[Thực hành] Ứng dụng Quản lý khách hàng: Quản lý tỉnh

Mục tiêu

Luyện tập sử dụng Spring Data Repositories.

Mô tả

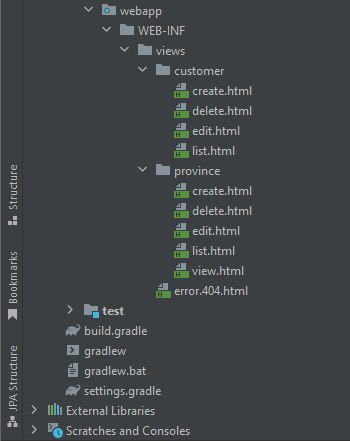
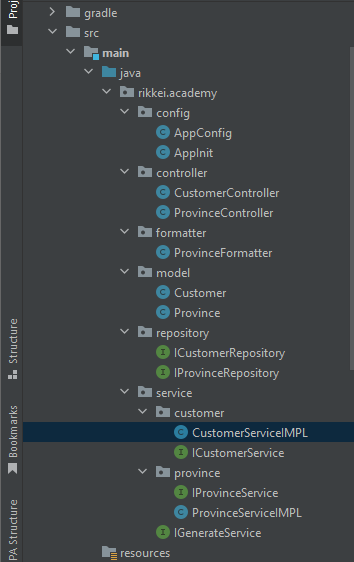
Trong phần này, chúng ta sẽ thay thế việc tự triển khai tầng repository bằng cách sử dụng Spring Data Repositories.

Cùng với đó, chúng ta sẽ bổ sung các tính năng mới sau:

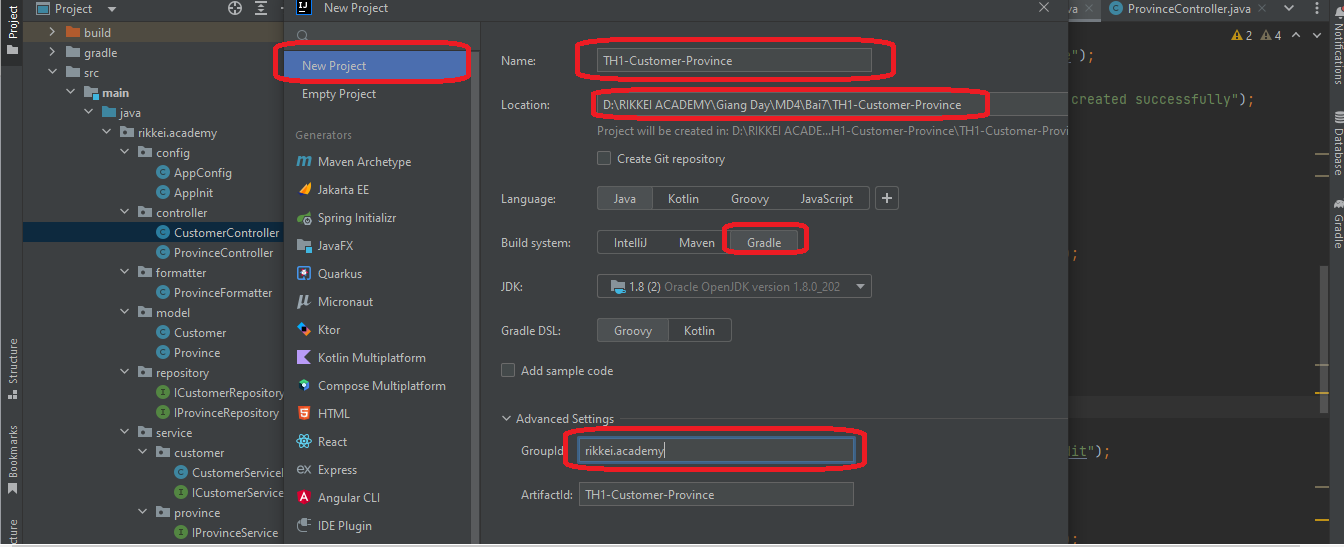
* Quản lý các tỉnh thành của khách hàng
* Hiển thị khách hàng theo từng tỉnh thành

Hướng dẫn

* Mô tả cấu trúc của toàn bộ dự án:



* Tạo project với hình ảnh dưới đây:



* Import thư viện vào : **build.gradle**

compileOnly('javax.servlet:javax.servlet-api:4.0.1')  
implementation group: 'org.springframework', name: 'spring-core', version: '5.3.2'  
implementation group: 'org.springframework', name: 'spring-context', version: '5.3.2'  
implementation group: 'org.springframework', name: 'spring-beans', version: '5.3.2'  
implementation group: 'org.springframework', name: 'spring-web', version: '5.3.2'  
implementation group: 'org.springframework', name: 'spring-webmvc', version: '5.3.2'  
implementation group: 'org.thymeleaf', name: 'thymeleaf-spring5', version: '3.0.11.RELEASE'  
implementation group: 'nz.net.ultraq.thymeleaf', name: 'thymeleaf-layout-dialect', version: '2.5.2'  
implementation group: 'org.hibernate', name: 'hibernate-core', version: '5.3.0.Final'  
implementation group: 'org.hibernate', name: 'hibernate-entitymanager', version: '5.3.0.Final'  
implementation group: 'org.springframework', name: 'spring-orm', version: '5.3.2'  
implementation group: 'mysql', name: 'mysql-connector-java', version: '8.0.22'  
implementation group: 'org.springframework.data', name: 'spring-data-jpa', version: '2.4.2'

* Tạo package: rikkei.academy.config => Tạo 2 class **AppInit** vs **AppConfig:**
* **AppConfig**:

package rikkei.academy.config;  
  
import org.springframework.beans.BeansException;  
import org.springframework.beans.factory.annotation.Qualifier;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.ApplicationContextAware;  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.ComponentScan;  
import org.springframework.context.annotation.Configuration;  
import org.springframework.data.jpa.repository.config.EnableJpaRepositories;  
import org.springframework.format.FormatterRegistry;  
import org.springframework.jdbc.datasource.DriverManagerDataSource;  
import org.springframework.orm.jpa.JpaTransactionManager;  
import org.springframework.orm.jpa.JpaVendorAdapter;  
import org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean;  
import org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter;  
import org.springframework.transaction.PlatformTransactionManager;  
import org.springframework.transaction.annotation.EnableTransactionManagement;  
import org.springframework.web.servlet.config.annotation.EnableWebMvc;  
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;  
import org.thymeleaf.spring5.SpringTemplateEngine;  
import org.thymeleaf.spring5.templateresolver.SpringResourceTemplateResolver;  
import org.thymeleaf.spring5.view.ThymeleafViewResolver;  
import org.thymeleaf.templatemode.TemplateMode;  
import rikkei.academy.formatter.ProvinceFormatter;  
import rikkei.academy.repository.ICustomerRepository;  
import rikkei.academy.service.customer.CustomerServiceIMPL;  
import rikkei.academy.service.customer.ICustomerService;  
import rikkei.academy.service.province.IProvinceService;  
import rikkei.academy.service.province.ProvinceServiceIMPL;  
  
  
import javax.persistence.EntityManager;  
import javax.persistence.EntityManagerFactory;  
import javax.sql.DataSource;  
import java.util.Properties;  
  
@Configuration  
@EnableWebMvc  
@EnableTransactionManagement  
@ComponentScan("rikkei.academy.controller")  
@EnableJpaRepositories("rikkei.academy.repository")  
public class AppConfig implements WebMvcConfigurer, ApplicationContextAware {  
 private ApplicationContext applicationContext;  
  
 @Override  
 public void setApplicationContext(ApplicationContext applicationContext) throws BeansException {  
 this.applicationContext = applicationContext;  
 }  
  
 //Cấu hình Thymleaf  
 @Bean  
 public SpringResourceTemplateResolver templateResolver() {  
 SpringResourceTemplateResolver templateResolver = new SpringResourceTemplateResolver();  
 templateResolver.setApplicationContext(applicationContext);  
 templateResolver.setPrefix("/WEB-INF/views");  
 templateResolver.setSuffix(".html");  
 templateResolver.setTemplateMode(TemplateMode.*HTML*);  
 templateResolver.setCharacterEncoding("UTF-8");  
 return templateResolver;  
 }  
  
 @Bean  
 public SpringTemplateEngine templateEngine() {  
 SpringTemplateEngine templateEngine = new SpringTemplateEngine();  
 templateEngine.setTemplateResolver(templateResolver());  
 return templateEngine;  
 }  
  
 @Bean  
 public ThymeleafViewResolver viewResolver() {  
 ThymeleafViewResolver viewResolver = new ThymeleafViewResolver();  
 viewResolver.setTemplateEngine(templateEngine());  
 viewResolver.setCharacterEncoding("UTF-8");  
 viewResolver.setContentType("UTF-8");  
 return viewResolver;  
 }  
  
 //Cấu hình JPA  
 @Bean  
 @Qualifier(value = "entityManager")  
 public EntityManager entityManager(EntityManagerFactory entityManagerFactory) {  
 return entityManagerFactory.createEntityManager();  
 }  
  
 @Bean  
 public LocalContainerEntityManagerFactoryBean entityManagerFactory() {  
 LocalContainerEntityManagerFactoryBean em = new LocalContainerEntityManagerFactoryBean();  
 em.setDataSource(dataSource());  
 em.setPackagesToScan("rikkei.academy.model");  
  
 JpaVendorAdapter vendorAdapter = new HibernateJpaVendorAdapter();  
 em.setJpaVendorAdapter(vendorAdapter);  
 em.setJpaProperties(additionalProperties());  
 return em;  
 }  
  
 @Bean  
 public DataSource dataSource() {  
 DriverManagerDataSource dataSource = new DriverManagerDataSource();  
 dataSource.setDriverClassName("com.mysql.cj.jdbc.Driver");  
 dataSource.setUrl("jdbc:mysql://localhost:3306/bai7\_th1\_province");  
 dataSource.setUsername("root");  
 dataSource.setPassword("Minhtri29092014");  
 return dataSource;  
 }  
  
 @Bean  
 public PlatformTransactionManager transactionManager(EntityManagerFactory emf) {  
 JpaTransactionManager transactionManager = new JpaTransactionManager();  
 transactionManager.setEntityManagerFactory(emf);  
 return transactionManager;  
 }  
  
 public Properties additionalProperties() {  
 Properties properties = new Properties();  
 properties.setProperty("hibernate.hbm2ddl.auto", "update");  
 properties.setProperty("hibernate.dialect", "org.hibernate.dialect.MySQL5Dialect");  
 return properties;  
 }  
  
}

* **AppInit**:

package rikkei.academy.config;  
  
  
import org.springframework.web.filter.CharacterEncodingFilter;  
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;  
  
  
import javax.servlet.Filter;  
  
public class AppInit extends AbstractAnnotationConfigDispatcherServletInitializer {  
  
 @Override  
 protected Class<?>[] getRootConfigClasses() {  
 return new Class[]{AppConfig.class};  
 }  
  
 @Override  
 protected Class<?>[] getServletConfigClasses() {  
 return new Class[0];  
 }  
  
 @Override  
 protected String[] getServletMappings() {  
 return new String[]{"/"};  
 }  
 @Override  
 protected Filter[] getServletFilters() {  
 CharacterEncodingFilter filter = new CharacterEncodingFilter();  
 filter.setForceEncoding(true);  
 filter.setEncoding("UTF-8");  
 return new Filter[]{filter};  
 }  
  
}

* Tạo package: **model** => Tạo class: **Province:**

package rikkei.academy.model;  
  
  
import javax.persistence.\*;  
import java.util.List;  
  
@Entity  
@Table(name = "provinces")  
public class Province {  
 @Id  
 @GeneratedValue(strategy= GenerationType.*IDENTITY*)  
 private Long id;  
  
 private String name;  
  
 @OneToMany(targetEntity = Customer.class)  
 private List<Customer> customers;  
  
 public Province() {  
 }  
  
 public Province(String name) {  
 this.name = name;  
 }  
  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public List<Customer> getCustomers() {  
 return customers;  
 }  
  
 public void setCustomers(List<Customer> customers) {  
 this.customers = customers;  
 }  
}

* Tạo class: **Customer** trong package: model

package rikkei.academy.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;  
 @ManyToOne  
 @JoinColumn(name = "province\_id")  
 private Province province;  
  
  
 public Customer() {  
 }  
  
 public Customer(String firstName, String lastName) {  
 this.firstName = firstName;  
 this.lastName = lastName;  
 }  
  
 public Customer(String firstName, String lastName, Province province) {  
 this.firstName = firstName;  
 this.lastName = lastName;  
 this.province = province;  
 }  
  
 @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;  
 }  
  
 public Province getProvince() {  
 return province;  
 }  
  
 public void setProvince(Province province) {  
 this.province = province;  
 }  
}

* Tạo package: **repository:** => Tầng này tạo ra nhằm mục đích tạo các câu query truy vấn xuống database
* Tạo interface: **IProvinceRepository** trong package: repository

package rikkei.academy.repository;  
  
import org.springframework.data.repository.PagingAndSortingRepository;  
import org.springframework.stereotype.Repository;  
import rikkei.academy.model.Customer;  
import rikkei.academy.model.Province;  
  
@Repository  
public interface IProvinceRepository extends PagingAndSortingRepository<Province, Long> {  
  
}

* Tạo interface: **ICustomerRepository** trong package: repository

package rikkei.academy.repository;  
  
import org.springframework.data.repository.PagingAndSortingRepository;  
import org.springframework.stereotype.Repository;  
import rikkei.academy.model.Customer;  
import rikkei.academy.model.Province;  
  
@Repository  
public interface ICustomerRepository extends PagingAndSortingRepository<Customer, Long> {  
 Iterable<Customer> findAllByProvince(Province province);  
}

* Tạo package: **service**
* Tạo interface: **IGenerateService** trong package: service

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

* Tạo package: **province** trong package: **service** => Tạo: **IProvinceService** trong package này

package rikkei.academy.service.province;  
  
import rikkei.academy.model.Province;  
import rikkei.academy.service.IGenerateService;  
  
public interface IProvinceService extends IGenerateService<Province> {  
}

* Tạo class: **ProvinceServiceIMPL** triển khai interface trên:

package rikkei.academy.service.province;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
import rikkei.academy.model.Province;  
import rikkei.academy.repository.IProvinceRepository;  
  
import java.util.Optional;  
  
@Service  
public class ProvinceServiceIMPL implements IProvinceService{  
 @Autowired  
 private IProvinceRepository provinceRepository;  
  
 @Override  
 public Iterable<Province> findAll() {  
 return provinceRepository.findAll();  
 }  
  
 @Override  
 public Optional<Province> findById(Long id) {  
 return provinceRepository.findById(id);  
 }  
  
 @Override  
 public void save(Province province) {  
 provinceRepository.save(province);  
 }  
  
 @Override  
 public void remove(Long id) {  
 provinceRepository.deleteById(id);  
 }  
  
}

* Tạo pakace: **customer** trong pakace: **service**
* Tạo interface: **ICustomerService** trong package này:
* package rikkei.academy.service.customer;  
    
  import rikkei.academy.model.Customer;  
  import rikkei.academy.model.Province;  
  import rikkei.academy.service.IGenerateService;  
    
  public interface ICustomerService extends IGenerateService<Customer> {  
   Iterable<Customer> findAllByProvince(Province province);  
  }
* Tạo tiếp class: **CustomerServiceIMPL** triển khai interface trên:

package rikkei.academy.service.customer;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import rikkei.academy.model.Customer;  
import rikkei.academy.model.Province;  
import rikkei.academy.repository.ICustomerRepository;  
  
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);  
 }  
  
 @Override  
 public Iterable<Customer> findAllByProvince(Province province) {  
 return customerRepository.findAllByProvince(province);  
 }  
}

* Tạo package: formatter (trong package: rikkei.academy)
* Tạo class: **ProvinceFormatter:**

package rikkei.academy.formatter;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.format.Formatter;  
import org.springframework.stereotype.Component;  
import rikkei.academy.model.Province;  
import rikkei.academy.service.province.IProvinceService;  
  
import java.text.ParseException;  
import java.util.Locale;  
import java.util.Optional;  
@Component  
public class ProvinceFormatter implements Formatter<Province> {  
 private IProvinceService provinceService;  
  
 @Autowired  
 public ProvinceFormatter(IProvinceService provinceService) {  
 this.provinceService = provinceService;  
 }  
  
 @Override  
 public Province parse(String text, Locale locale) throws ParseException {  
 Optional<Province> provinceOptional = provinceService.findById(Long.*parseLong*(text));  
 return provinceOptional.orElse(null);  
 }  
  
 @Override  
 public String print(Province object, Locale locale) {  
 return "[" + object.getId() + ", " +object.getName() + "]";  
 }  
}

* Class này tạo ra nhằm mục đích ép kiểu id Province từ text về Long (Bởi vì id Province select từ views là kiểu dữ liệu String)
* Thêm đoạn code sau vào AppConfig => Để tiêm được các tầng service vào controller:

@Bean  
ICustomerService customerService(){  
 return new CustomerServiceIMPL();  
}  
@Bean  
IProvinceService provinceService(){  
 return new ProvinceServiceIMPL();  
}

* Thêm tiếp đoạn code sau vào AppConfig để khởi tạo @Bean tiêm sự phụ thuộc thông qua Formatter cho lớp ProvinceService

@Override  
public void addFormatters(FormatterRegistry registry) {  
 registry.addFormatter(new ProvinceFormatter(applicationContext.getBean(ProvinceServiceIMPL.class)));  
}

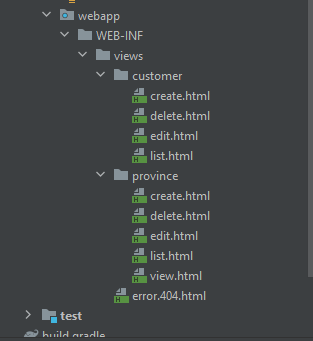
* Tạo package: **controller**
* Tạo class: **ProvinceController** trong package này:

package rikkei.academy.controller;  
  
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.PathVariable;  
import org.springframework.web.bind.annotation.PostMapping;  
import org.springframework.web.servlet.ModelAndView;  
import rikkei.academy.model.Customer;  
import rikkei.academy.model.Province;  
import rikkei.academy.service.customer.ICustomerService;  
import rikkei.academy.service.province.IProvinceService;  
  
import java.util.Optional;  
  
@Controller  
public class ProvinceController {  
 @Autowired  
 private IProvinceService provinceService;  
  
 @Autowired  
 private ICustomerService customerService;  
  
 @GetMapping("/provinces")  
 public ModelAndView listProvinces() {  
 Iterable<Province> provinces = provinceService.findAll();  
 ModelAndView modelAndView = new ModelAndView("/province/list");  
 modelAndView.addObject("provinces", provinces);  
 return modelAndView;  
 }  
  
 @GetMapping("/create-province")  
 public ModelAndView showCreateForm() {  
 ModelAndView modelAndView = new ModelAndView("/province/create");  
 modelAndView.addObject("province", new Province());  
 return modelAndView;  
 }  
  
 @PostMapping("/create-province")  
 public ModelAndView saveProvince(@ModelAttribute("province") Province province) {  
 provinceService.save(province);  
  
 ModelAndView modelAndView = new ModelAndView("/province/create");  
 modelAndView.addObject("province", new Province());  
 modelAndView.addObject("message", "New province created successfully");  
 return modelAndView;  
 }  
  
 @GetMapping("/edit-province/{id}")  
 public ModelAndView showEditForm(@PathVariable Long id) {  
 Optional<Province> province = provinceService.findById(id);  
 if (province.isPresent()) {  
 ModelAndView modelAndView = new ModelAndView("/province/edit");  
 modelAndView.addObject("province", province.get());  
 return modelAndView;  
  
 } else {  
 ModelAndView modelAndView = new ModelAndView("/error.404");  
 return modelAndView;  
 }  
 }  
  
 @PostMapping("/edit-province")  
 public ModelAndView updateProvince(@ModelAttribute("province") Province province) {  
 provinceService.save(province);  
 ModelAndView modelAndView = new ModelAndView("/province/edit");  
 modelAndView.addObject("province", province);  
 modelAndView.addObject("message", "Province updated successfully");  
 return modelAndView;  
 }  
  
 @GetMapping("/delete-province/{id}")  
 public ModelAndView showDeleteForm(@PathVariable Long id) {  
 Optional<Province> province = provinceService.findById(id);  
 if (province.isPresent()) {  
 ModelAndView modelAndView = new ModelAndView("/province/delete");  
 modelAndView.addObject("province", province.get());  
 return modelAndView;  
  
 } else {  
 ModelAndView modelAndView = new ModelAndView("/error.404");  
 return modelAndView;  
 }  
 }  
  
 @PostMapping("/delete-province")  
 public String deleteProvince(@ModelAttribute("province") Province province) {  
 provinceService.remove(province.getId());  
 return "redirect:provinces";  
 }  
  
 @GetMapping("/view-province/{id}")  
 public ModelAndView viewProvince(@PathVariable("id") Long id){  
 Optional<Province> provinceOptional = provinceService.findById(id);  
 if(!provinceOptional.isPresent()){  
 return new ModelAndView("/error.404");  
 }  
  
 Iterable<Customer> customers = customerService.findAllByProvince(provinceOptional.get());  
  
 ModelAndView modelAndView = new ModelAndView("/province/view");  
 modelAndView.addObject("province", provinceOptional.get());  
 modelAndView.addObject("customers", customers);  
 return modelAndView;  
 }  
}

* Tạo class: **CustomerController** trong package: controller

package rikkei.academy.controller;  
  
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.PathVariable;  
import org.springframework.web.bind.annotation.PostMapping;  
import org.springframework.web.servlet.ModelAndView;  
import rikkei.academy.model.Customer;  
import rikkei.academy.model.Province;  
import rikkei.academy.service.customer.ICustomerService;  
import rikkei.academy.service.province.IProvinceService;  
  
import java.util.Optional;  
  
@Controller  
public class CustomerController {  
 @Autowired  
 private ICustomerService customerService;  
  
 @Autowired  
 private IProvinceService provinceService;  
  
 @ModelAttribute("provinces")  
 public Iterable<Province> provinces(){  
 return provinceService.findAll();  
 }  
  
  
 @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;  
 }  
  
 @GetMapping("/customers")  
 public ModelAndView listCustomers() {  
 Iterable<Customer> customers = customerService.findAll();  
 ModelAndView modelAndView = new ModelAndView("/customer/list");  
 modelAndView.addObject("customers", customers);  
 return modelAndView;  
 }  
  
 @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 {  
 ModelAndView modelAndView = new ModelAndView("/error.404");  
 return modelAndView;  
 }  
 }  
  
 @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;  
 }  
  
 @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 {  
 ModelAndView modelAndView = new ModelAndView("/error.404");  
 return modelAndView;  
 }  
 }  
  
 @PostMapping("/delete-customer")  
 public String deleteCustomer(@ModelAttribute("customer") Customer customer) {  
 customerService.remove(customer.getId());  
 return "redirect:customers";  
 }  
}

* Add TomCat tạo database: **bai7\_th1\_province** trong MYSQL (Thay đổi các mật khẩu MySQL tương ứng của các bạn trong file **AppConfig**) Chạy chương trình để tạo table.
* Tạo thư mục: webapp/WEB-INF/views (Tầng views sẽ nhìn như sau)



* Tạo thư mục **province**:
* Tạo **create.html** trong **province:**

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

* Tạo thư mục: **list.html**

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

* Tạo thư mục : **delete.html** trong province

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

* Tạo **edit.html** trong **province**

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

* Tạo **view.html** trong **province**

<!DOCTYPE html>  
<html lang="en" xmlns:th="http://www.thymeleaf.org">  
<head>  
 <meta charset="UTF-8">  
 <title>View province</title>  
</head>  
<body>  
<h1>View province: <span th:text="${province.name}"></span></h1>  
<a href="/provinces">Province list</a>  
<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>

* Tạo tiếp thư mục **customer**
* Tạo file: **create.html** trong **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>Province:</td>  
 <td>  
 <select th:field="\*{province}">  
 <option th:each="p : ${provinces}" th:value="${p.id}" th:text="${p.name}"></option>  
 </select>  
 </td>  
 </tr>  
 <tr>  
 <td></td>  
 <td><input type="submit" value="Create customer"></td>  
 </tr>  
 </table>  
</form>  
</body>  
</html>

* Tạo file: **delete.html** trong **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>

* Tạo file: **edit.html** trong **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>Province:</td>  
 <td>  
 <select name="province">  
 <option th:each="p : ${provinces}"  
 th:value="${p.id}"  
 th:text="${p.name}"  
 th:selected="(${customer.province}) and (${p.id} == \*{province.id})"></option>  
 </select>  
 </td>  
 </tr>  
 <tr>  
 <td></td>  
 <td><input type="submit" value="Update customer"></td>  
 </tr>  
 </table>  
</form>  
</body>  
</html>

* Tạo tiếp file: **list.html** trong **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>  
 <th>Province</th>  
 <th>Edit</th>  
 <th>Delete</th>  
 </tr>  
 <th:block th:each="customer : ${customers}">  
 <tr>  
 <td th:text="${customer.firstName}"></td>  
 <td th:text="${customer.lastName}"></td>  
 <td th:if="${customer.province}" th:text="${customer.province.name}"></td>  
 <td th:unless="${customer.province}"></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>

* Tạo tiếp file: **error.404.html** trong thư mục **views** để check các trường hợp ngoại lệ lỗi

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

Chạy lại chương trình và test các kết quả =>Lưu ý: Nhớ rằng ở đây Province có quan hệ phụ thuộc 1-N với Customer nên phải tạo dữ liệu cho bảng Province trước.

https://github.com/nguyendongminhtri/MD1-Bai7-TH1-Province-Customer.git

### Hướng dẫn nộp bài:

Up code lên github.

Paste link github vào phần nộp bài.