



SPRING BOOT MVC INTRODUCTION

SLIDE 1.1

PHẦN I:

 GIỚI THIỆU SPRING, SPRING MVC VÀ SPRING BOOT

 CÀI ĐẶT MÔI TRƯỜNG PHÁT TRIỂN

 TẠO VÀ KHẢO SÁT DỰ ÁN SPRING BOOT

 QUẢN LÝ THƯ VIỆN PHỤ THUỘC

 XÂY DỰNG TRANG HELLO WORLD

PHẦN II:

 MÔ HÌNH XỬ LÝ REQUEST/RESPONSE

 LÀM VIỆC VỚI HTTP COMPONENTS TRONG SPRING BOOT MVC

 HTTPServletRequest

 HttpServletResponse

 HttpSession

 ServletContext



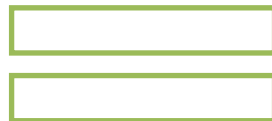


SPRING BOOT INTRODUCTION

- ☐ Spring là framework được giới thiệu 2003 bởi Rod Johnson.
- ☐ Spring là mã nguồn mở sử dụng để phát triển ứng dụng java phổ biến nhất hiện nay.
- ☐ Spring hỗ trợ IoC, DI giúp dễ dàng nâng cấp, mở rộng ứng dụng
- ☐ Spring rất nhẹ trong việc phát triển và triển khai.

- ❑ Spring Boot là một dự án bổ sung của Spring, nhằm đơn giản hóa việc phát triển ứng dụng Spring.
- ❑ Dễ dàng hơn trong việc thiết lập và phát triển ứng dụng.
- ❑ Nguyên tắc “Cấu hình mặc định” giảm thiểu viết mã cấu hình
- ❑ Cho phép nhúng Web Server vào ứng dụng.
- ❑ Có thể chạy ứng dụng web bằng cơ chế dòng lệnh hoặc xuất ra file war để triển khai lên Web Server.
- ❑ Dễ dàng tương tác với hệ sinh thái công nghệ của Spring (JDBC, ORM, Security...)

Spring Boot



Spring Framework



Web
Server



XML
Annotation
Configuration



ENVIRONMENT SETUP

- ❑ Có rất nhiều công cụ có thể sử dụng để tạo dự án Spring Boot. Một số công cụ thông dụng như Eclipse, IntelliJ, NetBean... Sau đây là các công cụ được khuyến nghị chính thức trên trang spring.io
- ❑ Spring Tools 4 for Visual Studio Code
 - ❖ <https://marketplace.visualstudio.com/items?itemName=Pivotal.vscode-boot-dev-pack>
- ❑ **Spring Tools 4 for Eclipse**
 - ❖ https://cdn.spring.io/spring-tools/release/STS4/4.27.0.RELEASE/dist/e4.34/spring-tool-suite-4-4.27.0.RELEASE-e4.34.0-win32.win32.x86_64.zip
- ❑ Spring Tools 4 for Theia
 - ❖ <https://github.com/spring-projects/sts4/wiki/Installation#theia>

- ❑ Bước 1: Cài JDK 17+
- ❑ Bước 2: Tải và giải nén
 - ❖ https://cdn.spring.io/spring-tools/release/STS4/4.27.0.RELEASE/dist/e4.34/spring-tool-suite-4-4.27.0.RELEASE-e4.34.0-win32.win32.x86_64.zip
 - ❖ Chú ý: bạn cần tải đúng phiên bản phù hợp với hệ điều hành trên máy tính của bạn. Tải tại đây: <https://spring.io/tools>
- ❑ Bước 3: Chạy SpringToolSuite4.exe
- ❑ Bước 4: Chọn Workspace (thư mục đặt các dự án Spring Boot của bạn)
- ❑ Đóng cửa sổ Welcome bạn sẽ có cửa sổ như slide sau



SPRING STARTER PROJECT

NewAlt+Shift+N >

Open File...

Open Projects from File System...

Recent Files >

Close EditorCtrl+W

Close All EditorsCtrl+Shift+W

SaveCtrl+S

Save As...

Save AllCtrl+Shift+S

Revert

Move...

Rename...F2

RefreshF5

Convert Line Delimiters To >

Print...Ctrl+P

Import...

Export...

PropertiesAlt+Enter

Switch Workspace >

Restart

Exit

Java Project

Spring Starter Project

Import Spring Getting Started Content

Project...

Package

Class

Interface

Enum

Record

Annotation

Source Folder

Java Working Set

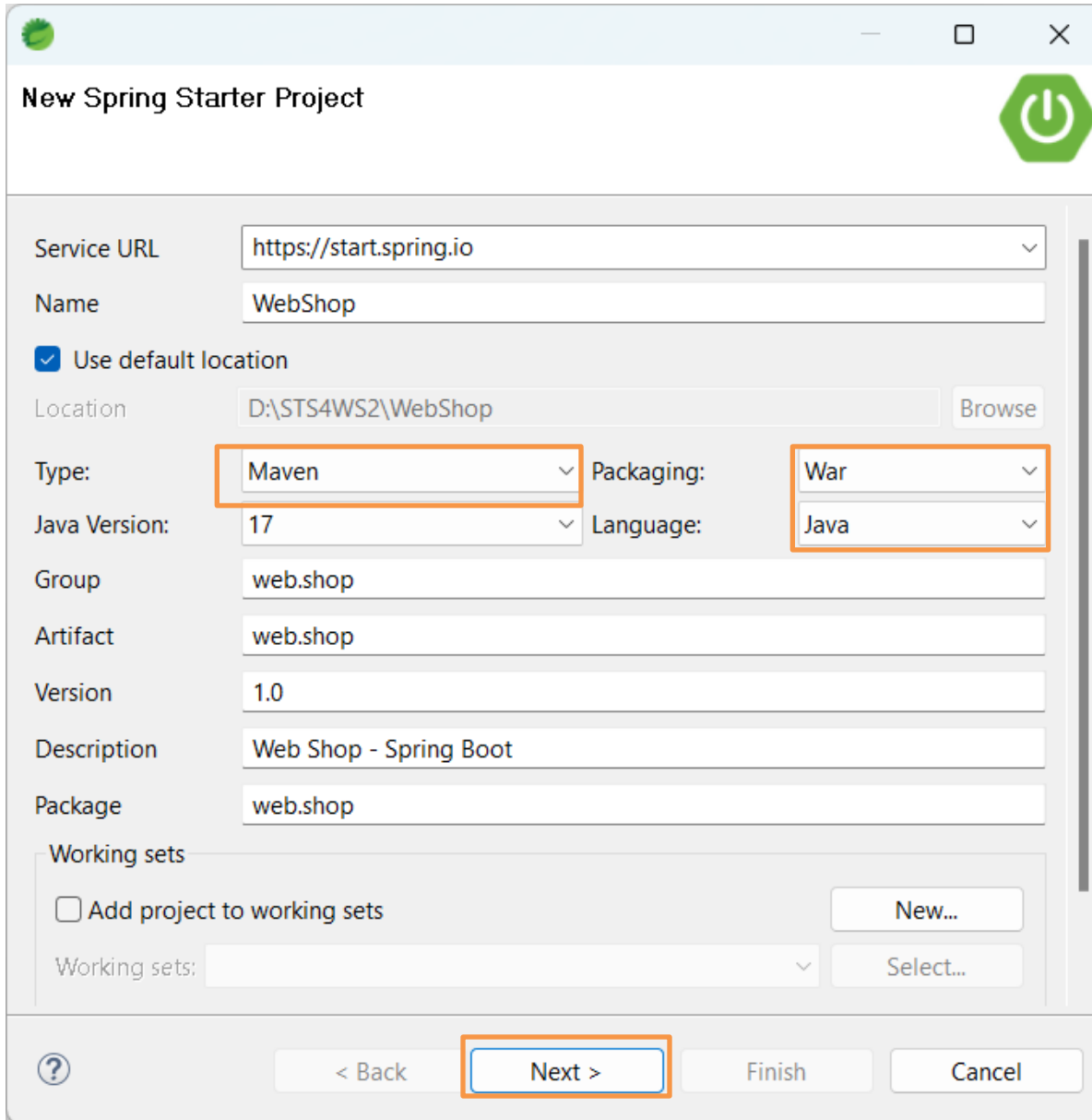
Folder

File

Untitled Text File

JUnit Test Case

Other...Ctrl+N



New Spring Starter Project

Service URL:

Name:

☒ Use default location

Location:

Type: Packaging:

Java Version: Language:

Group:

Artifact:

Version:

Description:

Package:

Working sets

☐ Add project to working sets

Working sets:

❑ Cung cấp đầy đủ thông tin cho dự án của bạn

❑ Cần chú ý:

❖ Type: Maven

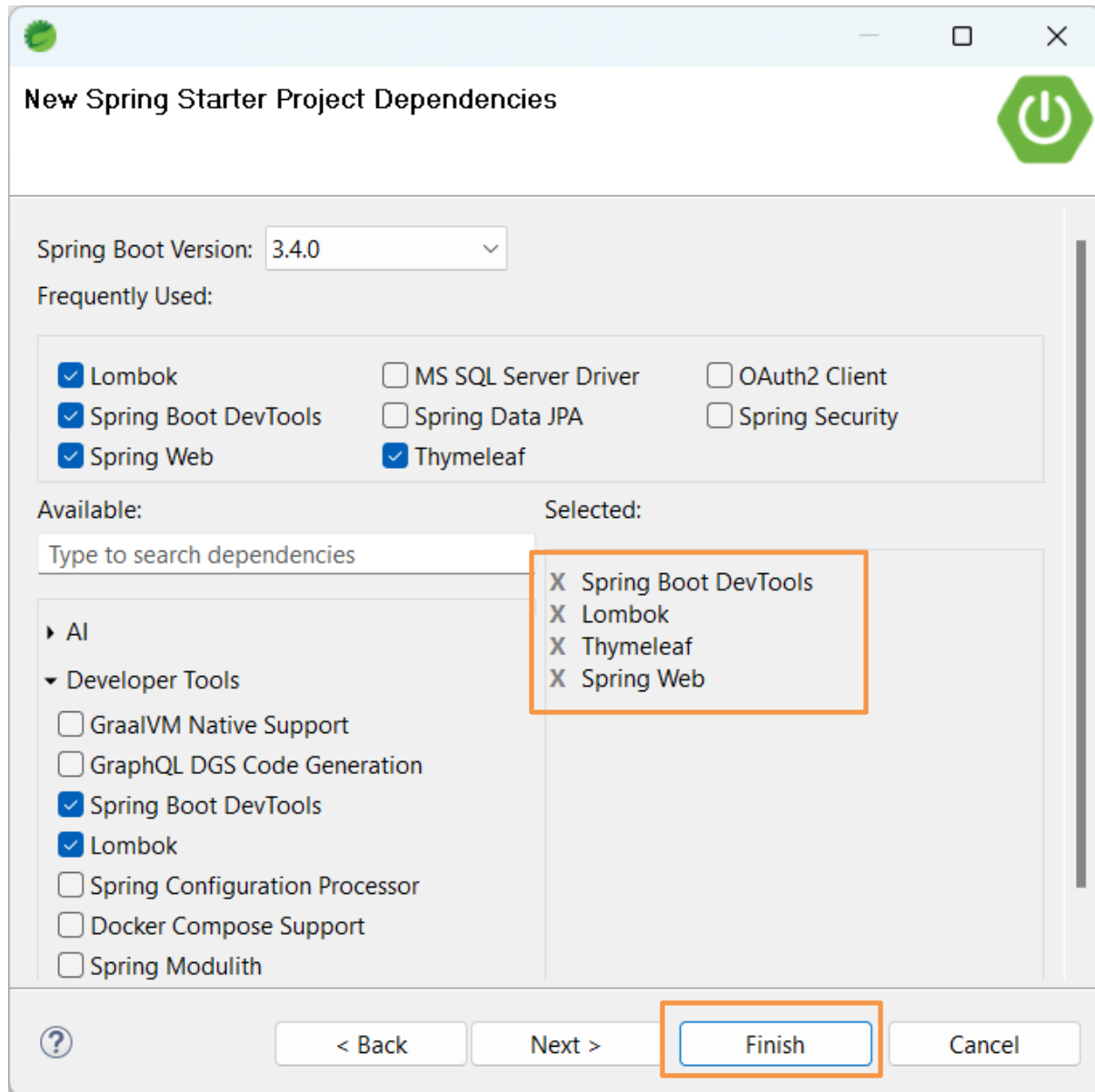
❖ Packaging

➤ War: nếu bạn muốn sản phẩm được đóng gói và triển khai lên Web Server (Tomcat)

➤ Jar: nếu bạn muốn đóng gói như 1 ứng dụng độc lập (đã bao gồm web server)

❖ Language: Java

THƯ VIỆN PHỤ THUỘC (DEPENDENCIES)



☐ Bạn cần chọn những thư viện cần thiết cho dự án của mình khi tạo dự án hoặc có thể bổ sung sau khi cần

☐ Với dự án này

❖ Developer Tools

➤ Spring Boot DevTools

➤ Lombok

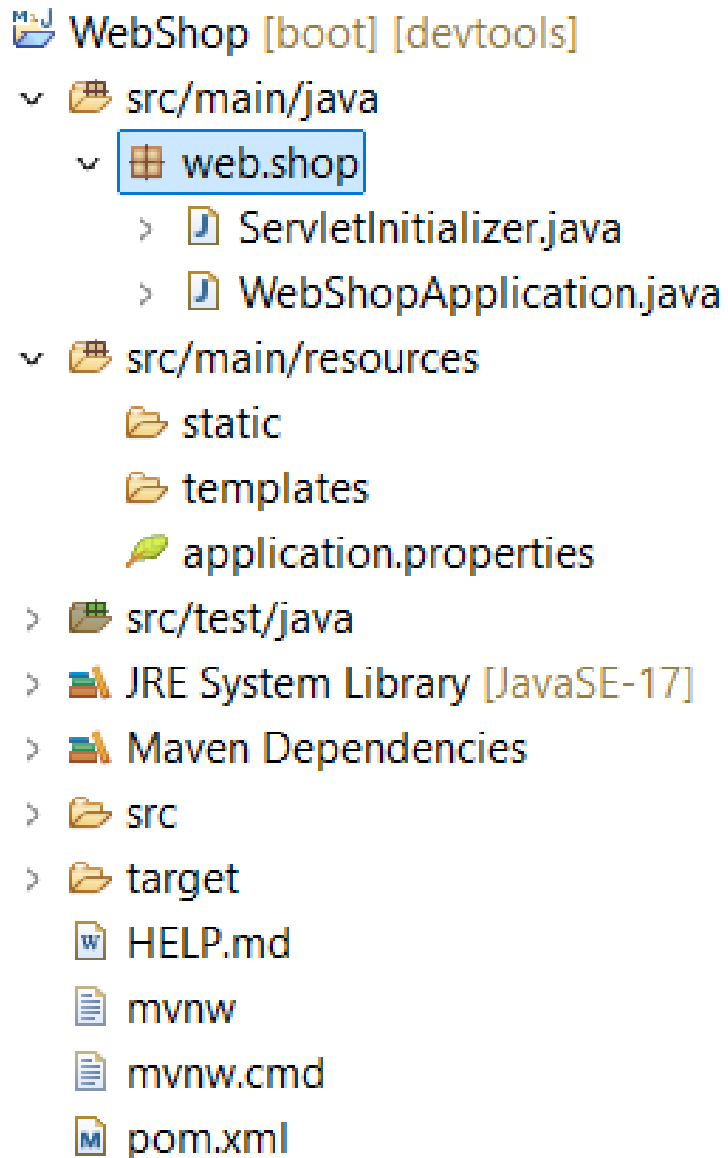
❖ Template

➤ Thymeleaf

❖ Web

➤ Spring Web

☐ *Chú ý: Thư viện phụ thuộc được STS4 tự động tải từ Internet nên bạn cần có kết nối Internet*



□ /src/main/java

- ❖ Thư mục này chứa mã nguồn java. Khi tạo dự án mình chọn Type: War nên trong package web.shop có 2 file cấu hình, nếu chọn Type: Jar thì dự án chỉ tạo 1 file

□ /src/main/resources

- ❖ Static: thư mục chứa các tài nguyên tĩnh (js, css, hình ảnh, file tải về...)
- ❖ Templates: Thư mục chứa file template (thymeleaf)
- ❖ Application.properties
 - Khai báo các thông số của ứng dụng (CSDL, SMTP, upload file, đăng nhập từ mạng xã hội...)

□ pom.xml

- ❖ Khai báo các thư viện phụ thuộc (dependencies) lúc tạo dự án và bổ sung thêm sau này.

□ Maven Dependencies

- ❖ Chứa các thư viện phụ thuộc của dự án

```
<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-thymeleaf</artifactId>
  </dependency>

  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>

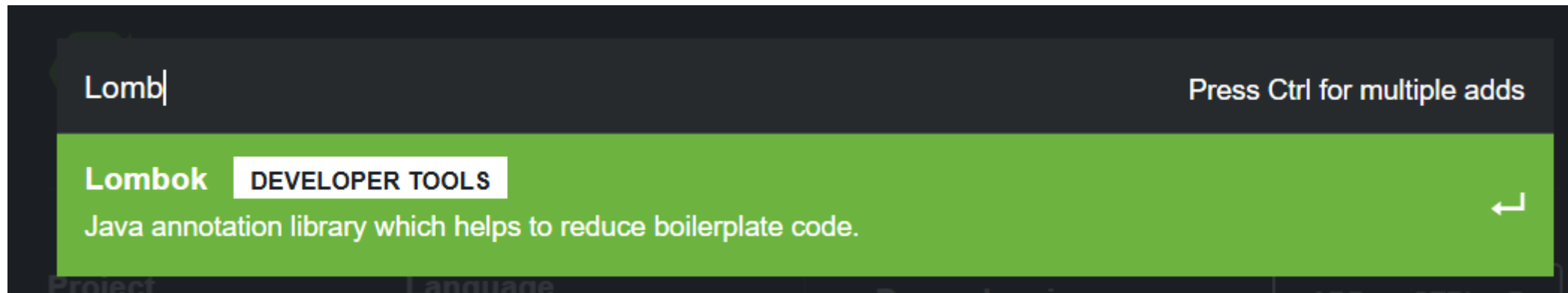
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-devtools</artifactId>
    <scope>runtime</scope>
    <optional>true</optional>
  </dependency>
```

- ☐ <dependency> được sử dụng để khai báo thư viện phụ thuộc được chọn khi chúng ta tạo dự án.
- ☐ Bạn có thể khai báo bổ sung khi cần

- ❑ Vào <https://start.spring.io/> để tìm kiếm và sao chép dependency vào pom.xml khi cần.



- ❑ Nhấp chuột vào nút [Add Dependencies]
- ❑ Nhập tên thư viện phụ thuộc vào ô tìm kiếm
- ❑ Nhấp vào hộp màu xanh thư viện được tìm thấy
 - ❖ Lập lại bước này nếu bạn muốn thêm thư viện phụ thuộc khác



❑ Nhấp vào nút [Explore] và chép các thẻ <dependency> vào pom.xml



WebShop [boot] [devtools]














- > src/main/java
- > src/main/resources
- > src/test/java
- > JRE System Library [JavaSE-17]

Maven Dependencies

- > accessors-smart-2.5.1.jar - C:\Users\DELL\.m2\repository\net\minidev\accessors-smart\2.5.1
- > android-json-0.0.20131108.vaadin1.jar - C:\Users\DELL\.m2\repository\com\vaadin\external\google\android-json\0.0.20131108.vaadin1
- > apiguardian-api-1.1.2.jar - C:\Users\DELL\.m2\repository\org\apiguardian\apiguardian-api\1.1.2
- > asm-9.6.jar - C:\Users\DELL\.m2\repository\org\ow2\asm\asm\9.6
- > assertj-core-3.26.3.jar - C:\Users\DELL\.m2\repository\org\assertj\assertj-core\3.26.3
- > attoparser-2.0.7.RELEASE.jar - C:\Users\DELL\.m2\repository\org\attoparser\attoparser\2.0.7.RELEASE
- > awaitility-4.2.2.jar - C:\Users\DELL\.m2\repository\org\awaitility\awaitility\4.2.2
- > byte-buddy-1.15.10.jar - C:\Users\DELL\.m2\repository\net\bytebuddy\byte-buddy\1.15.10
- > byte-buddy-agent-1.15.10.jar - C:\Users\DELL\.m2\repository\net\bytebuddy\byte-buddy-agent\1.15.10
- > hamcrest-2.2.jar - C:\Users\DELL\.m2\repository\org\hamcrest\hamcrest\2.2
- > jackson-annotations-2.18.1.jar - C:\Users\DELL\.m2\repository\com\fasterxml\jackson\core\jackson-annotations\2.18.1
- > jackson-core-2.18.1.jar - C:\Users\DELL\.m2\repository\com\fasterxml\jackson\core\jackson-core\2.18.1
- > jackson-databind-2.18.1.jar - C:\Users\DELL\.m2\repository\com\fasterxml\jackson\core\jackson-databind\2.18.1
- > jackson-datatype-jdk8-2.18.1.jar - C:\Users\DELL\.m2\repository\com\fasterxml\jackson\datatype\jackson-datatype-jdk8\2.18.1
- > jackson-datatype-jsr310-2.18.1.jar - C:\Users\DELL\.m2\repository\com\fasterxml\jackson\datatype\jackson-datatype-jsr310\2.18.1
- > jackson-module-parameter-names-2.18.1.jar - C:\Users\DELL\.m2\repository\com\fasterxml\jackson\module\jackson-module-parameter-names\2.18.1
- > jakarta.activation-api-2.1.3.jar - C:\Users\DELL\.m2\repository\jakarta\activation\jakarta.activation-api\2.1.3
- > jakarta.annotation-api-2.1.1.jar - C:\Users\DELL\.m2\repository\jakarta\annotation\jakarta.annotation-api\2.1.1
- > jakarta.xml.bind-api-4.0.2.jar - C:\Users\DELL\.m2\repository\jakarta\xml\bind\jakarta.xml.bind-api\4.0.2



XÂY DỰNG HELLO WORLD PAGE

- ▼  WebShop [boot] [devtools]
 - ▼  src/main/java
 - ▼  web.shop
 - >  ServletInitializer.java
 - >  WebShopApplication.java
 - ▼  web.shop.controller
 - >  HelloController.java
 - ▼  src/main/resources
 -  static
 - ▼  templates
 - ▼  demo
 -  hello.html
 -  application.properties

❑ HelloController.java

- ❖ Điều khiển các request từ trình duyệt

❑ /demo/hello.html

- ❖ Giao diện response cho trình duyệt

❑ Chú ý: Theo cấu hình mặc định thì

- ❖ Package của controller và các thành phần khác trong dự án phải bắt đầu từ package chứa file cấu hình của ứng dụng (web.shop).
- ❖ Tất cả các template (file giao diện thymeleaf) phải đặt trong thư mục templates

TỔ CHỨC MÃ NGUỒN HELLO WORLD PAGE

HelloController.java ×

```
1 package web.shop.controller;
2
3 import org.springframework.stereotype.Controller;
4 import org.springframework.ui.Model;
5 import org.springframework.web.bind.annotation.RequestMapping;
6
7 @Controller
8 public class HelloController {
9     @RequestMapping("/poly/hello")
10    public String sayHello(Model model) {
11        model.addAttribute("title", "FPT Polytechnic");
12        model.addAttribute("subject", "Spring Boot MVC");
13        return "/demo/hello.html";
14    }
15 }
```

hello.html ×

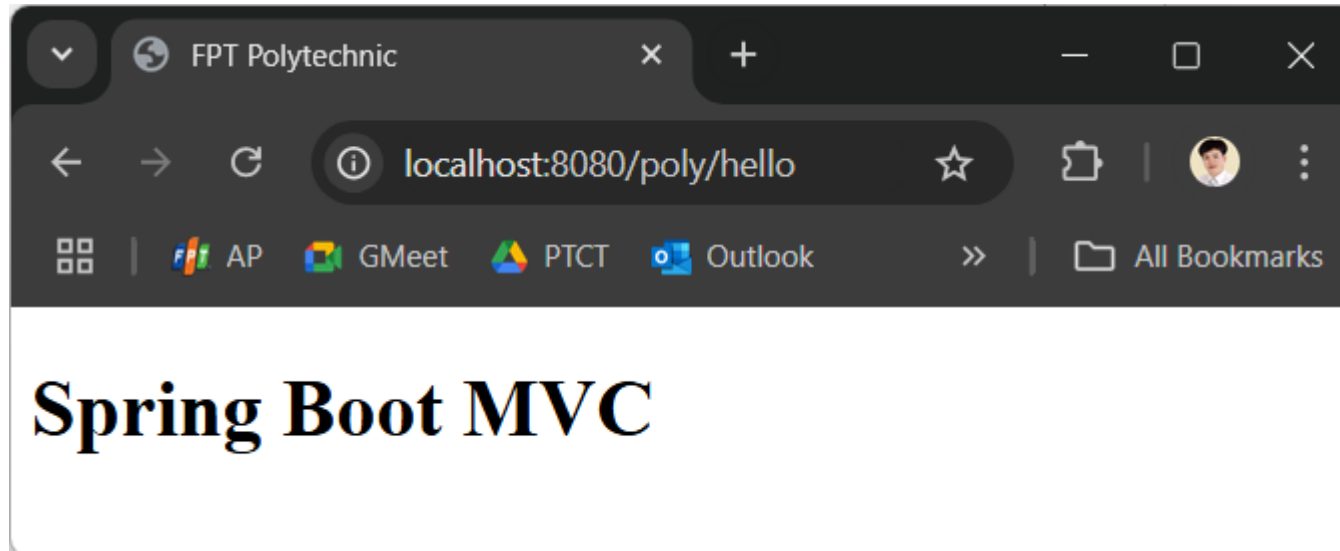
```
1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3
4 <head>
5     <title th:text="${title}">Page Title</title>
6 </head>
7
8 <body>
9     <h1 th:text="${subject}">Page Subject</h1>
10 </body>
11
12 </html>
13
14
15
```

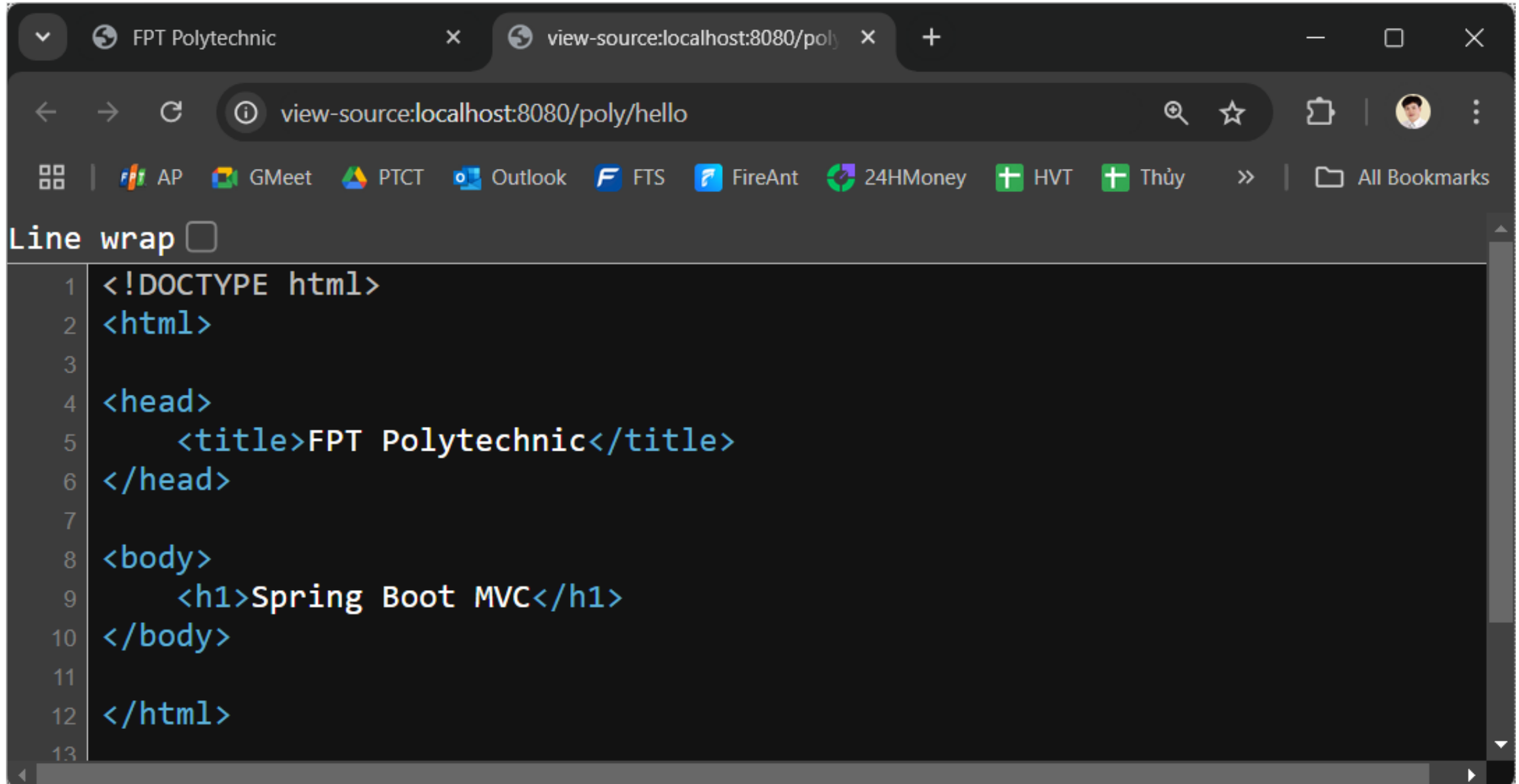
❑ Bước 1: Start Web Server

❖ Nhấp phải chuột vào dự án >> Run As >> Spring Boot App

```
in] o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on port 35729
in] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8080 (http) with context path '/'
in] web.shop.WebShopApplication : Started WebShopApplication in 1.779 seconds (process running for 2.599)
c-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherServlet'
c-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
c-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 1 ms
```

❑ Bước 2: Gửi yêu cầu (request) từ trình duyệt





```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5     <title>FPT Polytechnic</title>
6 </head>
7
8 <body>
9     <h1>Spring Boot MVC</h1>
10 </body>
11
12 </html>
13
```



SPRING BOOT MVC INTRODUCTION

SLIDE 1.2

❑ @Controller

- ❖ Chú thích cho hệ thống biết class này chứa các phương thức điều khiển request từ người dùng

❑ @RequestMapping("URL") -> sayHello(){}

- ❖ Ánh xạ URL với phương thức, nghĩa là khi request chứa URL này thì phương thức này sẽ chạy để điều khiển request đó.

❑ Lệnh return "/demo/hello.html"

- ❖ Chuyển tiếp sang giao diện hello.html đặt trong thư mục templates/demo.
- ❖ Theo cấu hình mặc định thì có thể bỏ ~~html~~ thành return "demo/hello"

❑ Model

- ❖ Thành phần chứa các biến dữ liệu được Controller chia sẻ cho View (giao diện)

❑ Model.addAttribute("subject", "Spring Boot MVC")

- ❖ Bổ sung một biến dữ liệu vào model

❑ `<html xmlns:th="http://www.thymeleaf.org">`

- ❖ Khai báo namespace cho trang Thymeleaf trong đó th là tiếp đầu ngữ mặc định, nghĩ là nếu không có khai báo này thì mã thymeleaf trong trang phải bắt đầu là **th**:

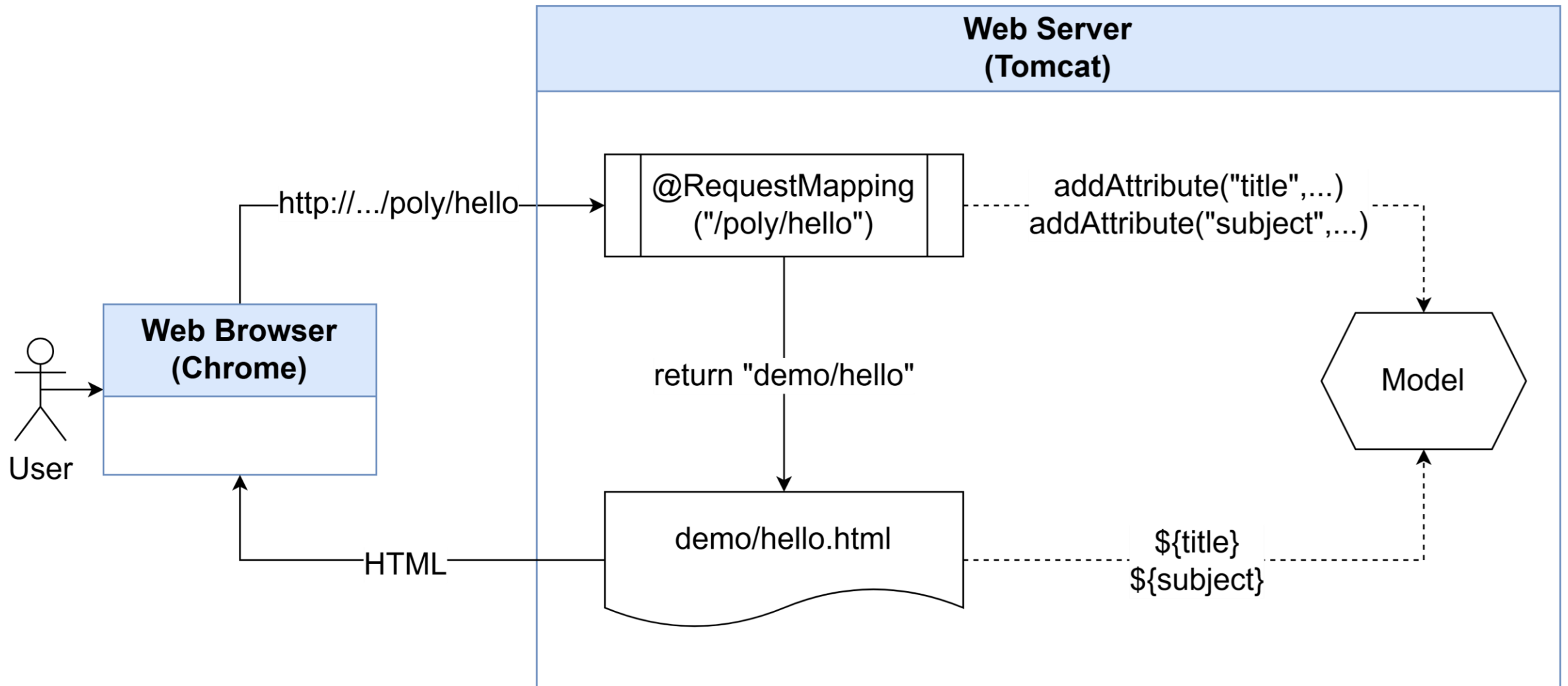
❑ `<h1 th:text="${subject}">Page Subject</h1>`

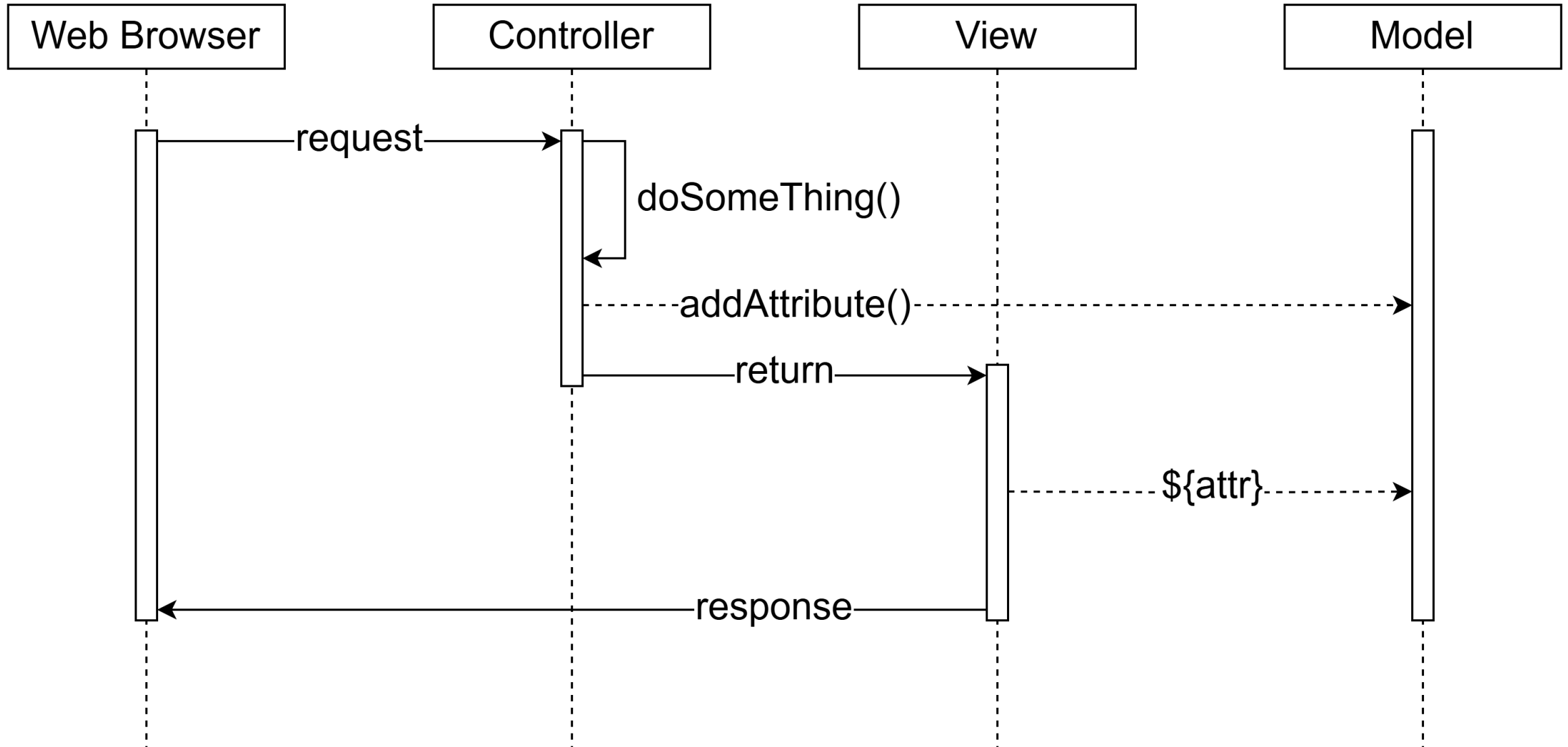
- ❖ th:text: chỉ thị (directive) này được sử dụng để chèn giá trị của nó vào giữa thẻ

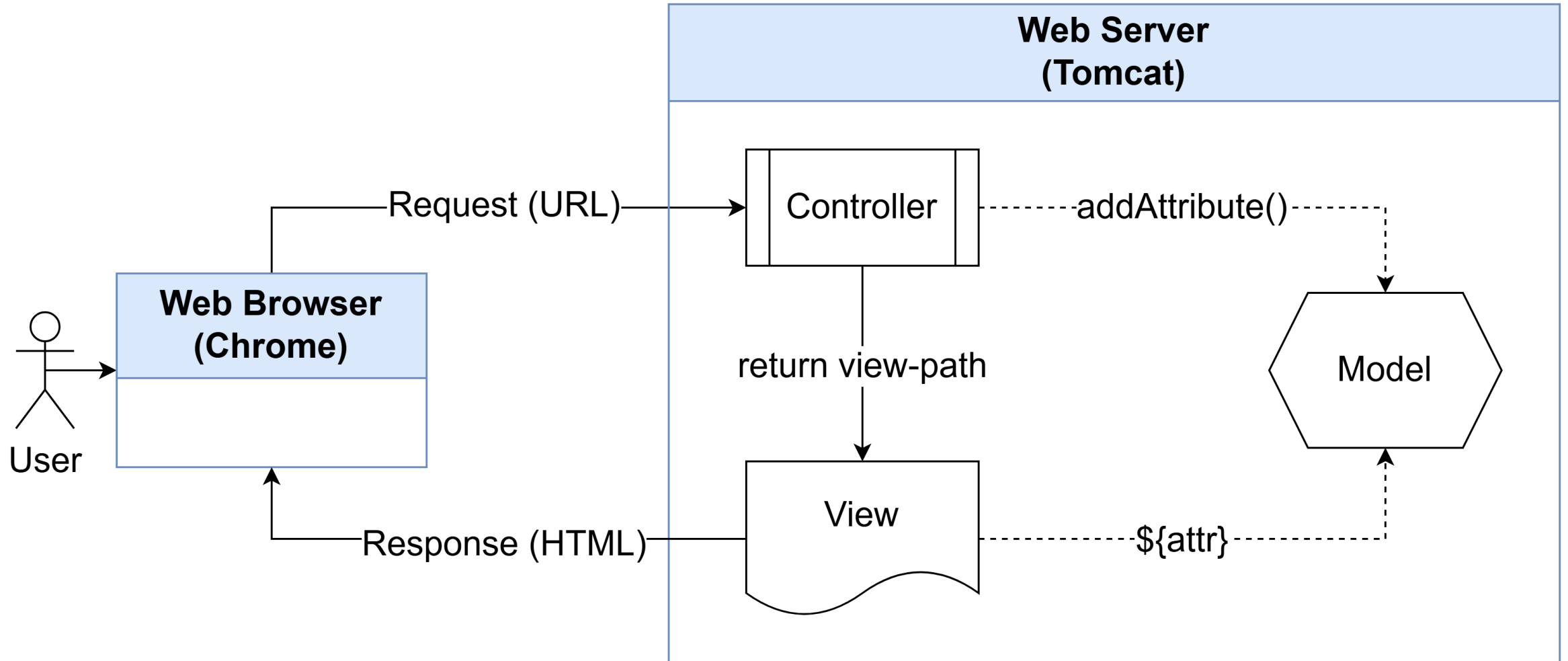
❖ **`${subject}`**

- `${}`: biểu thức (expression) này dùng để truy xuất các biến trong Model được Controller đặt vào trong đó (title, subject)
- subject: tên biến

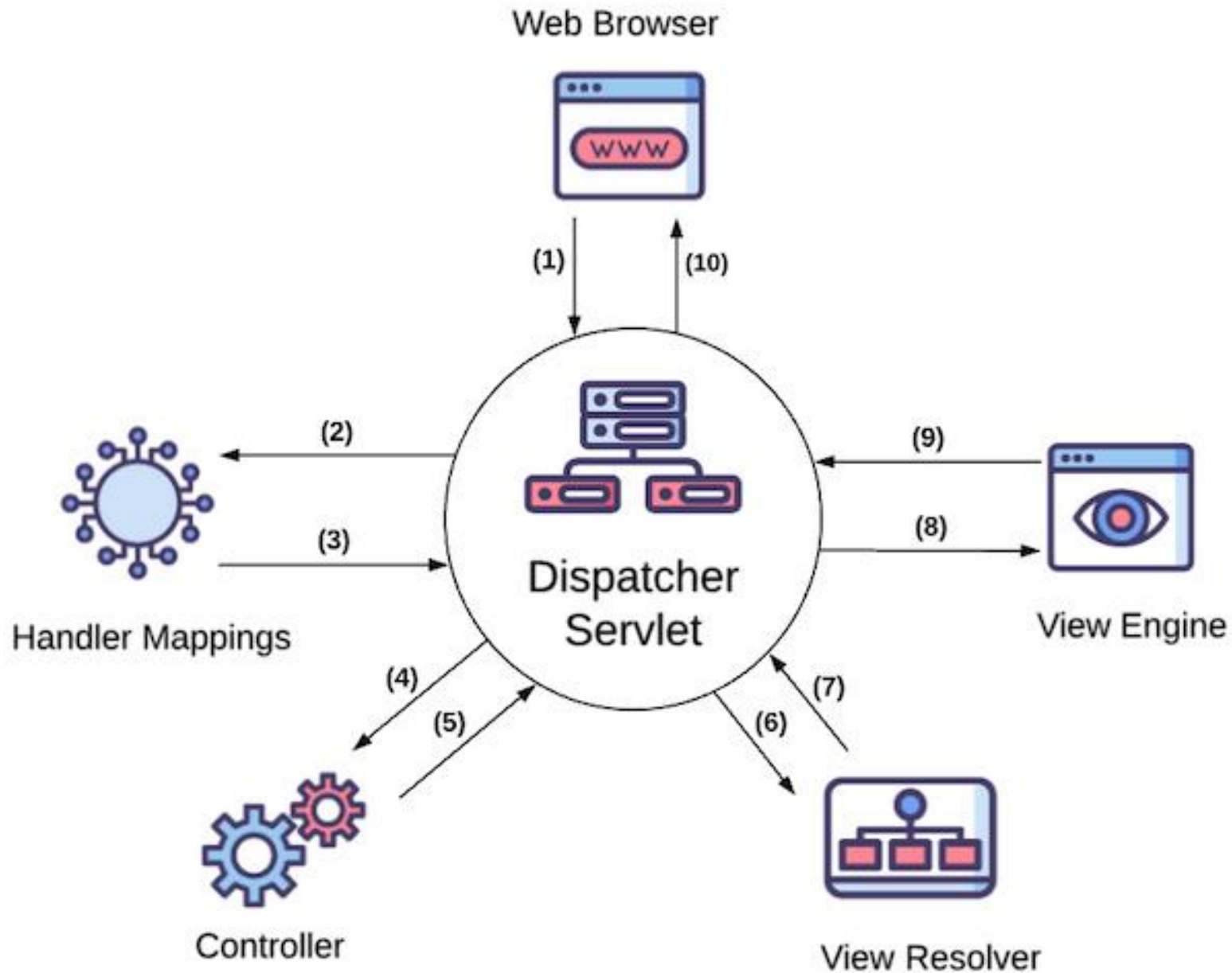
MÔ HÌNH HOẠT ĐỘNG HELLO WORLD PAGE







QUY TRÌNH XỬ LÝ REQUEST & RESPONSE



- ❑ Spring Boot MVC được xây dựng dựa trên nền tảng của Servlet. Khi lập trình Servlet thì chúng ta thường xuyên sử dụng các thành phần:
 - ❖ HttpServletRequest
 - ❖ HttpServletResponse
 - ❖ HttpSession
 - ❖ ServletContext
- ❑ Để sử dụng các thành phần ở trên trong các thành phần của Spring Boot MVC bạn chỉ việc “tiêm” (inject) các thành phần trên vào field của class hoặc argument của phương thức.
 - ❖ Sử dụng @Autowire để tiêm vào field
 - ❖ Thêm vào đối số của các phương thức điều khiển (mapping method) để tiêm vào phương thức

```
@Controller
public class MyController {
    @Autowired
    ServletContext application;
    @Autowired
    HttpSession session;
    @Autowired
    HttpServletRequest request;
    @Autowired
    HttpServletResponse response;
    @RequestMapping("/url.php")
    public String sayHello() {
        String fullname = request.getParameter("hoten");
        request.setAttribute("message", "Http Components");
        return "hello";
    }
}
```

```
@Controller
public class MyController {
    @Autowired
    ServletContext application;
    @RequestMapping("/url.php")
    public String sayHello(HttpSession session,
        HttpServletRequest request, HttpServletResponse response;) {
        String fullname = request.getParameter("hoten");
        request.setAttribute("message", "Http Components");
        return "hello";
    }
}
```

- ❑ Chú ý: ServletContext chỉ được phép tiêm vào field, không cho phép tiêm qua đối số của phương thức điều khiển.

☑ PHẦN I:

- ☑ GIỚI THIỆU SPRING, SPRING MVC VÀ SPRING BOOT
- ☑ CÀI ĐẶT MÔI TRƯỜNG PHÁT TRIỂN
- ☑ TẠO VÀ KHẢO SÁT DỰ ÁN SPRING BOOT
- ☑ QUẢN LÝ THƯ VIỆN PHỤ THUỘC
- ☑ XÂY DỰNG TRANG HELLO WORLD

☑ PHẦN II:

- ☑ MÔ HÌNH XỬ LÝ REQUEST/RESPONSE
- ☑ LÀM VIỆC VỚI HTTP COMPONENTS TRONG SPRING BOOT MVC
 - ☑ HTTPServletRequest
 - ☑ HttpServletResponse
 - ☑ HttpSession
 - ☑ ServletContext





FPT Education

FPT POLYTECHNIC

Thank you