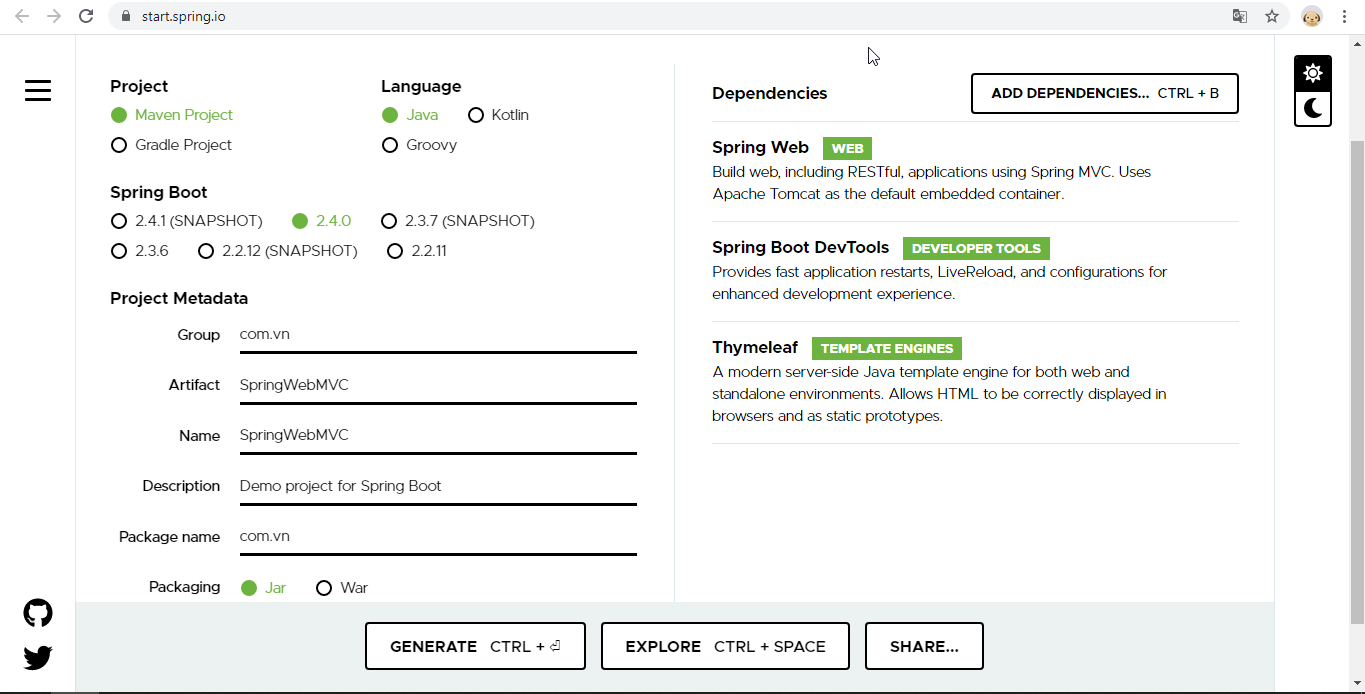
# Tạo project và import

## Step1: Thực hiện generate project và import vào IDE

<https://start.spring.io/>

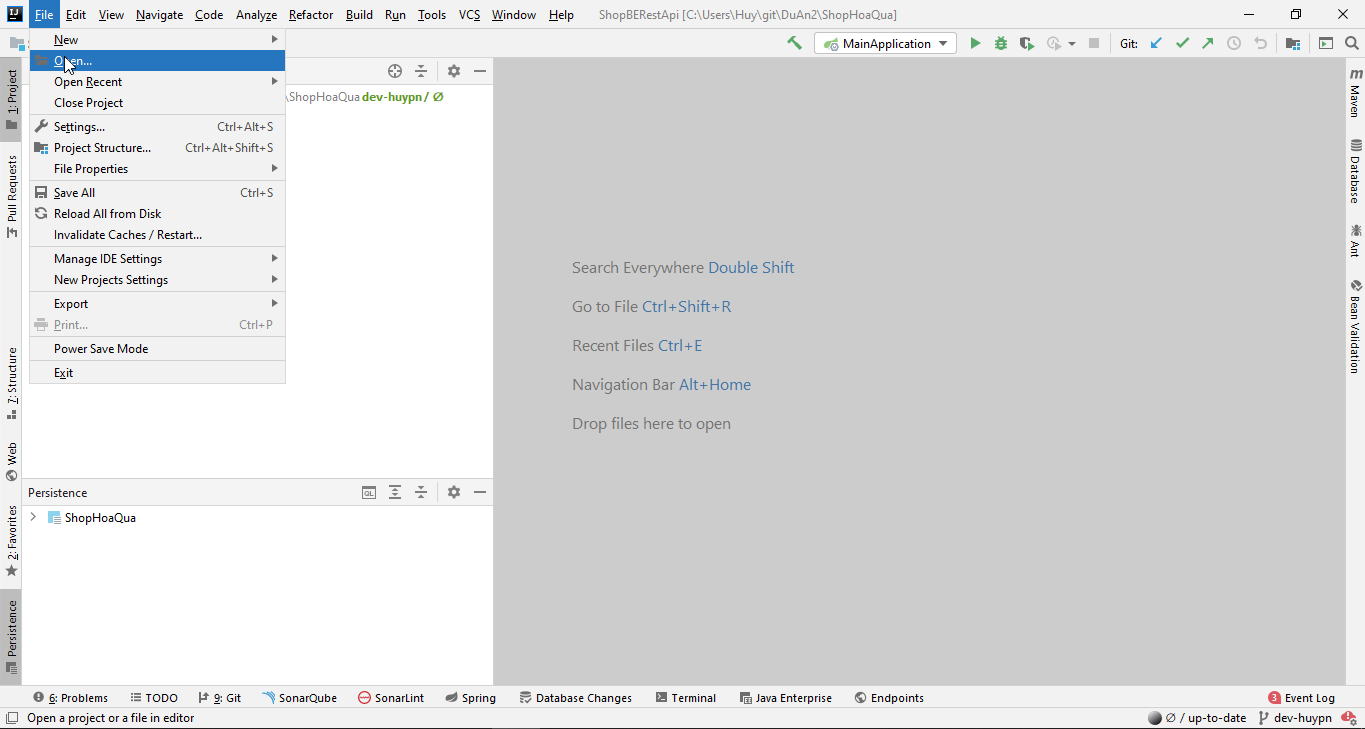
Trong giao diện mô tả rất rõ các thư viện

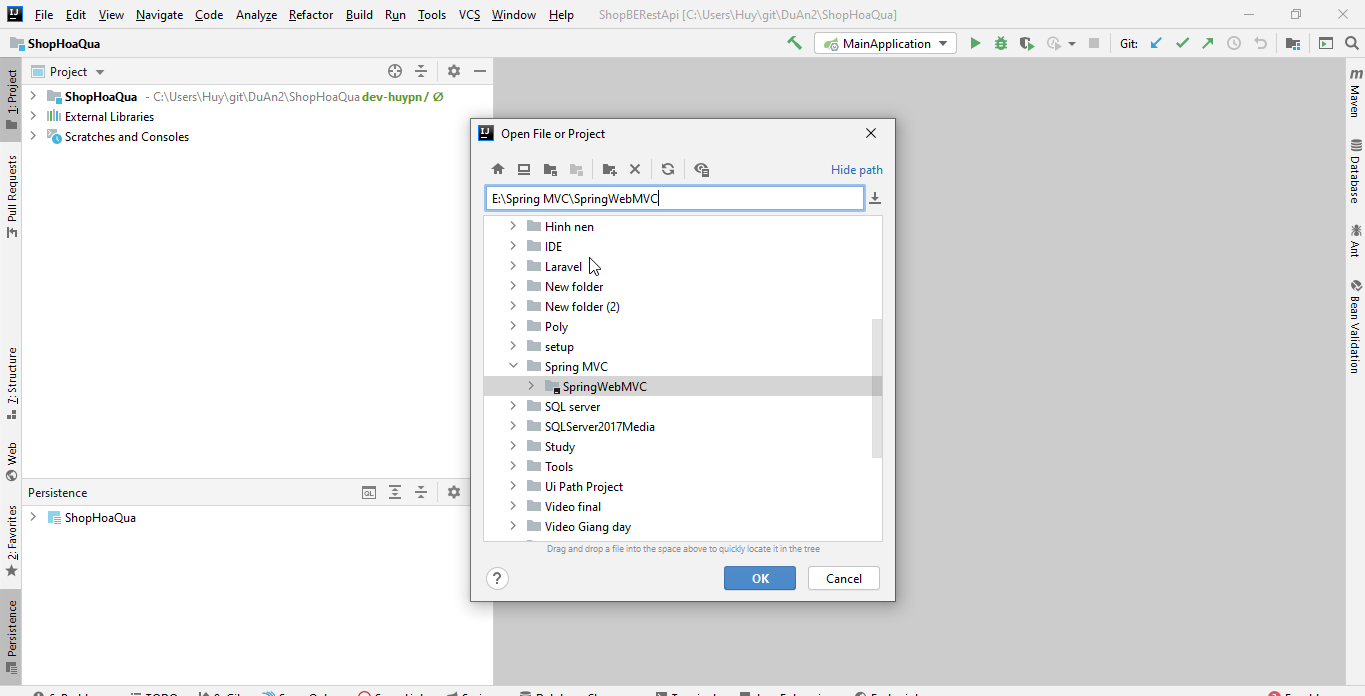
* Spring web cho phát triển web
* Spring boot devetool cho phát triên tool
* Thymeleaf là template engine

Các bước tiếp theo làm là import project cho IDE

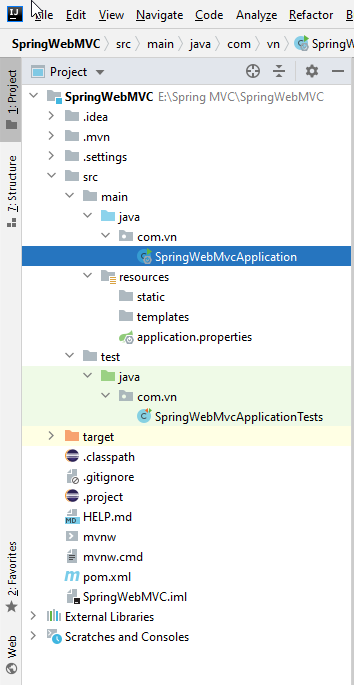
1. Nhấn generate để tải về file zip
2. Giair nén file zip để có project
3. Import project vào IDE: Eclipse IntelliJ nếu muốn

Step2: Import project và update thư viện



Chọn nơi chứa project và import project Chờ đợi việc import thư viện thành công chúng ta có thể xem trước các bước tiếp theo

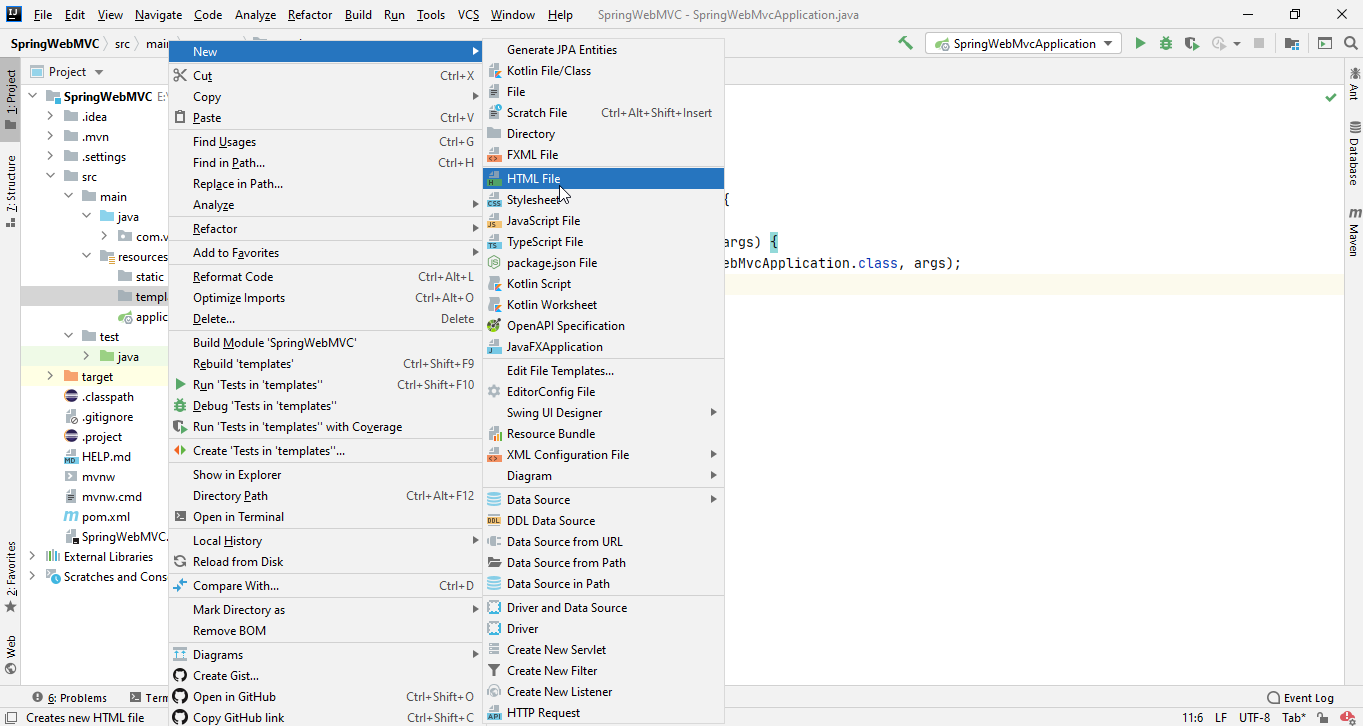
## Step2: Tìm hiểu cấu trúc project



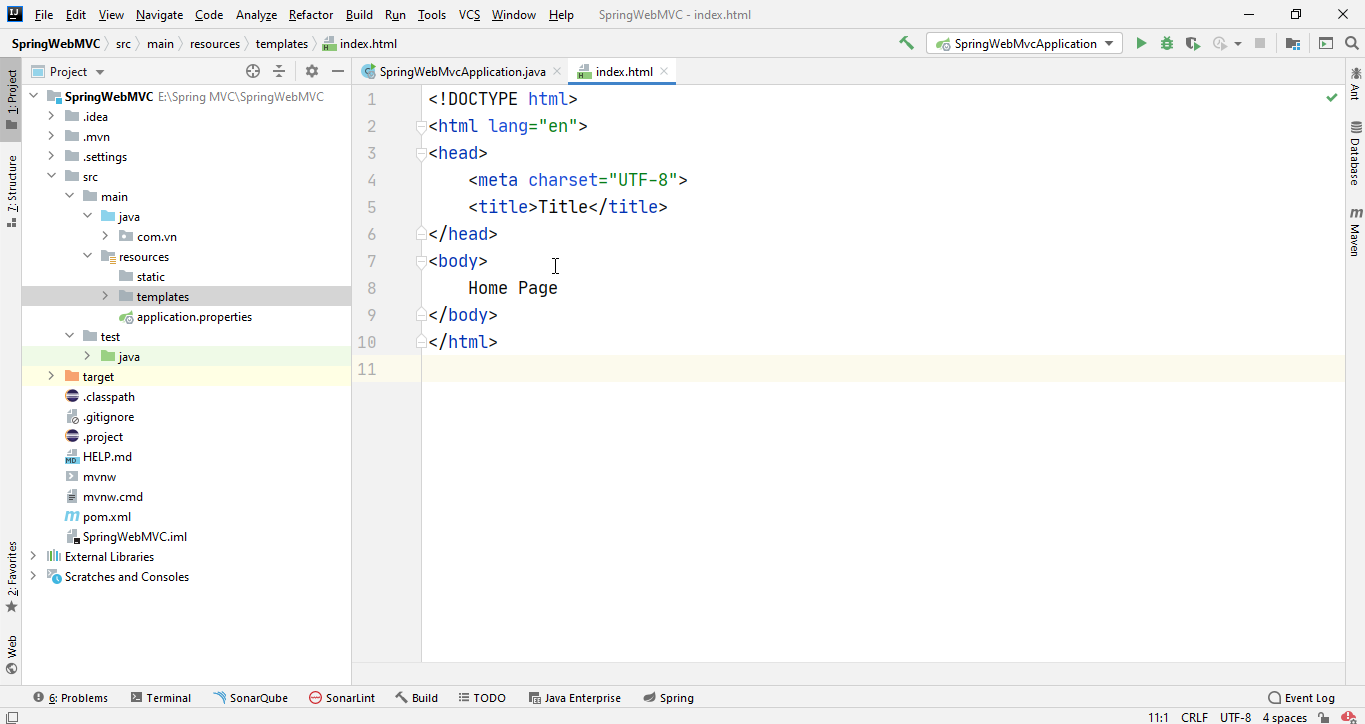
* SpringWebMvcApplication file application chạy ứng dụng main Spring boot
* SpringWebMvcApplicationTests cho việc chạy test ứng dụng
* Src/main/resource chứa các nguồn của chương trình
  1. Static chứa các file tĩnh js, css, image
  2. Template chứa các file html
  3. Pom.xml là file mô tả các thư viện

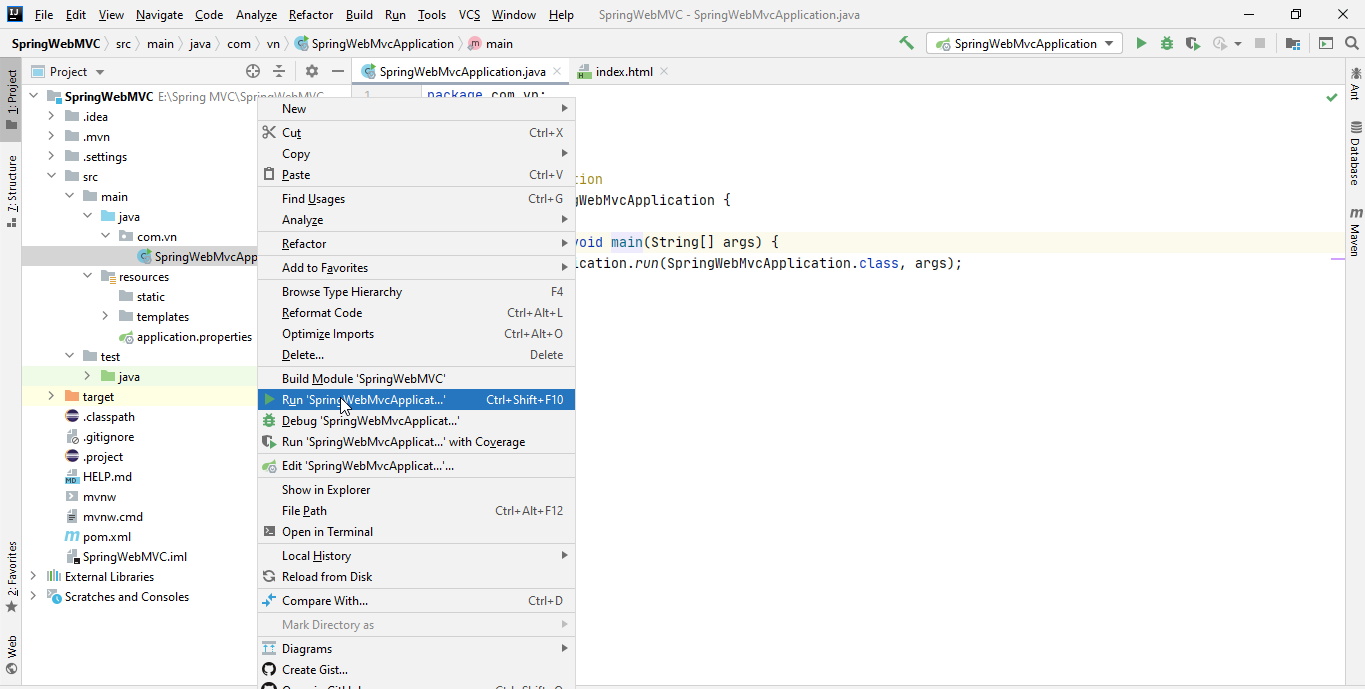
# Tạo Hello World với Thymeleaf

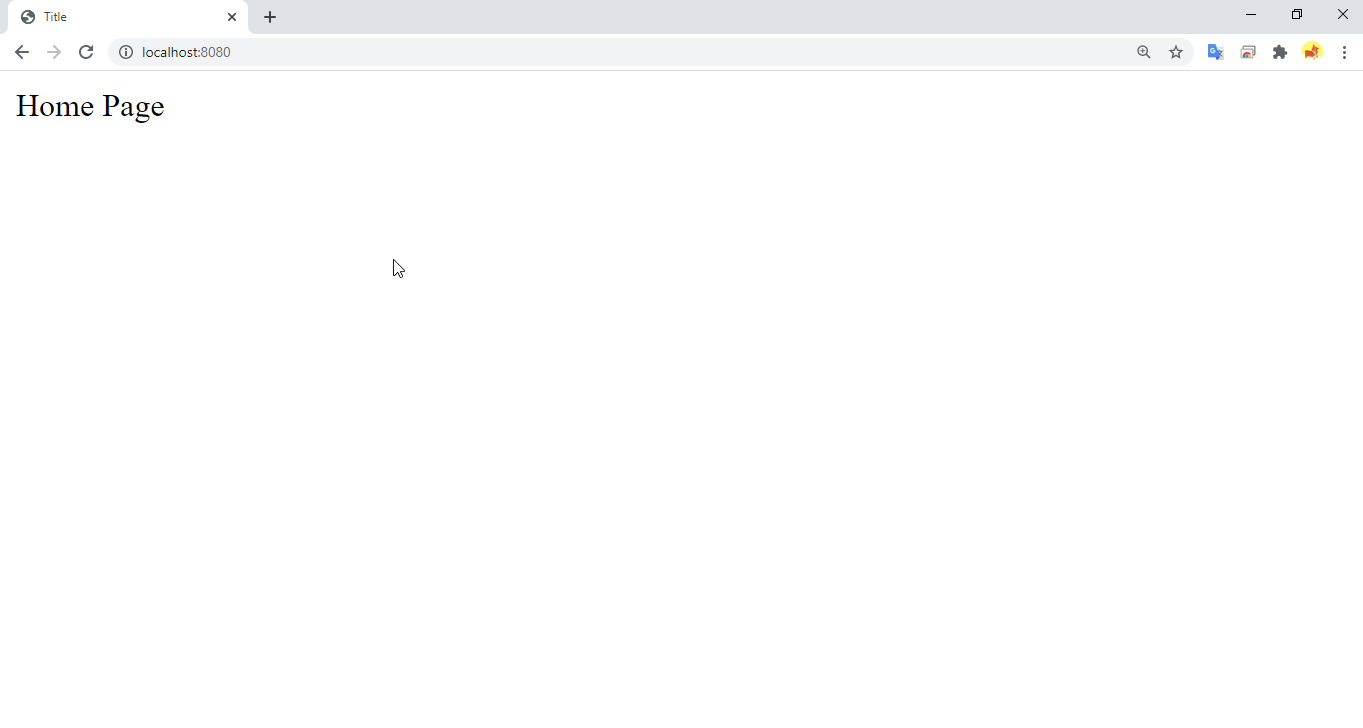
## Step3: Tạo file index.html trong template



Tạo file html index.html để chạy thử ứng dụng





Hình ảnh sau khi chạy ứng dụng

Gõ <http://localhost:8080/> để có thể xem kết quả

# Cấu hình đầy đủ cho Template Engine Thymeleaf

## Step4: Tìm hiểu Thymeleaf

* Hãy đoc hiểu Thymeleaf là gì <https://www.thymeleaf.org/doc/tutorials/3.0/usingthymeleaf.html#introducing-thymeleaf>

Việc biểu thức chính quy với mỗi template engine có đặc tính cách viết riêng

* Xem cách thức syntax (cú pháp) của nó

<https://www.thymeleaf.org/doc/tutorials/3.0/usingthymeleaf.html#standard-expression-syntax>

|  |  |  |
| --- | --- | --- |
| Variable Expressions | **${...}** |  |
| Selection Variable Expressions | **\*{...}** |  |
| Message Expressions | **#{...}** |  |
| Link URL Expressions: | **@{...}** |  |
| Fragment Expressions | **~{...}** |  |
|  |  |  |

Các toán tử

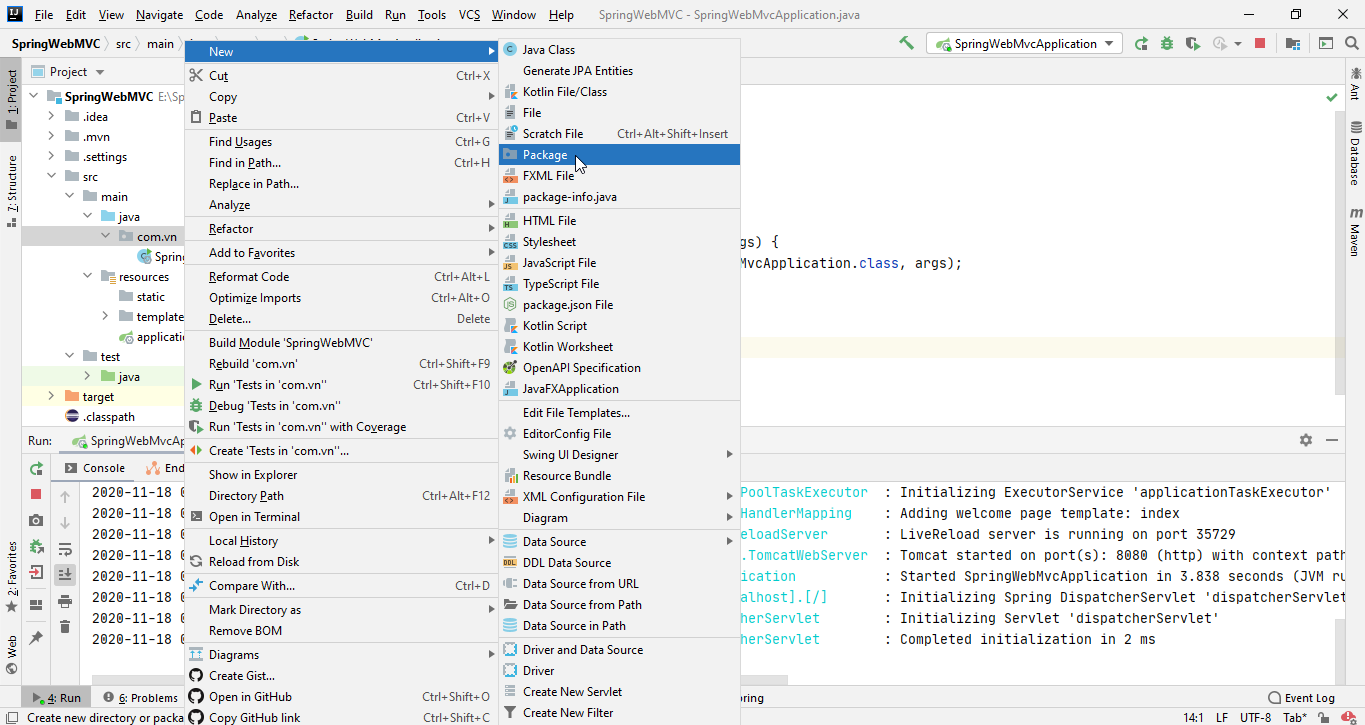
|  |  |  |
| --- | --- | --- |
| If-then | **(if) ? (then)** |  |
| If-then-else | **(if) ? (then) : (else)** |  |
| Default | **(value) ?: (defaultvalue)** |  |
|  |  |  |
|  |  |  |

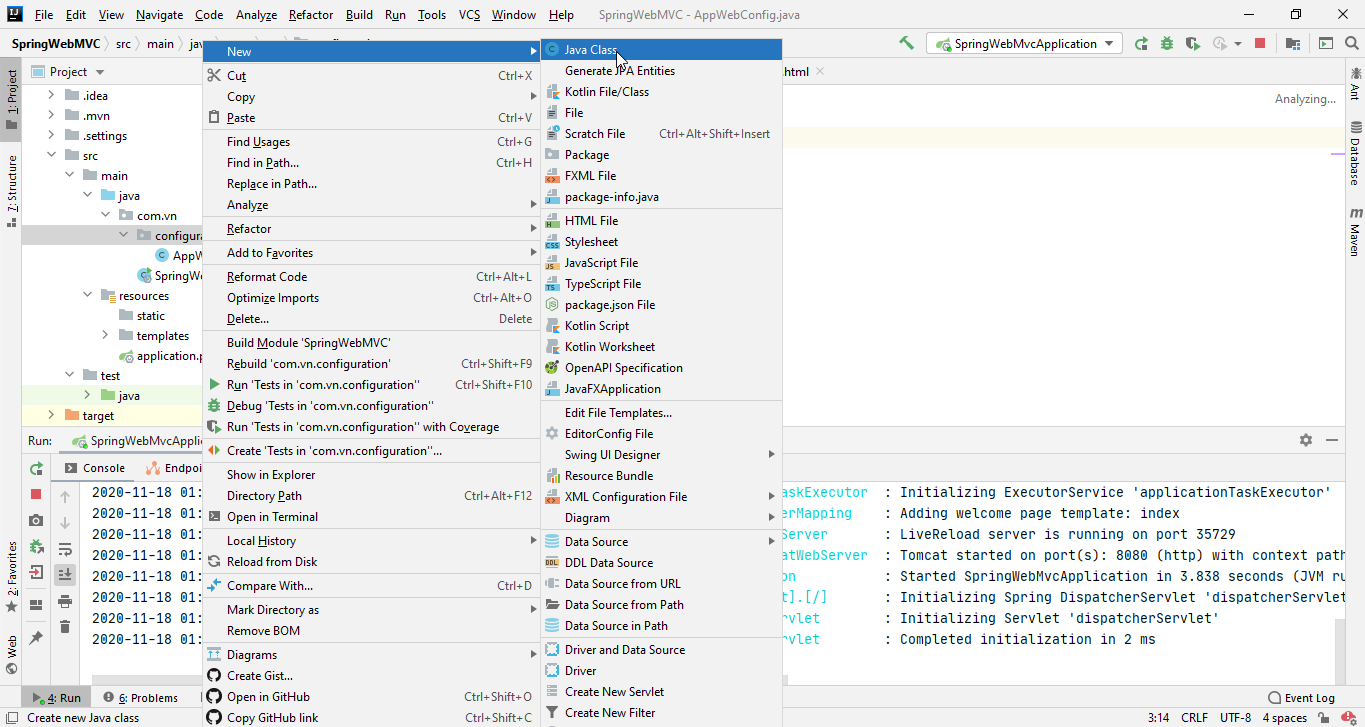
# Tích hợp Thymeleaf với Spring

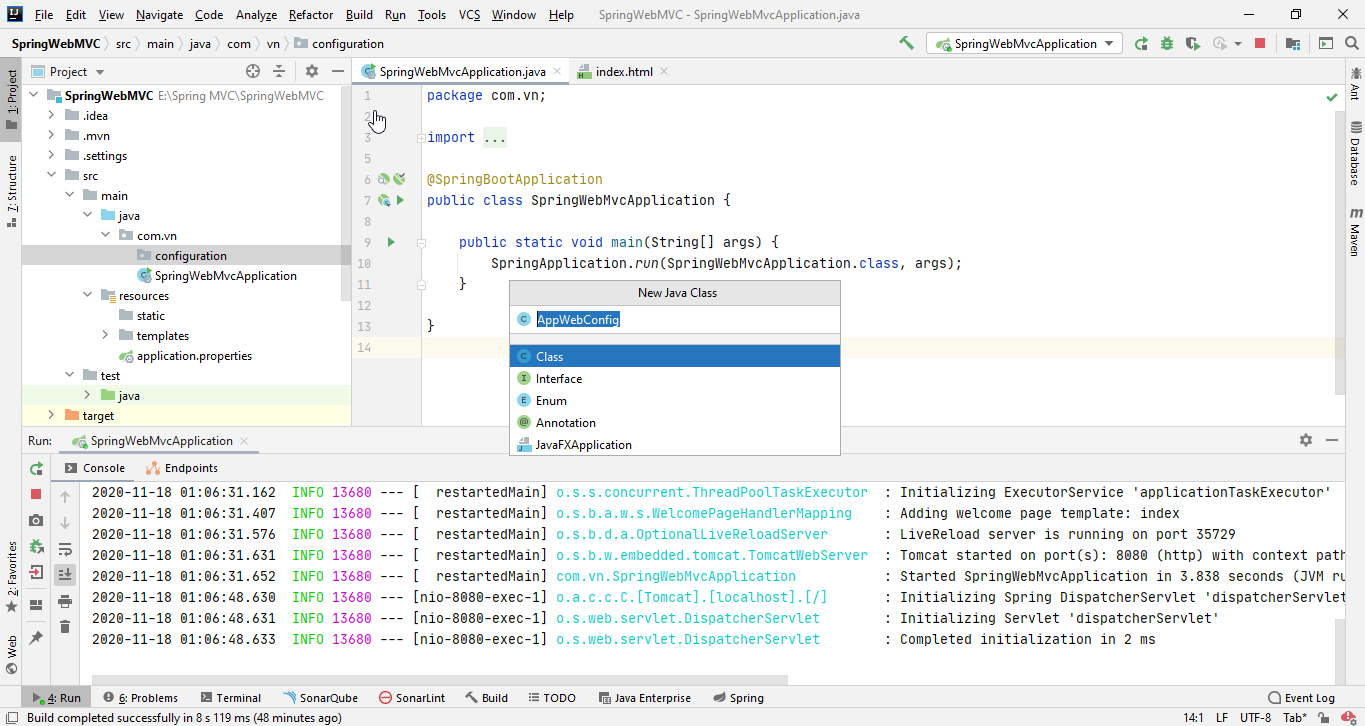
Tuy mặc định có thê sử dụng Thymeleaf ngay lập tức nhưng nên custom để có thể tùy biến

Tham khảo: <https://www.thymeleaf.org/doc/tutorials/3.0/thymeleafspring.html>

## Step5: Tạo config cấu hình



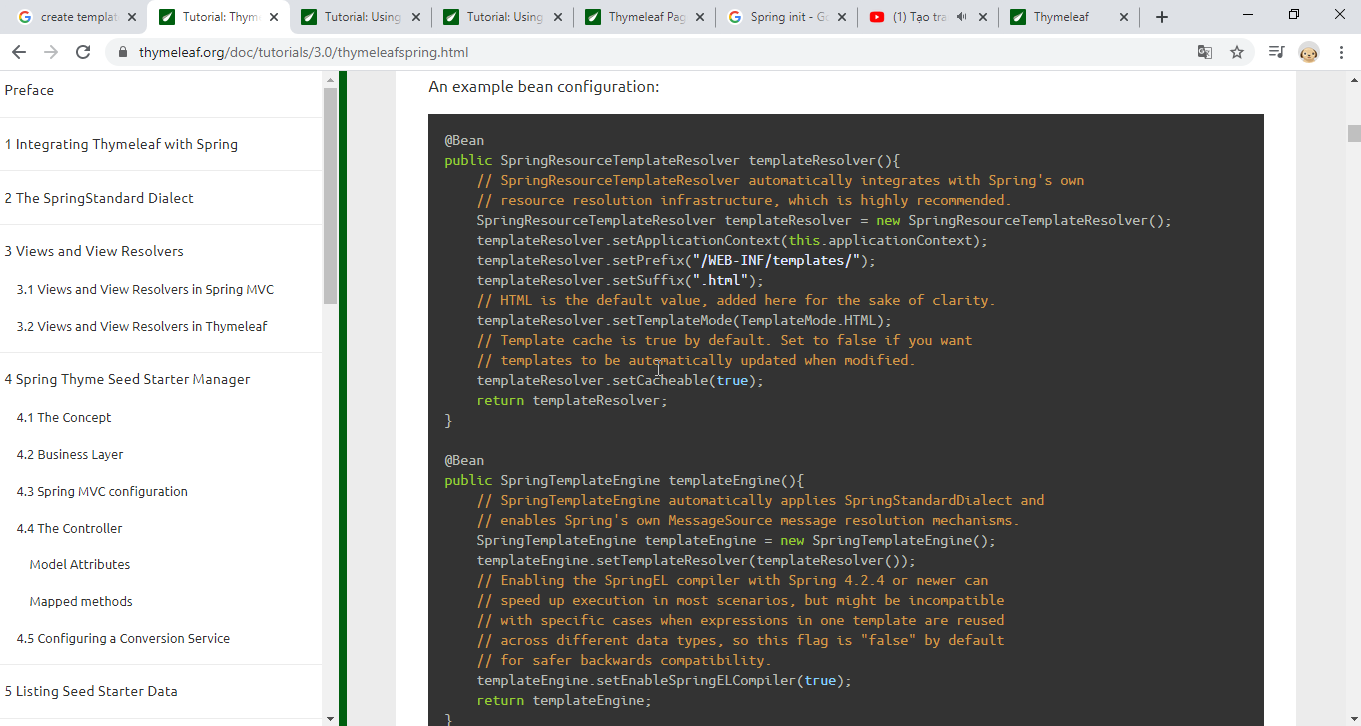




Tạo class config

## Step6: Cấu hình config từ trang chủ Thymeleaf

https://www.thymeleaf.org/doc/tutorials/3.0/thymeleafspring.html

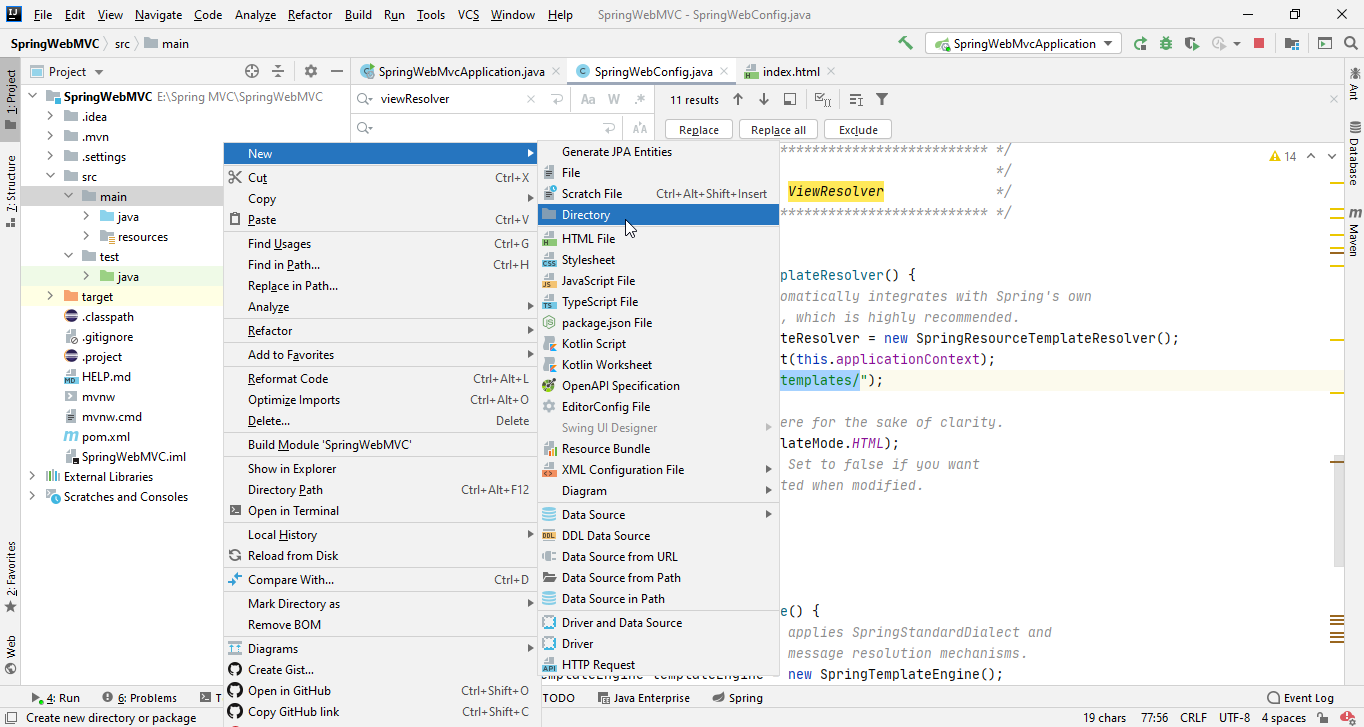


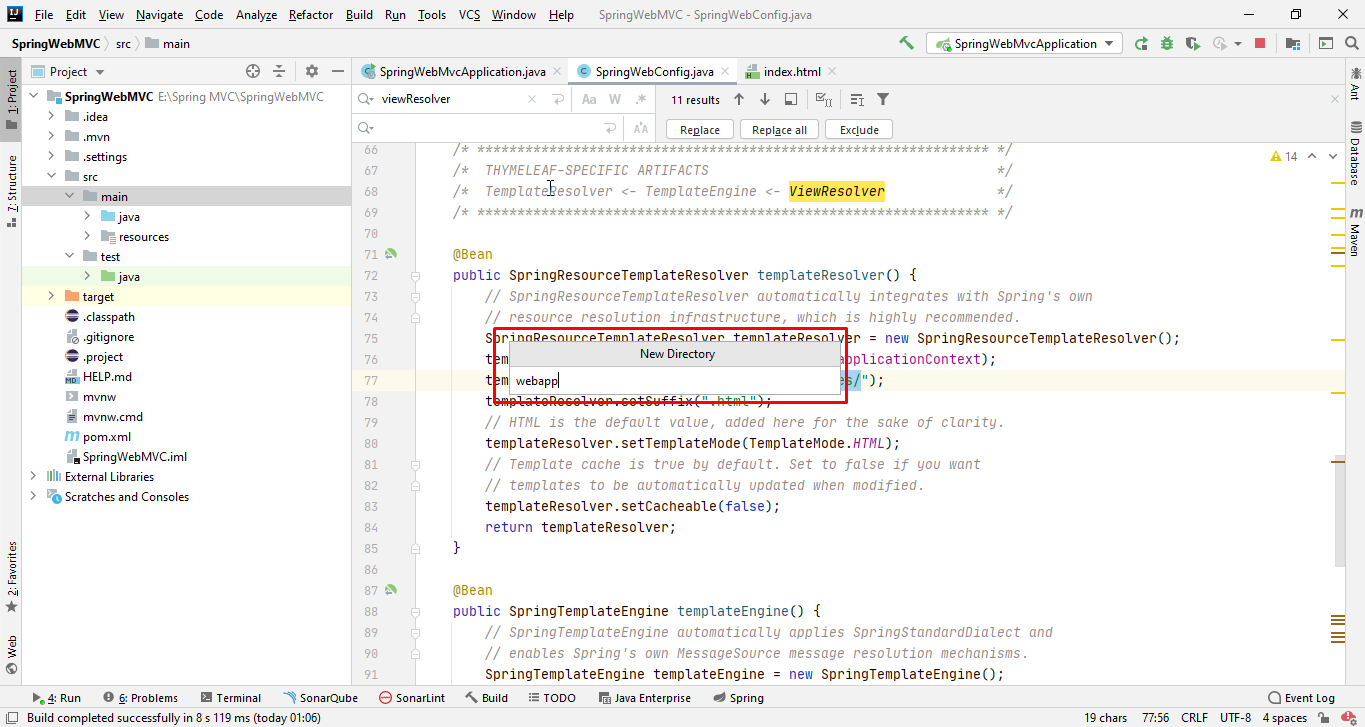
Cấu hình các bean

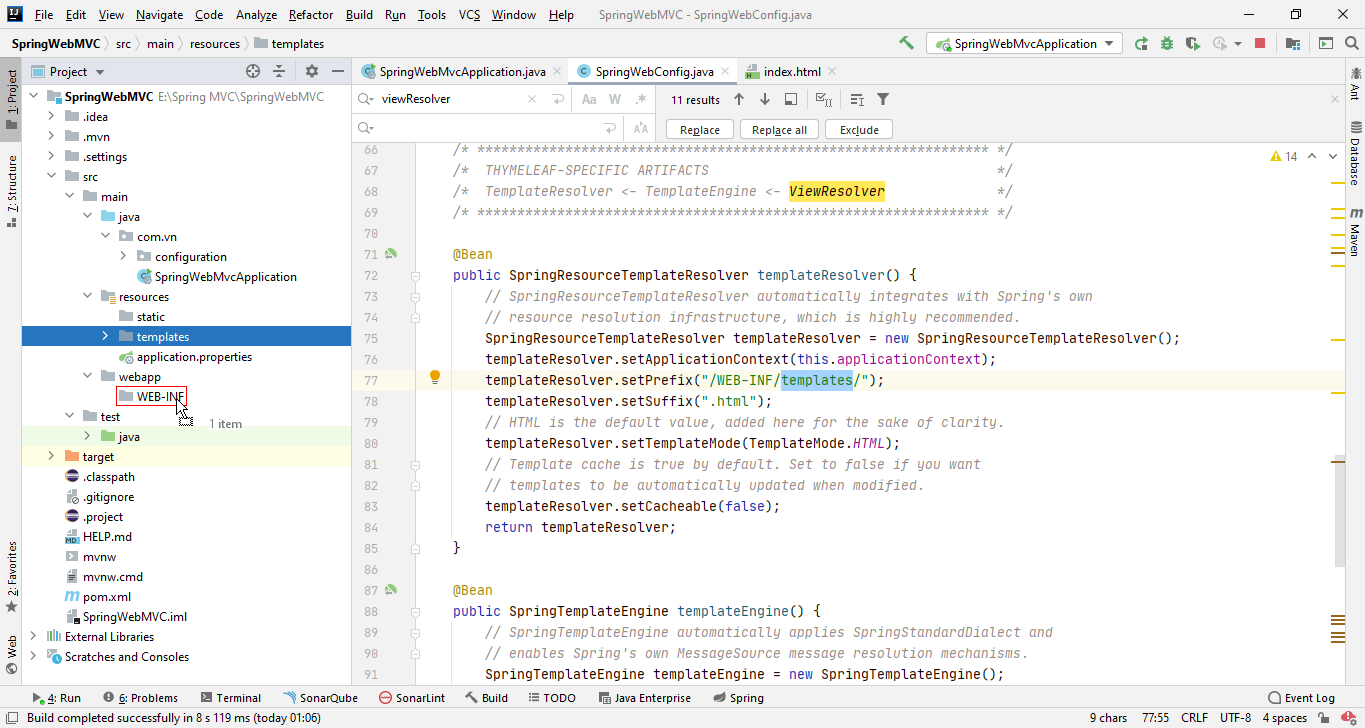
|  |
| --- |
| package com.vn.configuration;  import org.springframework.beans.BeansException; import org.springframework.beans.factory.annotation.Autowired; import org.springframework.context.ApplicationContext; import org.springframework.context.ApplicationContextAware; import org.springframework.context.annotation.Bean; import org.springframework.context.annotation.ComponentScan; import org.springframework.context.annotation.Configuration; import org.springframework.context.annotation.Scope; import org.springframework.context.support.ResourceBundleMessageSource; import org.springframework.format.FormatterRegistry; import org.springframework.web.servlet.config.annotation.EnableWebMvc; import org.springframework.web.servlet.config.annotation.ResourceHandlerRegistry; import org.springframework.web.servlet.config.annotation.WebMvcConfigurer; import org.springframework.web.servlet.config.annotation.WebMvcConfigurerAdapter; import org.thymeleaf.spring5.SpringTemplateEngine; import org.thymeleaf.spring5.templateresolver.SpringResourceTemplateResolver; import org.thymeleaf.spring5.view.ThymeleafView; import org.thymeleaf.spring5.view.ThymeleafViewResolver; import org.thymeleaf.templatemode.TemplateMode;  import java.util.Collections;  @Configuration public class SpringWebConfig  implements WebMvcConfigurer, ApplicationContextAware {   private ApplicationContext applicationContext;    public SpringWebConfig() {  super();  }    public void setApplicationContext(final ApplicationContext applicationContext)  throws BeansException {  this.applicationContext = applicationContext;  }     */\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/  /\* GENERAL CONFIGURATION ARTIFACTS \*/  /\* Static Resources, i18n Messages, Formatters (Conversion Service) \*/  /\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/* @Override  public void addResourceHandlers(final ResourceHandlerRegistry registry) {  registry.addResourceHandler("/images/\*\*").addResourceLocations("/images/");  registry.addResourceHandler("/css/\*\*").addResourceLocations("/css/");  registry.addResourceHandler("/js/\*\*").addResourceLocations("/js/");  registry.addResourceHandler("/static/\*\*").addResourceLocations("/static/");   }   @Bean  public ResourceBundleMessageSource messageSource() {  ResourceBundleMessageSource messageSource = new ResourceBundleMessageSource();  messageSource.setBasename("Messages");  return messageSource;  }   */\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/  /\* THYMELEAF-SPECIFIC ARTIFACTS \*/  /\* TemplateResolver <- TemplateEngine <- ViewResolver \*/  /\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/* @Bean  public SpringResourceTemplateResolver templateResolver() {  *// SpringResourceTemplateResolver automatically integrates with Spring's own  // resource resolution infrastructure, which is highly recommended.* SpringResourceTemplateResolver templateResolver = new SpringResourceTemplateResolver();  templateResolver.setApplicationContext(this.applicationContext);  templateResolver.setPrefix("/WEB-INF/templates/");  templateResolver.setSuffix(".html");  *// HTML is the default value, added here for the sake of clarity.* templateResolver.setTemplateMode(TemplateMode.*HTML*);  *// Template cache is true by default. Set to false if you want  // templates to be automatically updated when modified.* templateResolver.setCacheable(false);  return templateResolver;  }   @Bean  public SpringTemplateEngine templateEngine() {  *// SpringTemplateEngine automatically applies SpringStandardDialect and  // enables Spring's own MessageSource message resolution mechanisms.* SpringTemplateEngine templateEngine = new SpringTemplateEngine();  templateEngine.setTemplateResolver(templateResolver());  *// Enabling the SpringEL compiler with Spring 4.2.4 or newer can  // speed up execution in most scenarios, but might be incompatible  // with specific cases when expressions in one template are reused  // across different data types, so this flag is "false" by default  // for safer backwards compatibility.* templateEngine.setEnableSpringELCompiler(true);  return templateEngine;  }   @Bean  public ThymeleafViewResolver viewResolver() {  ThymeleafViewResolver viewResolver = new ThymeleafViewResolver();  viewResolver.setTemplateEngine(templateEngine());  *// NOTE 'order' and 'viewNames' are optional* viewResolver.setOrder(1);  viewResolver.setViewNames(new String[]{".html", ".xhtml"});  return viewResolver;  }  } |

* Trong quá trình development có thuộc tính templateResolver.setCacheable(true); sẽ khiến việc giao diện không update khi html thay đổi, nên đặt là templateResolver.setCacheable(false);
* Trong config mẫu từ trang chủ nên tạo các folder có cấu trúc như config để đọc html: templateResolver.setPrefix("/WEB-INF/templates/");

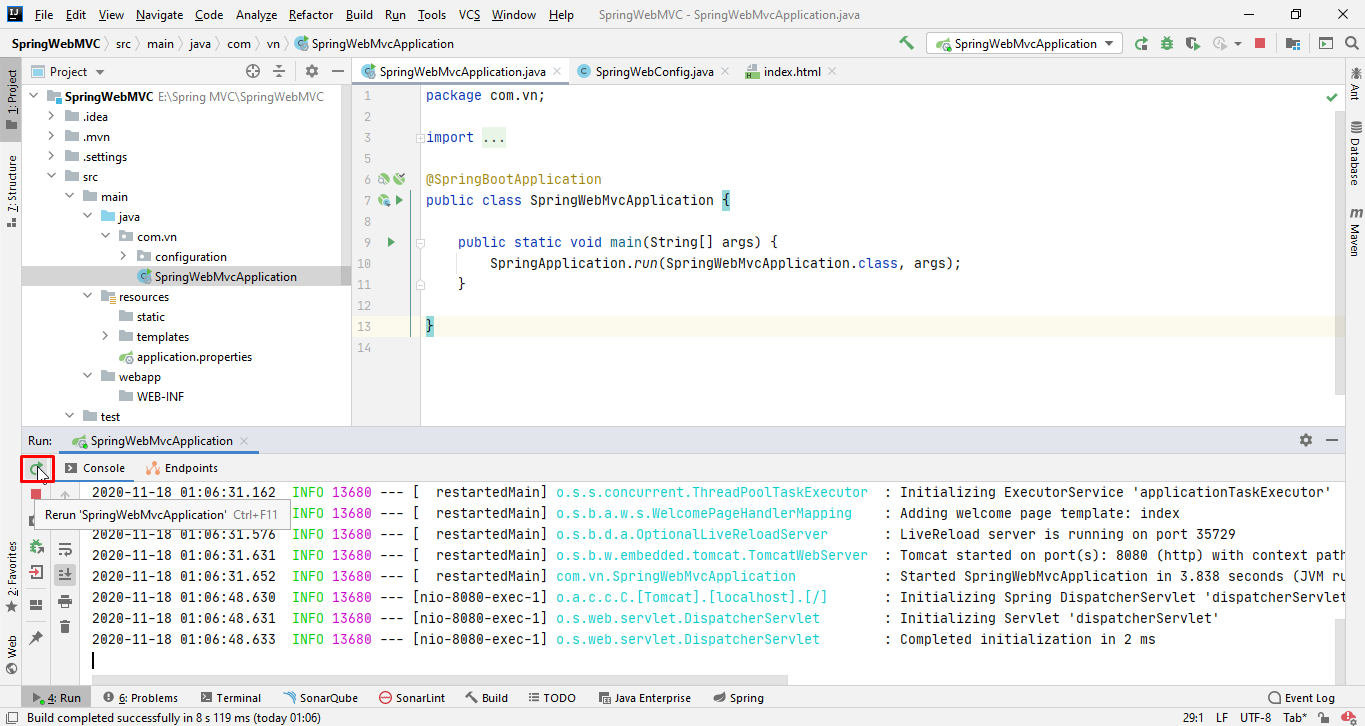
Tạo folder /WEB-INF/templates/



Tương tự tạo folder WEB-INF

Folder template thì “kéo” từ folder static sang như hình

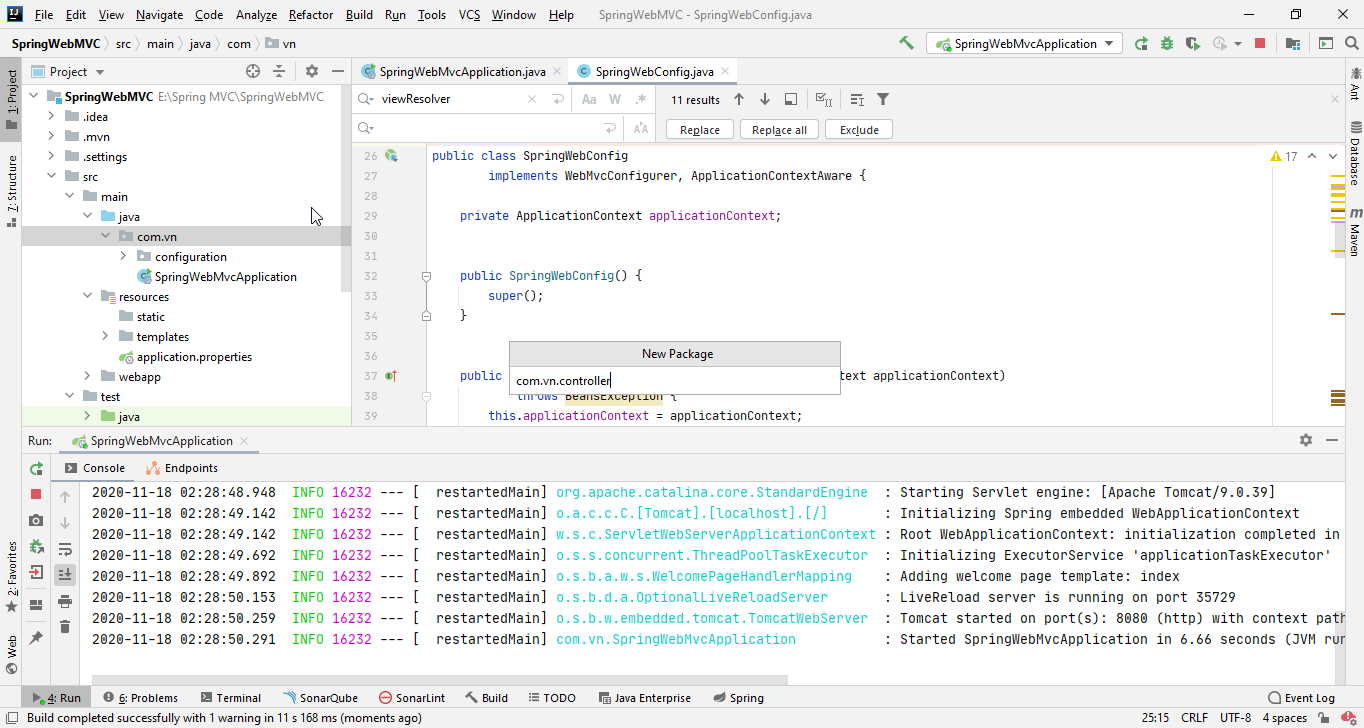
## Step7: Chạy kiểm tra lại việc tích hợp



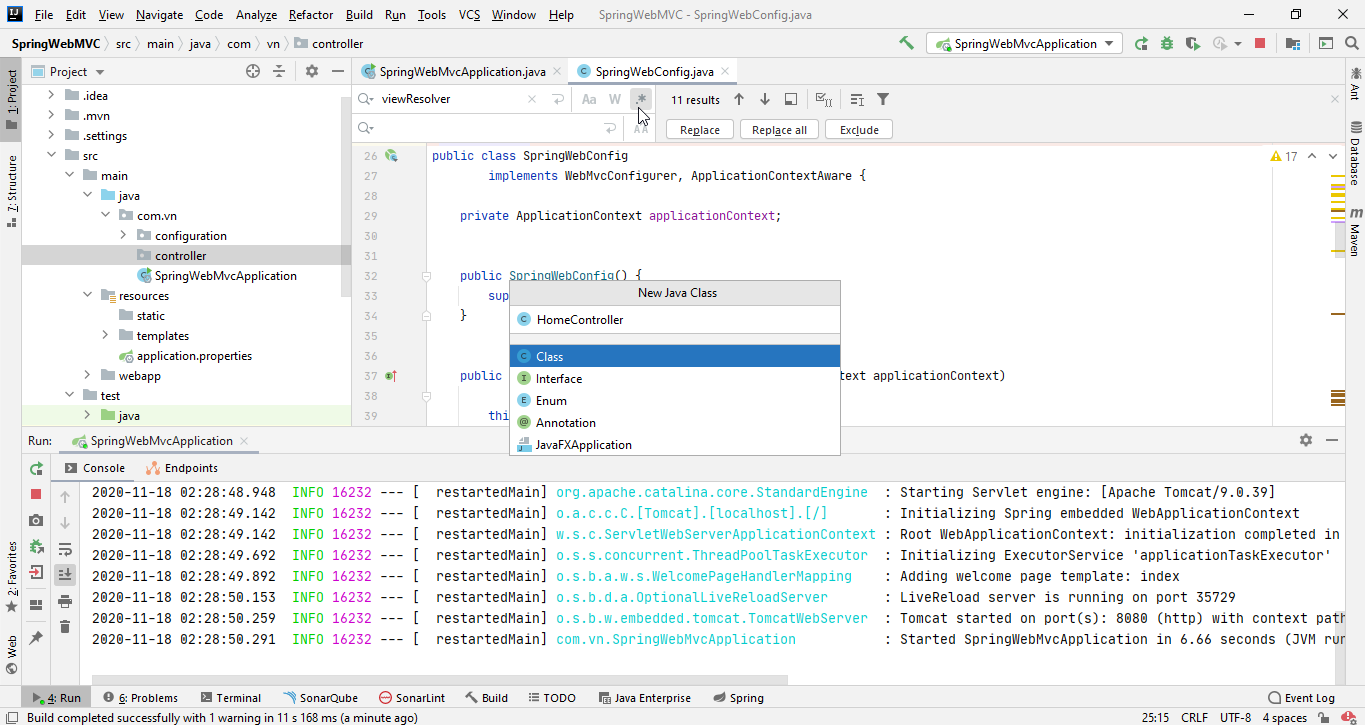
Chương trình chạy <http://localhost:8080/> sẽ không chạy được như mong muốn là rat trang chủ, tạo theo bước sau để hoàn thành

## Step8: Tạo controller home page

Tạo package,class HomeController co trang chủ

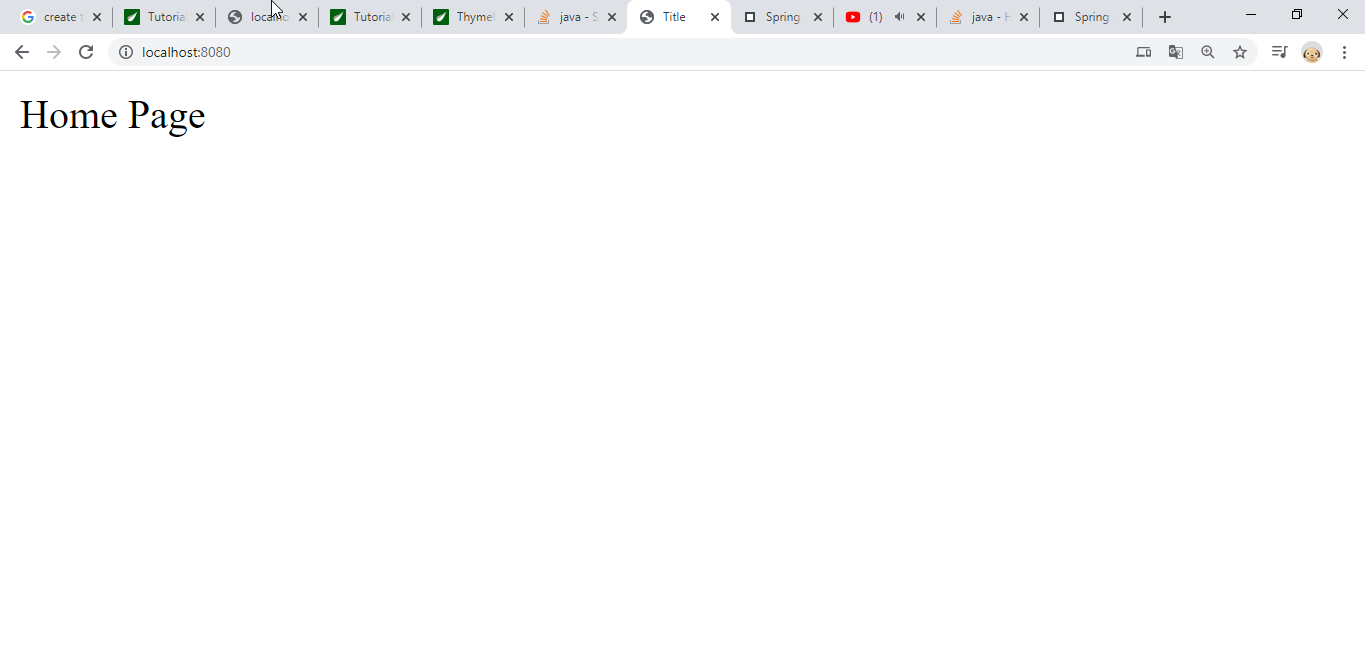


Tạo HomeController



|  |
| --- |
| package com.vn.controller;  import org.springframework.stereotype.Controller; import org.springframework.web.bind.annotation.GetMapping;  @Controller public class HomeController {  @GetMapping(value = {"", "/", "/home"})  public String getHomePage() {  return "index";  } } |

Step9: Chạy kết quả



Kết quả sau khi tạo controller

Nguyên nhân bước trước không chạy được vì sau khi config template thì trang chủ cần phải gọi qua cấu hình chứ không gọi mặc định được nữa

# Tạo template với Thymeleaf