[Thực hành] Ứng dụng quản lý khách hàng sử dụng Spring Boot

Mục tiêu

Luyện tập sử dụng Spring Boot

Điều kiện

Có kiến thức căn bản về việc sử dụng Spring Boot.

Mô tả

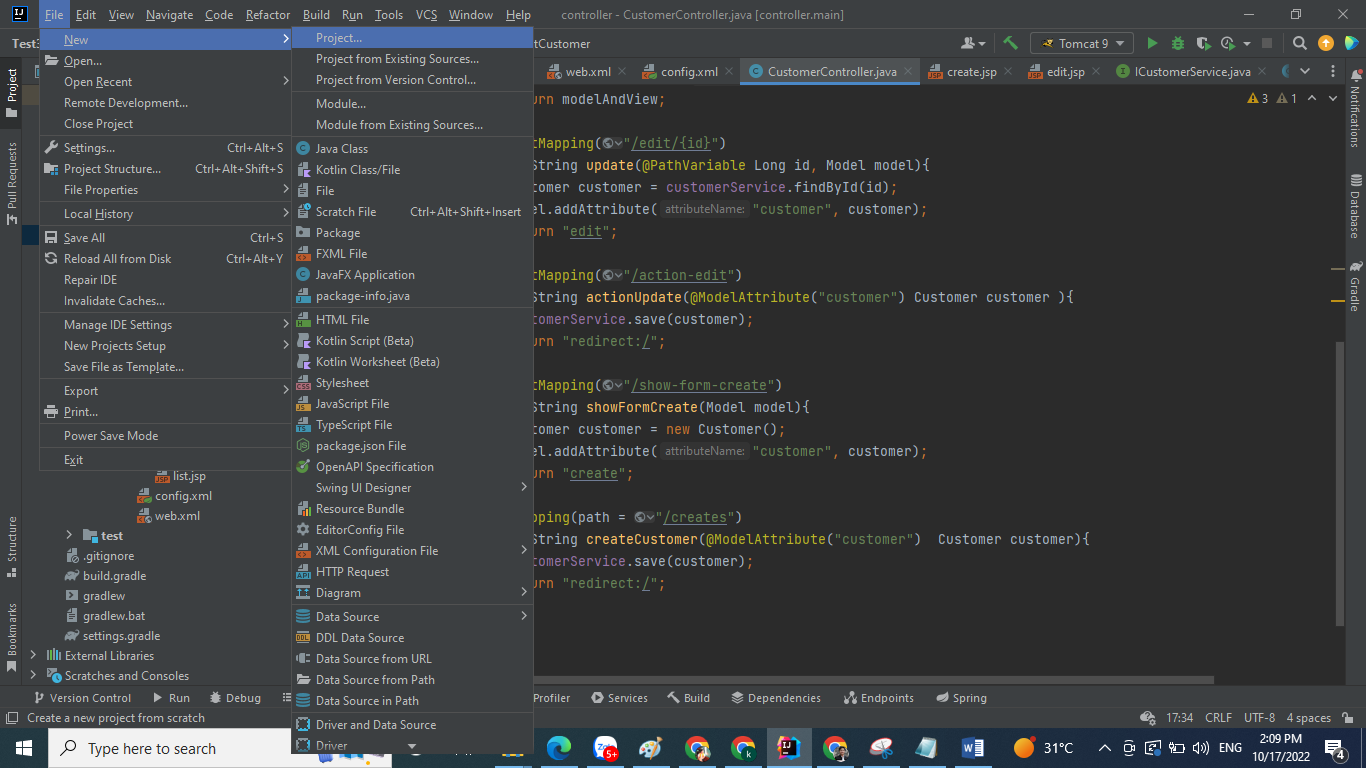
Trong phần này, chúng ta sẽ phát triển một ứng dụng quản lý khách hàng, sử dụng Spring Boot.

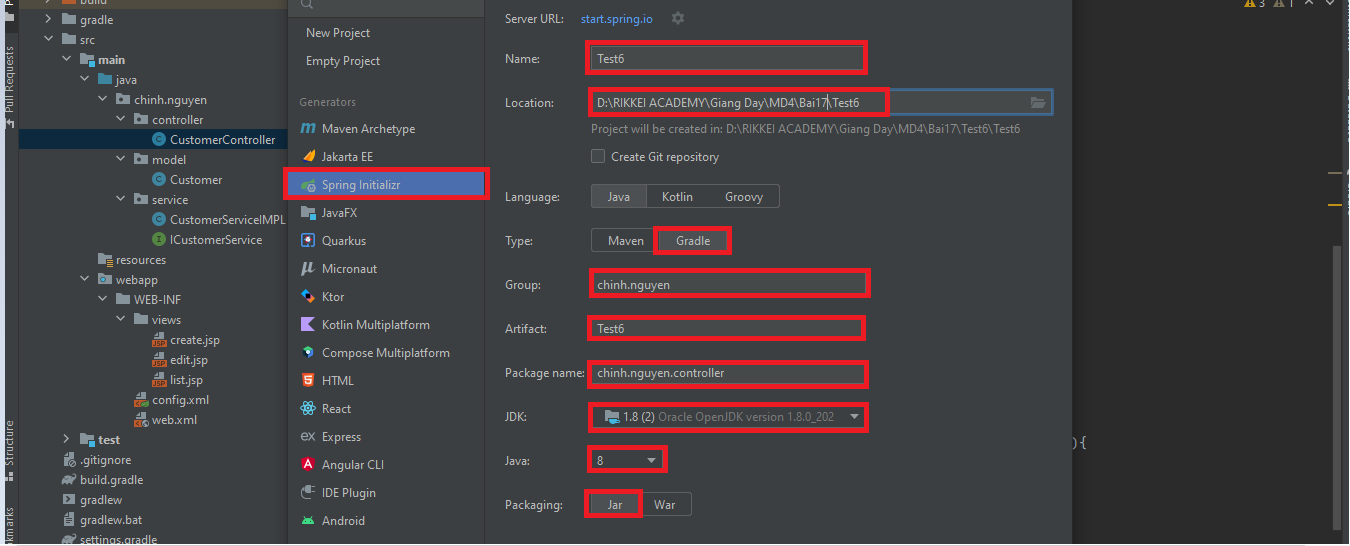
Ứng dụng có các chức năng chính:

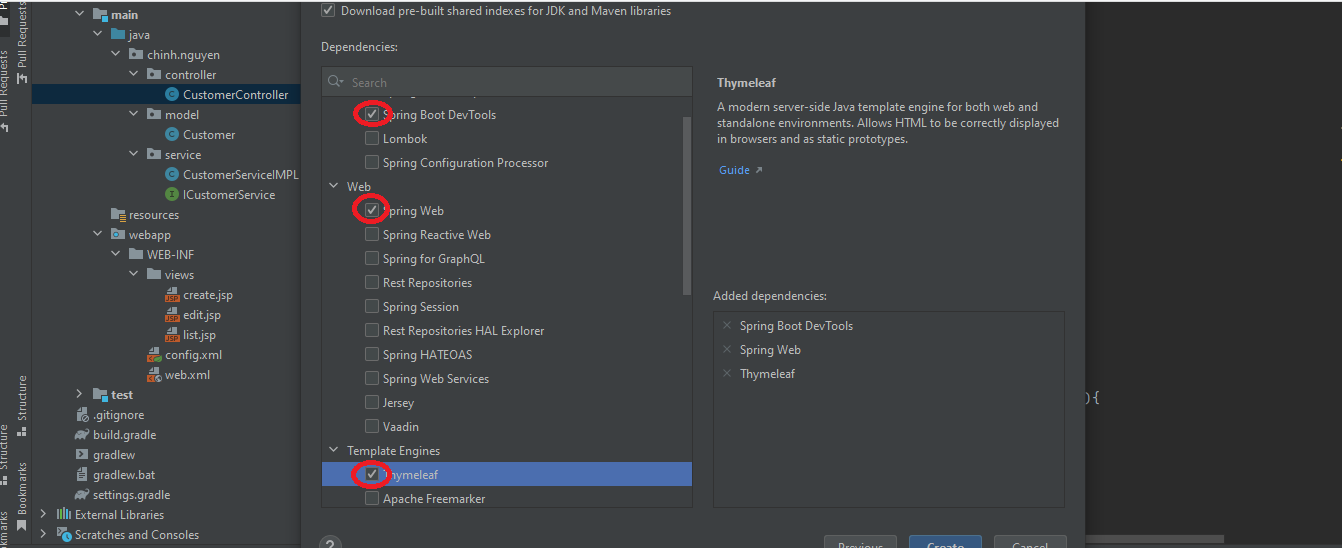
* Hiển thị danh sách khách hàng
* Thêm một khách hàng mới
* Xoá một khách hàng
* Chỉnh sửa thông tin khách hàng

Hướng dẫn

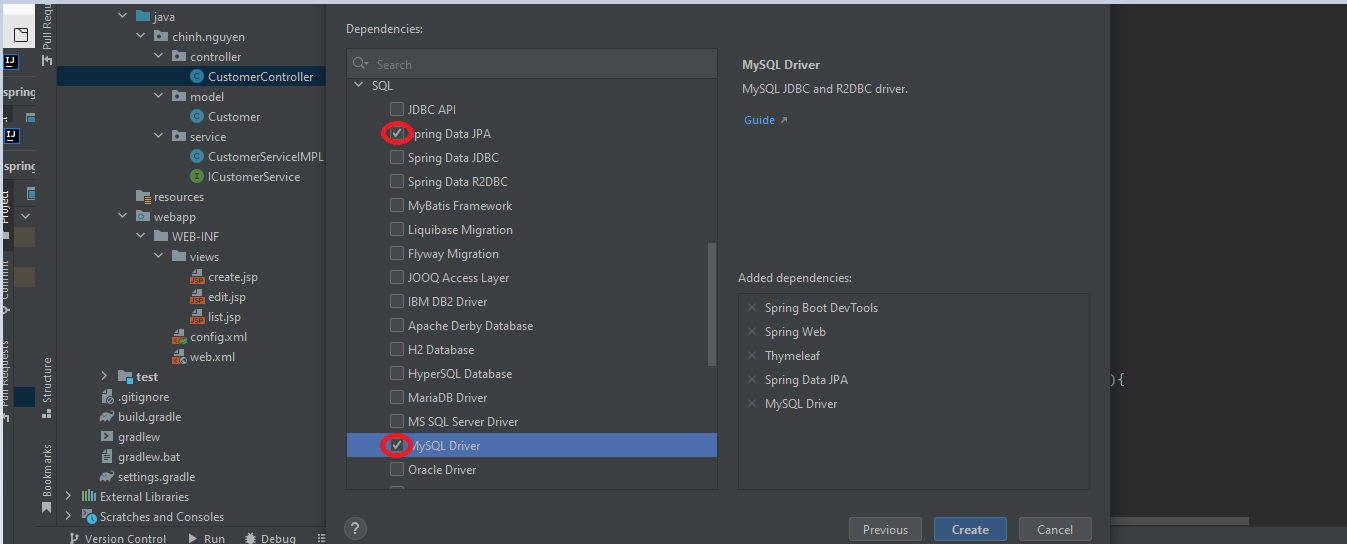
* Tạo dự án mới **Spring Boot:** File => New => Project







* Trong mục SQL ta tích chọn:



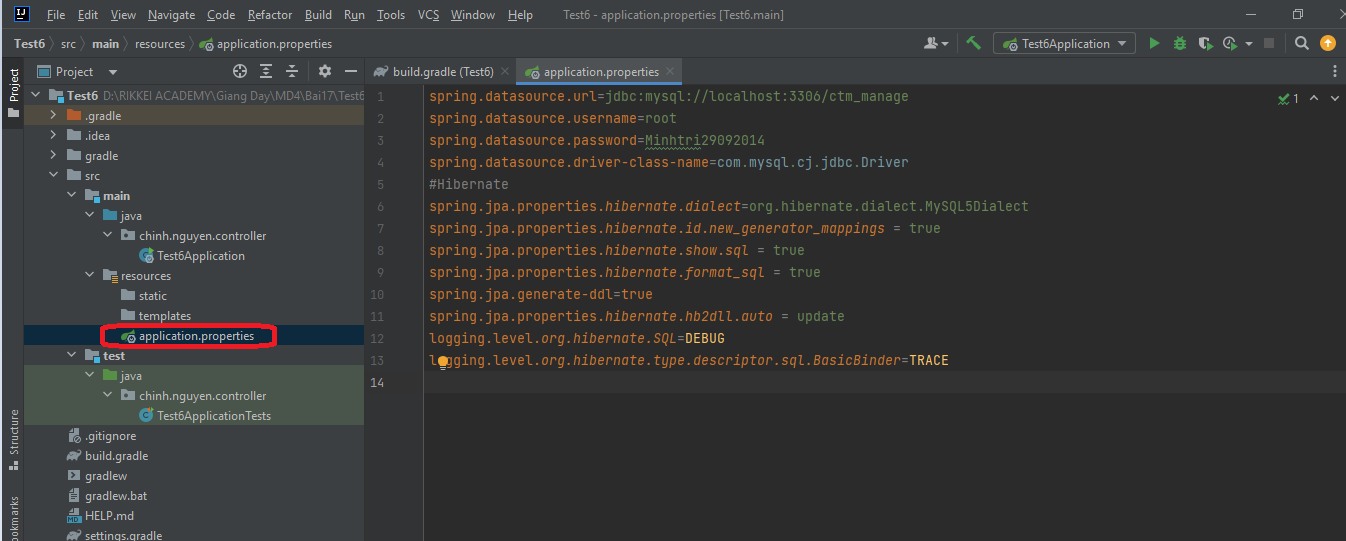
* Quan sát file: build.gradle sau các bước tạo dự án ở trên:

dependencies **{** implementation 'org.springframework.boot:spring-boot-starter-data-jpa'  
 implementation 'org.springframework.boot:spring-boot-starter-thymeleaf'  
 implementation 'org.springframework.boot:spring-boot-starter-web'  
 developmentOnly 'org.springframework.boot:spring-boot-devtools'  
 runtimeOnly 'mysql:mysql-connector-java'  
 testImplementation 'org.springframework.boot:spring-boot-starter-test'  
**}**

* Cấu hình ứng dụng trong file **application.properties (config MySQL)**

spring.datasource.url=jdbc:mysql://localhost:3306/ctm\_manage  
spring.datasource.username=root  
spring.datasource.password=Minhtri29092014  
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  
#Hibernate  
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect  
spring.jpa.properties.hibernate.id.new\_generator\_mappings = true  
spring.jpa.properties.hibernate.show.sql = true  
spring.jpa.properties.hibernate.format\_sql = true  
spring.jpa.generate-ddl=true  
spring.jpa.properties.hibernate.hb2dll.auto = update  
logging.level.org.hibernate.SQL=DEBUG  
logging.level.org.hibernate.type.descriptor.sql.BasicBinder=TRACE

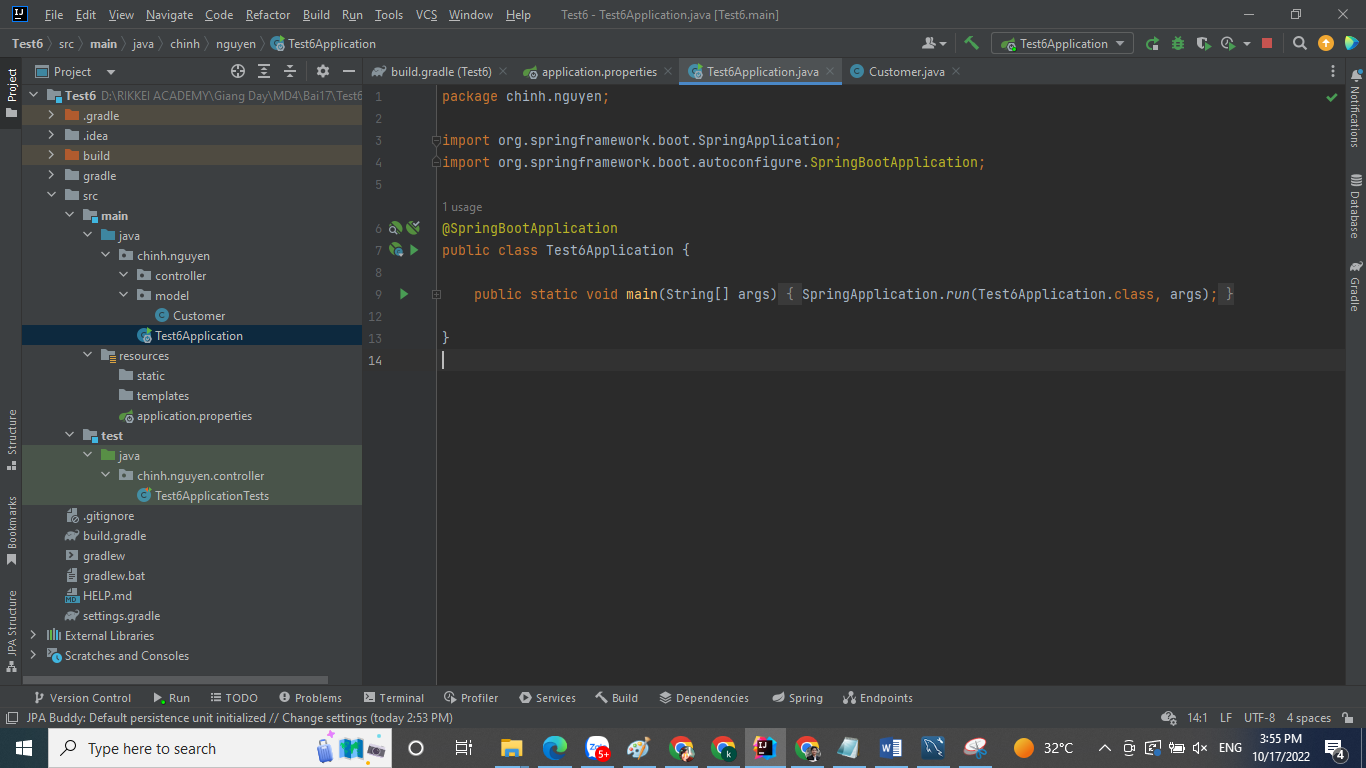
* Cấu trúc dự án như hình dưới:



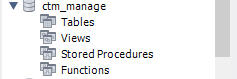
* Tạo package: **model** trong package: **chinh.nguyen** => Tạo class: **Customer** trong package này (class: này là 1 **Entity** ánh xạ xuống database)

package chinh.nguyen.model;  
  
import javax.persistence.\*;  
  
@Entity  
@Table(name = "customers")  
public class Customer {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
 private String firstName;  
 private String lastName;  
  
 public Customer() {  
 }  
  
 public Customer(String firstName, String lastName) {  
 this.firstName = firstName;  
 this.lastName = lastName;  
 }  
  
 @Override  
 public String toString() {  
 return String.*format*("Customer[id=%d, firstName='%s', lastName='%s']", id, firstName, lastName);  
 }  
  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public String getFirstName() {  
 return firstName;  
 }  
  
 public void setFirstName(String firstName) {  
 this.firstName = firstName;  
 }  
  
 public String getLastName() {  
 return lastName;  
 }  
  
 public void setLastName(String lastName) {  
 this.lastName = lastName;  
 }  
  
}

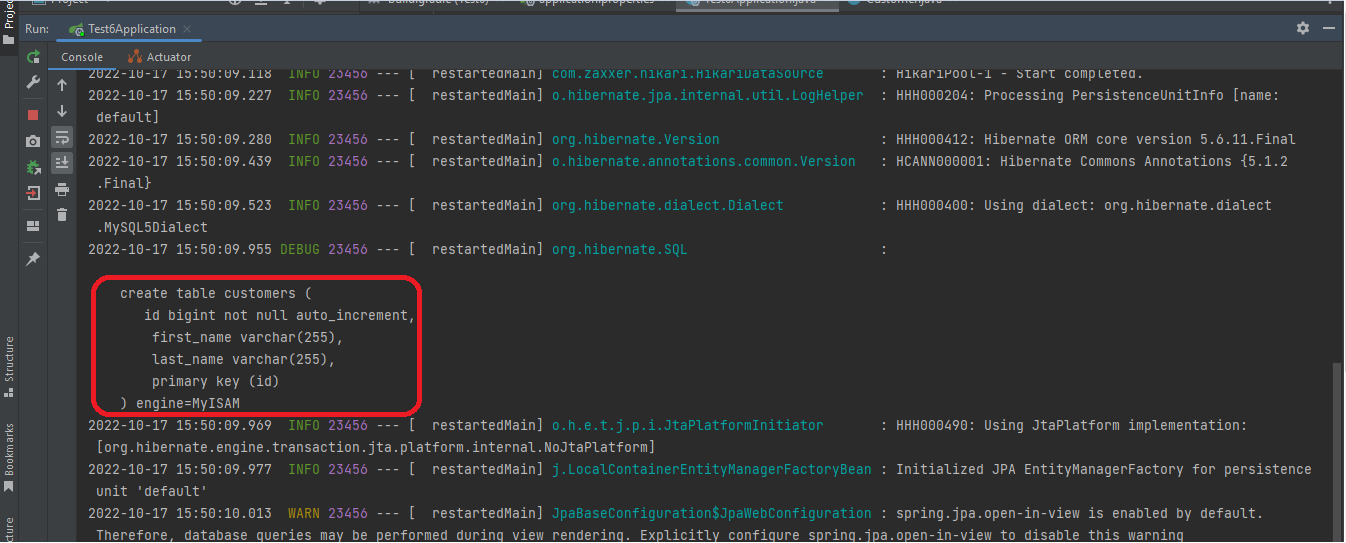
* Kéo class: **Test6Application** ra package gốc: **chinh.nguyen** => Cấu trúc dự án lúc này như sau:

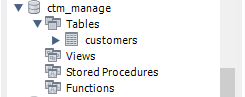


* Tạo database: **ctm\_manage** trong MySQL :



* Run Spring Boot và quan sát kết quả trong database:





* Tạo interface **ICustomerRepository** kế thừa interface **JpaRepository** có sẵn của Spring trong package: **chinh.nguyen.repository**

package chinh.nguyen.repository;  
  
import chinh.nguyen.model.Customer;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.stereotype.Repository;  
  
@Repository  
public interface ICustomerRepository extends JpaRepository<Customer, Long> {  
}

* Tạo các **service**: interface **IGeneralService** khai báo các phương thức chung cho các **service**

package chinh.nguyen.service;  
  
import java.util.Optional;  
  
public interface IGeneralService<T> {  
 Iterable<T> findAll();  
  
 Optional<T> findById(Long id);  
  
 void save(T t);  
  
 void remove(Long id);  
  
}

* interface **ICustomerService** khai báo các phương thức riêng cho Customer được được trong package **chinh.nguyen.service.customer**

package chinh.nguyen.service.customer;  
  
import chinh.nguyen.model.Customer;  
import chinh.nguyen.service.IGeneralService;  
  
public interface ICustomerService extends IGeneralService<Customer> {  
}

Lớp **CustomerServiceIMPL** triển khai các phương thức cho interface **ICustomerService**

package chinh.nguyen.service.customer;  
  
import chinh.nguyen.model.Customer;  
import chinh.nguyen.repository.ICustomerRepository;  
import org.springframework.beans.factory.annotation.Autowired;  
  
import java.util.Optional;  
  
public class CustomerServiceIMPL implements ICustomerService{  
 @Autowired  
 private ICustomerRepository customerRepository;  
  
 @Override  
 public Iterable<Customer> findAll() {  
 return customerRepository.findAll();  
 }  
  
 @Override  
 public Optional<Customer> findById(Long id) {  
 return customerRepository.findById(id);  
 }  
  
 @Override  
 public void save(Customer customer) {  
 customerRepository.save(customer);  
 }  
  
 @Override  
 public void remove(Long id) {  
 customerRepository.deleteById(id);  
 }  
  
}

* Tạo **CustomerController** trong package: **chinh.nguyen.controller**

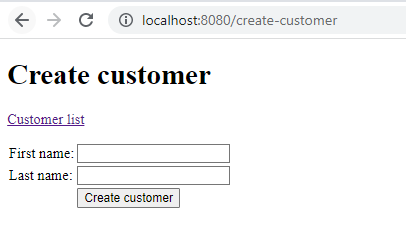
package chinh.nguyen.controller;  
  
import chinh.nguyen.model.Customer;  
import chinh.nguyen.service.customer.ICustomerService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.ModelAttribute;  
import org.springframework.web.bind.annotation.PostMapping;  
import org.springframework.web.servlet.ModelAndView;  
  
@Controller  
public class CustomerController {  
 @Autowired  
 private ICustomerService customerService;  
  
 @GetMapping("/create-customer")  
 public ModelAndView showCreateForm() {  
 ModelAndView modelAndView = new ModelAndView("/customer/create");  
 modelAndView.addObject("customer", new Customer());  
 return modelAndView;  
 }  
  
 @PostMapping("/create-customer")  
 public ModelAndView saveCustomer(@ModelAttribute("customer") Customer customer) {  
 customerService.save(customer);  
 ModelAndView modelAndView = new ModelAndView("/customer/create");  
 modelAndView.addObject("customer", new Customer());  
 modelAndView.addObject("message", "New customer created successfully");  
 return modelAndView;  
 }

* Tạo file **create.html** trong thư mục **resources/templates/customer**

<!DOCTYPE html>  
<html lang="en" xmlns:th="http://www.thymeleaf.org">  
<head>  
 <meta charset="UTF-8">  
 <title>Create customer</title>  
</head>  
<body>  
<h1>Create customer</h1>  
<p>  
 <a href="/customers">Customer list</a>  
</p>  
<th:block th:if="${message}">  
 <p th:text="${message}"></p>  
</th:block>  
<form th:action="@{/create-customer}" th:object="${customer}" method="post">  
 <table>  
 <tr>  
 <td>First name:</td>  
 <td><input type="text" th:field="\*{firstName}"/></td>  
 </tr>  
 <tr>  
 <td>Last name:</td>  
 <td><input type="text" th:field="\*{lastName}"/></td>  
 </tr>  
 <tr>  
 <td></td>  
 <td><input type="submit" value="Create customer"></td>  
 </tr>  
 </table>  
</form>  
</body>  
</html>

* Chạy ứng dụng và quan sát kết quả

Đi đến đường dẫn http://localhost:8080/create-customer và nhập dữ liệu của Customer.



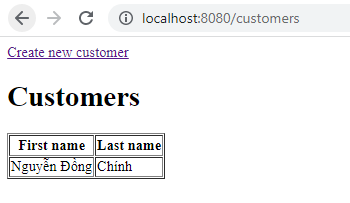
* Nhập dữ liệu và kiểm tra dưới database.
* Tạo chức năng hiển thị danh sách khách hàng
* Cập nhật **CustomerController**

@GetMapping("/customers")  
public ModelAndView listCustomers() {  
 ModelAndView modelAndView = new ModelAndView("/customer/list");  
 modelAndView.addObject("customers", customerService.findAll());  
 return modelAndView;  
}

* Tạo file **list.html** trong thư mục **resources/templates/customer**

<!DOCTYPE html>  
<html lang="en" xmlns:th="http://www.thymeleaf.org">  
<head>  
 <meta charset="UTF-8">  
 <title>Title</title>  
</head>  
<body>  
<a href="/create-customer">Create new customer</a>  
<h1>Customers</h1>  
<table border="1">  
 <tr>  
 <th>First name</th>  
 <th>Last name</th>  
 </tr>  
 <th:block th:each="customer : ${customers}">  
 <tr>  
 <td th:text="${customer.firstName}"></td>  
 <td th:text="${customer.lastName}"></td>  
 </tr>  
 </th:block>  
</table>  
</body>  
</html>

Chạy ứng dụng và đi đến trang http://localhost:8080/customers để quan sát kết quả.



* Tạo chức năng cập nhật Customer

@GetMapping("/edit-customer/{id}")  
public ModelAndView showEditForm(@PathVariable Long id) {  
 Optional<Customer> customer = customerService.findById(id);  
 if (customer.isPresent()) {  
 ModelAndView modelAndView = new ModelAndView("/customer/edit");  
 modelAndView.addObject("customer", customer.get());  
 return modelAndView;  
 } else {  
 return new ModelAndView("/error.404");  
 }  
}  
  
@PostMapping("/edit-customer")  
public ModelAndView updateCustomer(@ModelAttribute("customer") Customer customer) {  
 customerService.save(customer);  
 ModelAndView modelAndView = new ModelAndView("/customer/edit");  
 modelAndView.addObject("customer", customer);  
 modelAndView.addObject("message", "Customer updated successfully");  
 return modelAndView;  
}

* Tạo file **edit.html** trong thư mục **resources/templates/customer**

<!DOCTYPE html>  
<html lang="en" xmlns:th="http://www.thymeleaf.org">  
<head>  
 <meta charset="UTF-8">  
 <title>Edit customer</title>  
</head>  
<body>  
<h1>Edit customer</h1>  
<p>  
 <a href="/customers">Customer list</a>  
</p>  
<th:block th:if="${message}">  
 <p th:text="${message}"></p>  
</th:block>  
<form th:action="@{/edit-customer}" th:object="${customer}" method="post">  
 <input th:type="hidden" name="id" th:field="\*{id}">  
 <table>  
 <tr>  
 <td>First name:</td>  
 <td><input type="text" th:field="\*{firstName}"/></td>  
 </tr>  
 <tr>  
 <td>Last name:</td>  
 <td><input type="text" th:field="\*{lastName}"/></td>  
 </tr>  
 <tr>  
 <td></td>  
 <td><input type="submit" value="Update customer"></td>  
 </tr>  
 </table>  
</form>  
</body>  
</html>

Tạo file **error.404.html** trong thư mục **resources/templates**

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Error 404</title>  
</head>  
<body>  
NOT FOUND  
</body>  
</html>

* Cập nhật file **/templates/customer/list.html**, thêm đường link đến trang edit.

<!DOCTYPE html>  
<html lang="en" xmlns:th="http://www.thymeleaf.org">  
<head>  
 <meta charset="UTF-8">  
 <title>Title</title>  
</head>  
<body>  
<a href="/create-customer">Create new customer</a>  
<h1>Customers</h1>  
<table border="1">  
 <tr>  
 <th>First name</th>  
 <th>Last name</th>  
 <th>Edit</th>  
 </tr>  
 <th:block th:each="customer : ${customers}">  
 <tr>  
 <td th:text="${customer.firstName}"></td>  
 <td th:text="${customer.lastName}"></td>  
 <td><a th:href="@{/edit-customer/\_\_${customer.id}\_\_ }">Edit</a></td>  
 </tr>  
 </th:block>  
</table>  
</body>  
</html>

* Chạy ứng dụng và test chức năng Edit.
* Tạo chức năng xoá Customer

Cập nhật CustomerController

@GetMapping("/delete-customer/{id}")  
public ModelAndView showDeleteForm(@PathVariable Long id) {  
 Optional<Customer> customer = customerService.findById(id);  
 if (customer.isPresent()) {  
 ModelAndView modelAndView = new ModelAndView("/customer/delete");  
 modelAndView.addObject("customer", customer.get());  
 return modelAndView;  
  
 } else {  
 return new ModelAndView("/error.404");  
 }  
}  
  
@PostMapping("/delete-customer")  
public String deleteCustomer(@ModelAttribute("customer") Customer customer) {  
 customerService.remove(customer.getId());  
 return "redirect:customers";  
}

* Tạo file **delete.html** trong thư mục **resources/templates/customer**

<!DOCTYPE html>  
<html lang="en" xmlns:th="http://www.thymeleaf.org">  
<head>  
 <meta charset="UTF-8">  
 <title>Delete customer</title>  
</head>  
<body>  
<h1>Delete customer</h1>  
<h2>Are you sure?</h2>  
<p>  
 <a href="/customers">Customer list</a>  
</p>  
<form th:action="@{/delete-customer}" th:object="${customer}" method="post">  
 <input th:type="hidden" name="id" th:field="\*{id}">  
 <table>  
 <tr>  
 <td>First name:</td>  
 <td th:text="${customer.firstName}"></td>  
 </tr>  
 <tr>  
 <td>Last name:</td>  
 <td th:text="${customer.lastName}"></td>  
 </tr>  
 <tr>  
 <td></td>  
 <td><input type="submit" value="Delete customer"></td>  
 </tr>  
 </table>  
</form>  
</body>  
</html>

Cập nhật file **/templates/customer/list.html**, thêm đường link đến trang delete.

<!DOCTYPE html>  
<html lang="en" xmlns:th="http://www.thymeleaf.org">  
<head>  
 <meta charset="UTF-8">  
 <title>Title</title>  
</head>  
<body>  
<a href="/create-customer">Create new customer</a>  
<h1>Customers</h1>  
<table border="1">  
 <tr>  
 <th>First name</th>  
 <th>Last name</th>  
 <th>Edit</th>  
 <th>Delete</th>  
 </tr>  
 <th:block th:each="customer : ${customers}">  
 <tr>  
 <td th:text="${customer.firstName}"></td>  
 <td th:text="${customer.lastName}"></td>  
 <td><a th:href="@{/edit-customer/\_\_${customer.id}\_\_ }">Edit</a></td>  
 <td><a th:href="@{/delete-customer/\_\_${customer.id}\_\_ }">Delete</a></td>  
 </tr>  
 </th:block>  
</table>  
</body>  
</html>

* Chạy ứng dụng và test chức năng delete.

<https://github.com/nguyendongminhtri/MD4-Bai17-TH2-CRUD-SpringBoot-ThymeLeaf-MySQL.git>

* ***Hà Nội : Đêm chở gió – 17/10/2022 – Chính NĐ \****

Hướng dẫn nộp bài:

Up code lên github.

Paste link github vào phần nộp bài.