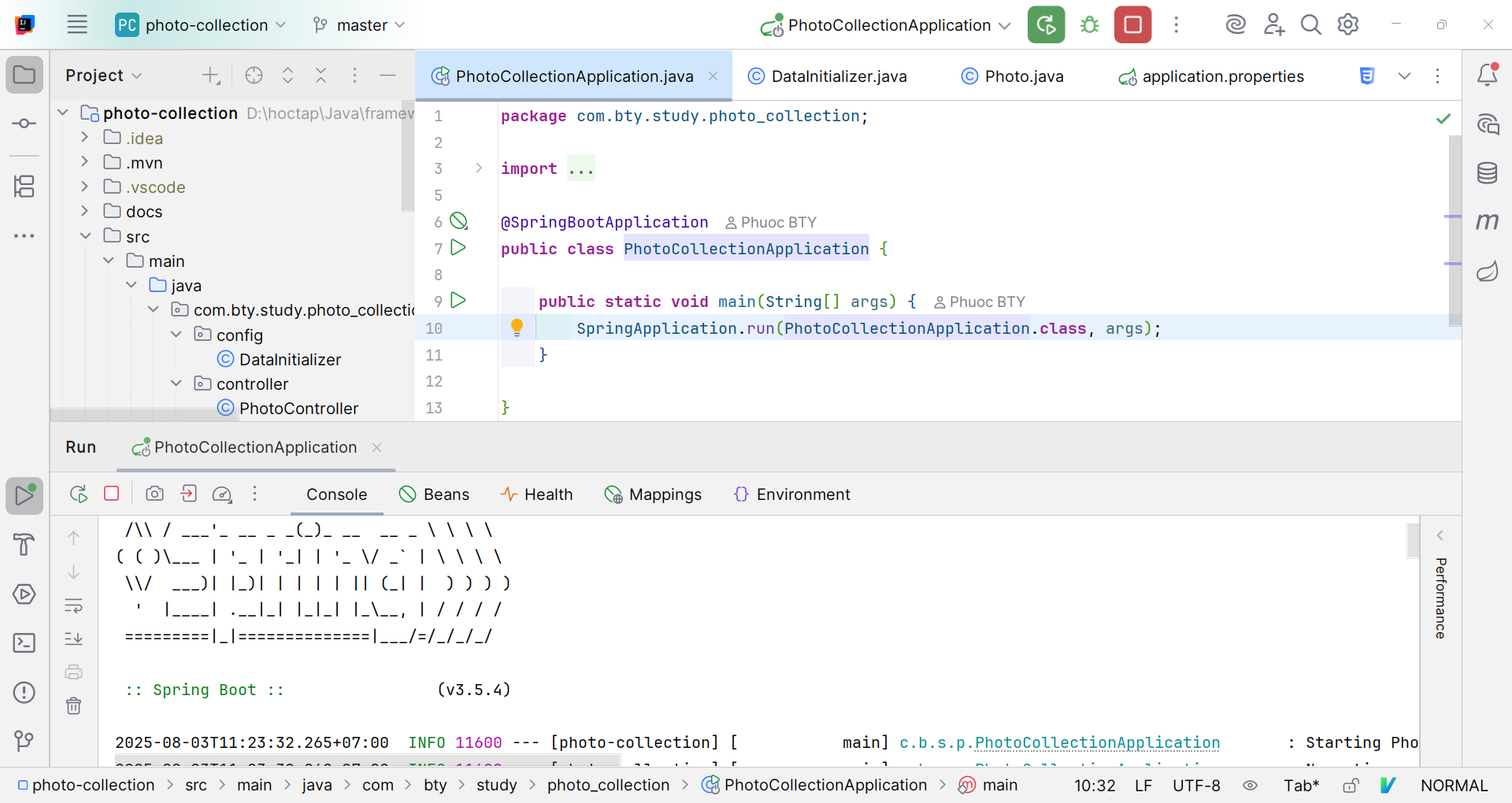
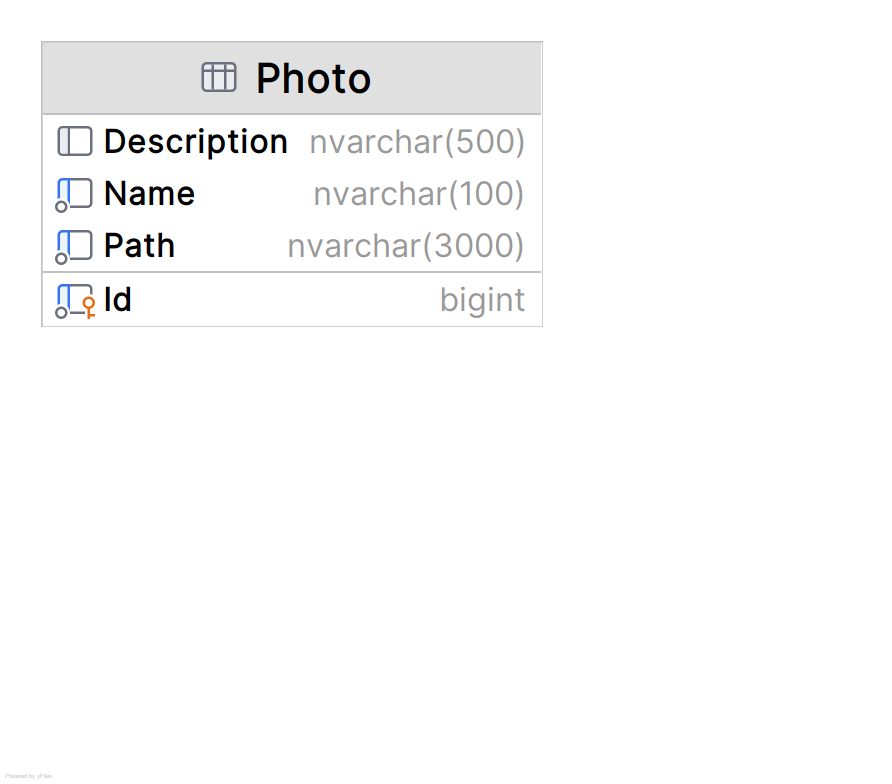
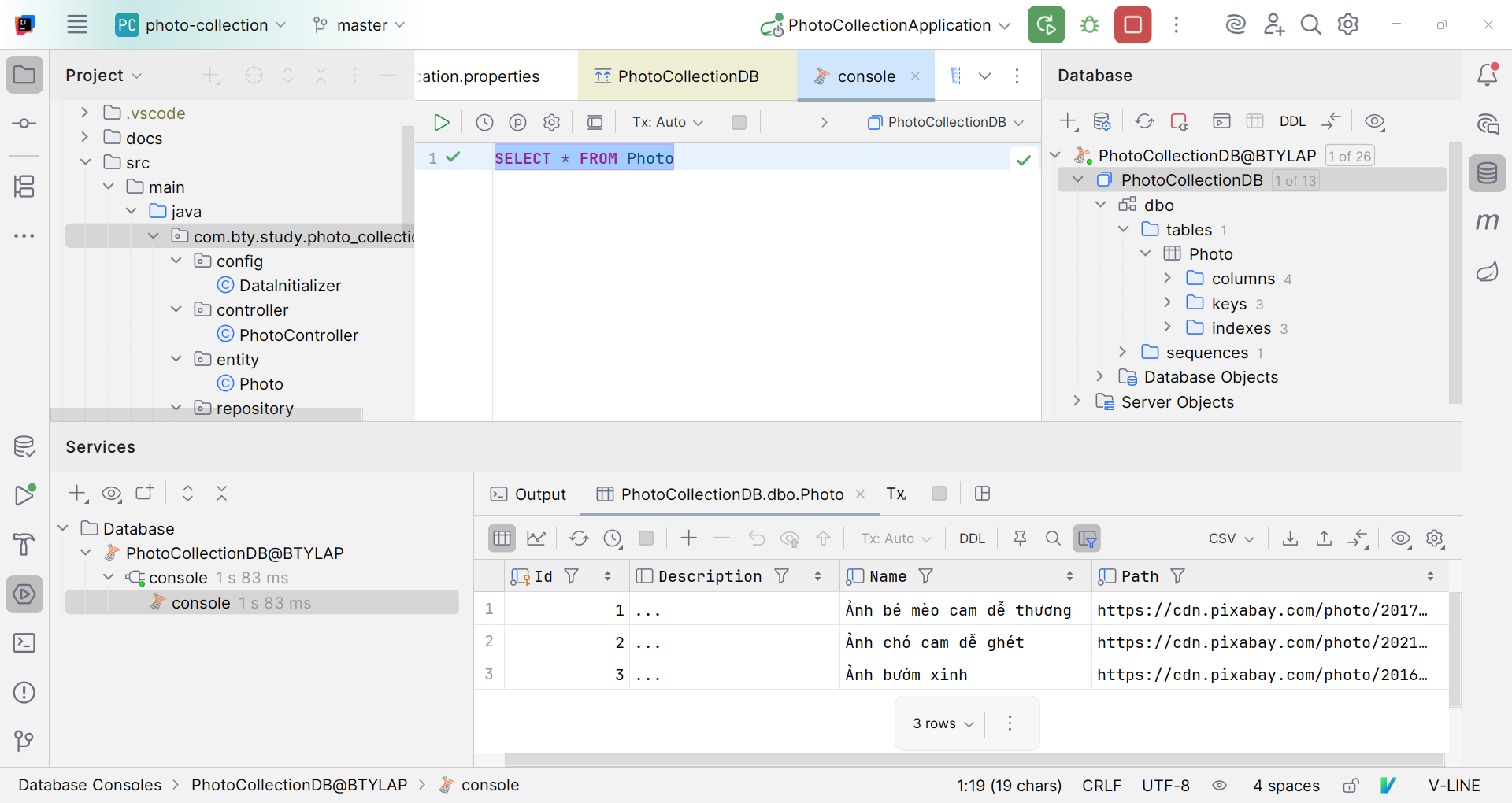
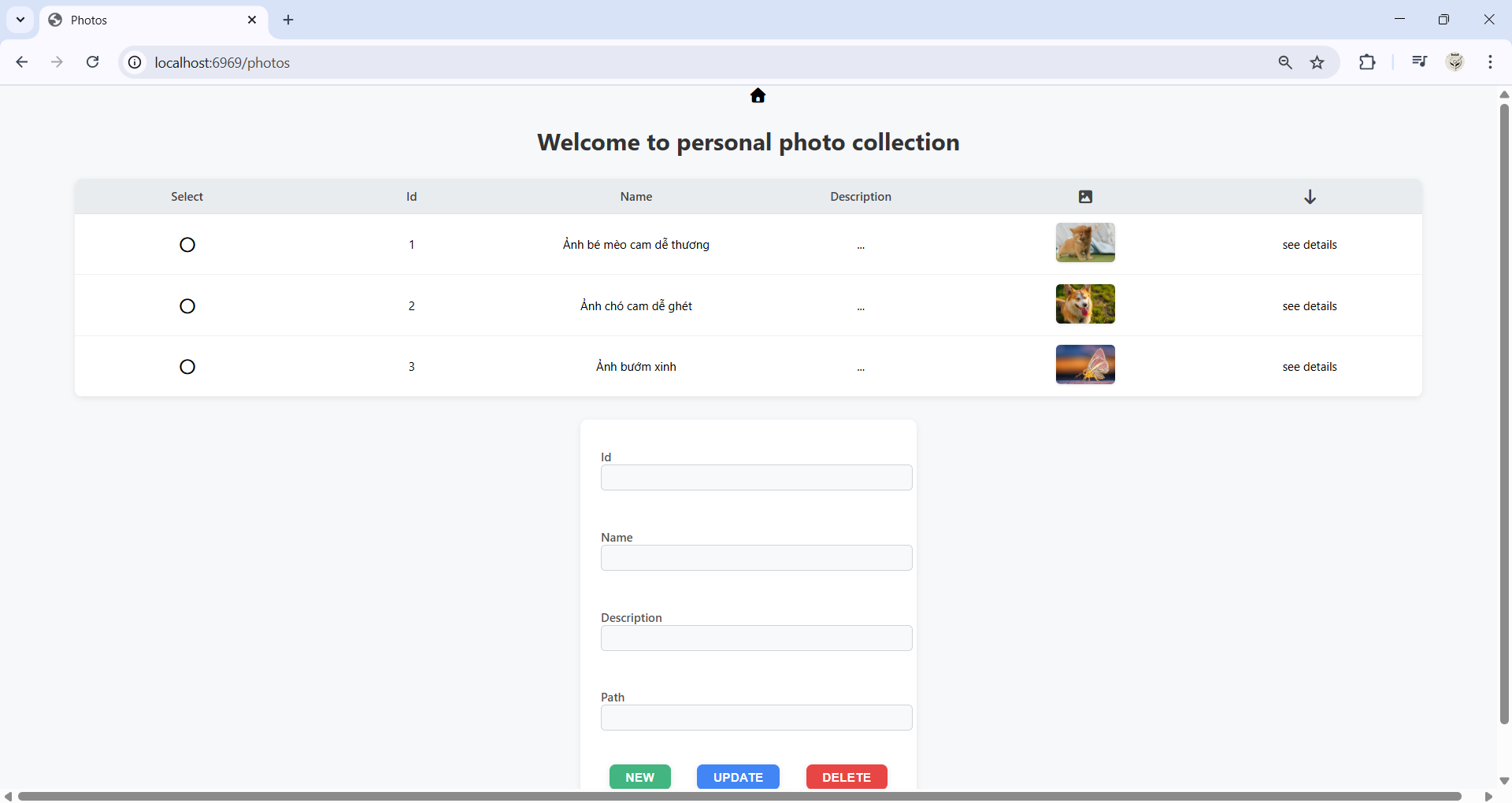


Table **Photo** được tự động tạo ra trong CSDL:

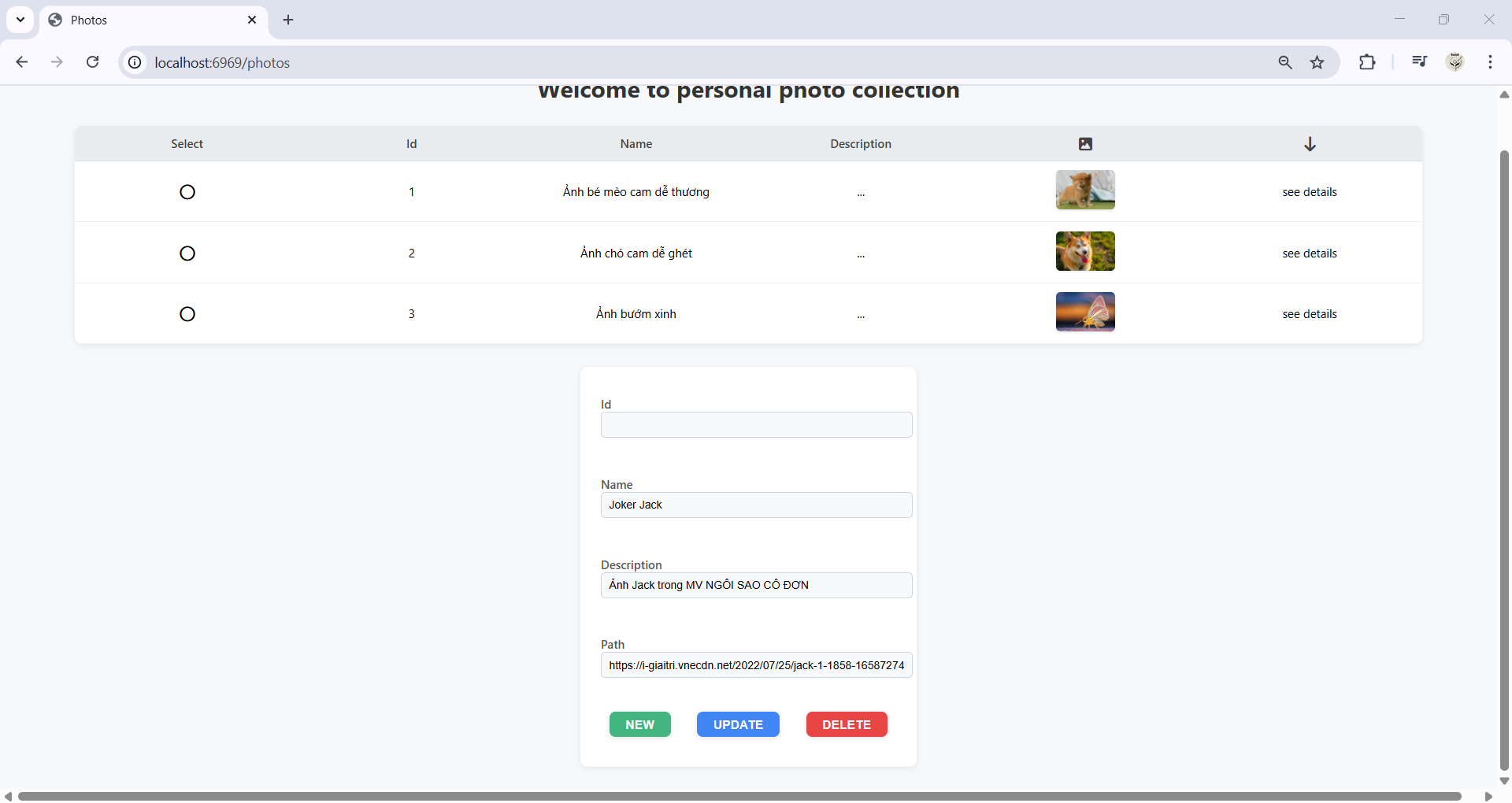


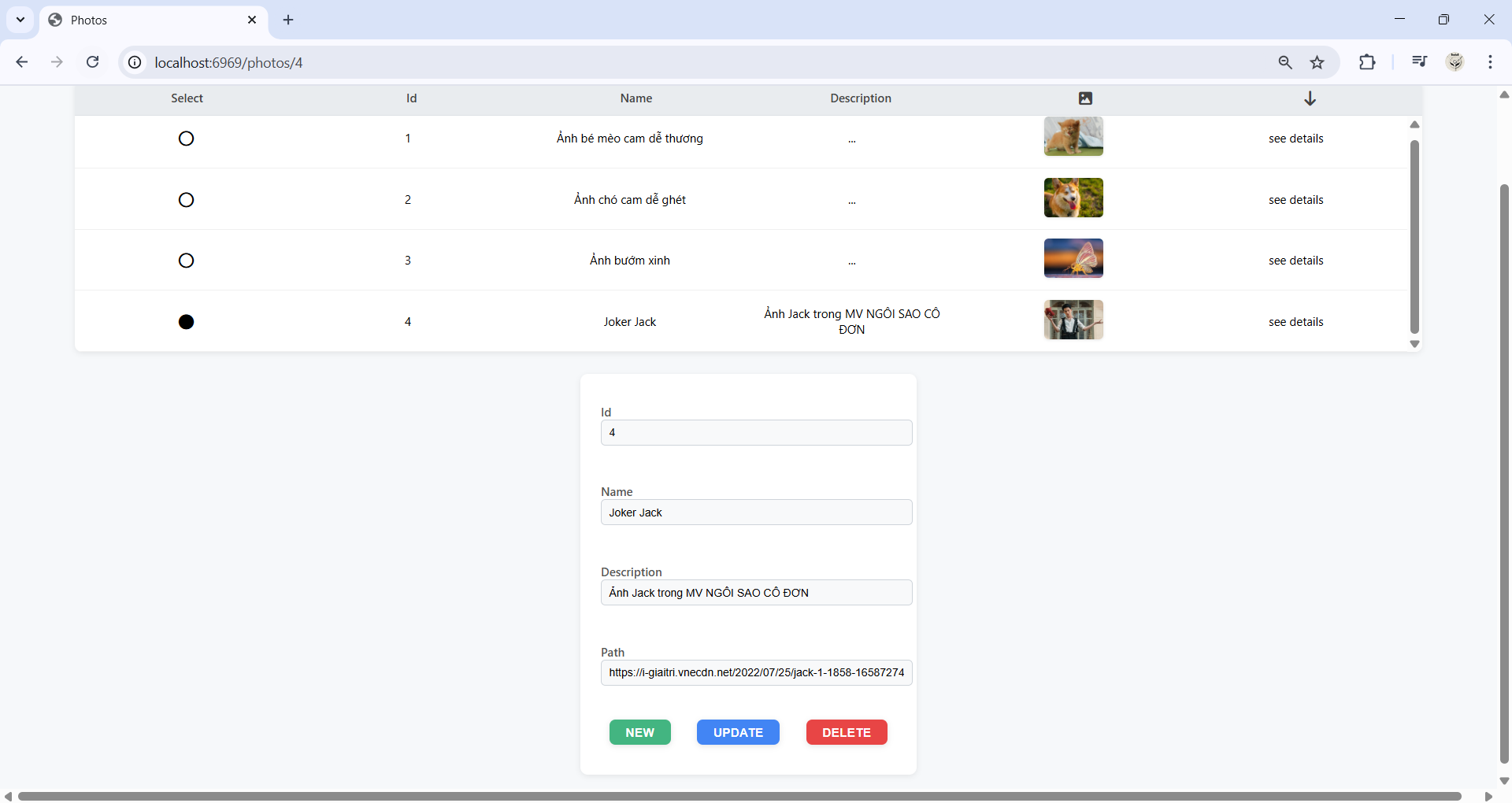


Truy cập danh sách các hình ảnh trên trình duyệt:

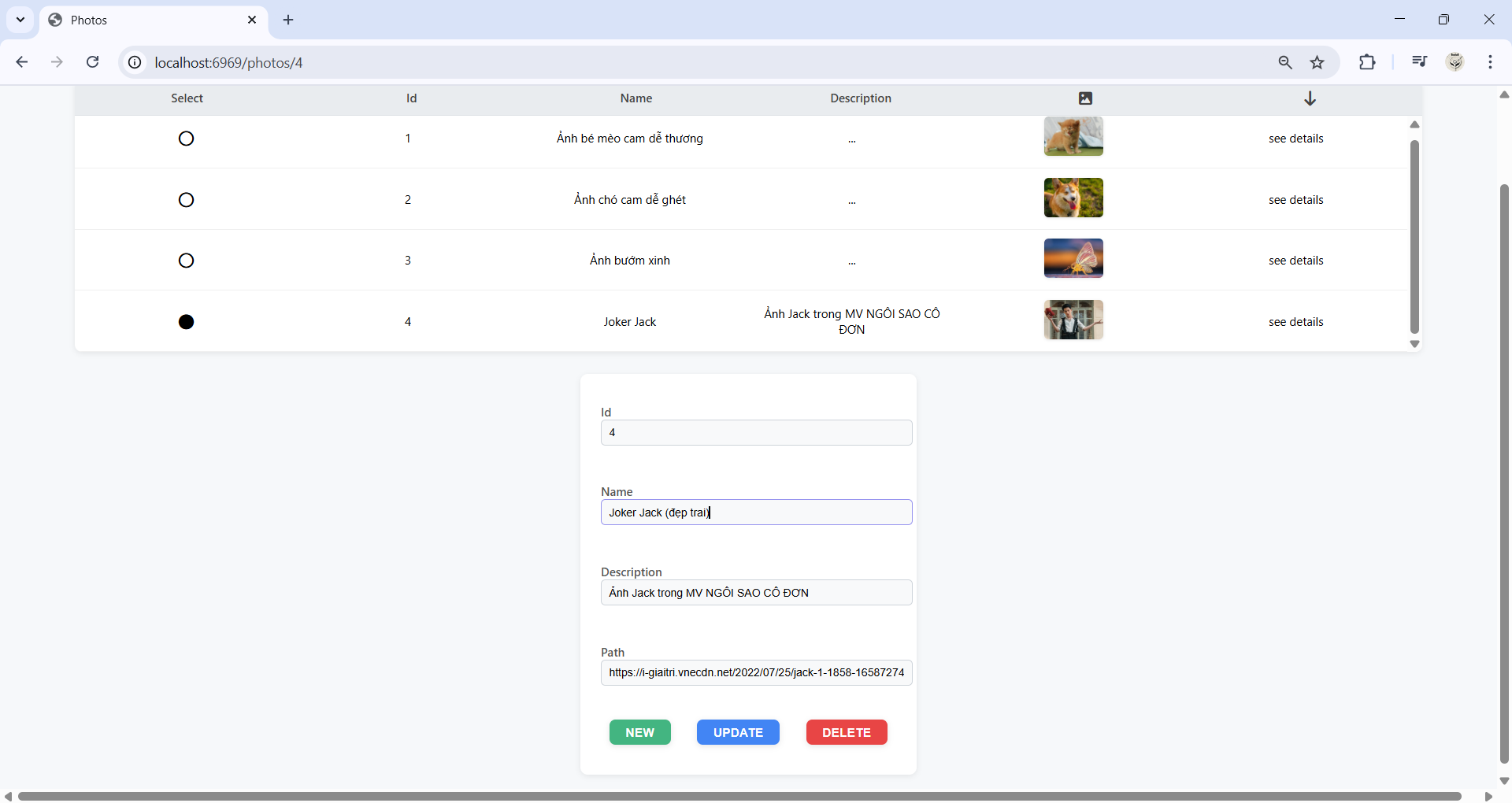


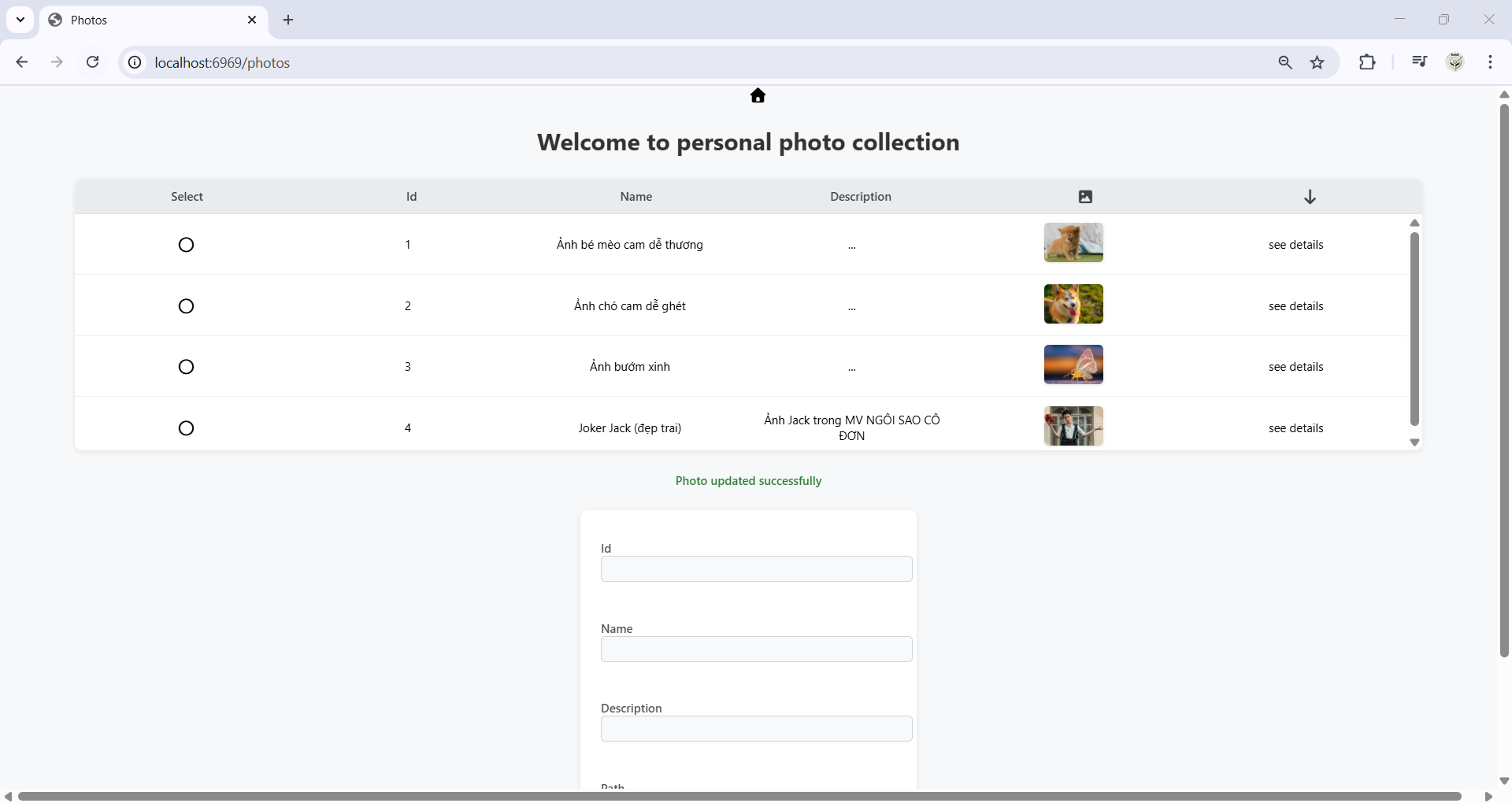
**Tạo mới** hình ảnh:





**Cập nhật** hình ảnh:





**Xóa** hình ảnh:

