MỤC TIÊU:

Kết thúc bài thực hành này bạn có khả năng

- ✓ Thực hiện 2 way data binding
- ✓ Thực hiện form validation
- ✓ Tổ chức layout
- ✓ Tổ chức website đa ngôn ngữ

PHẦN I

Bài 1: Databinding (2 điểm)

Hãy xây dựng trang web thực hiện ràng buộc dữ liệu 2 chiều giữa bean và form sau đây để duy trì dữ liệu đã nhập sau khi submit form.

Bean	Form

@NoArgsConstructor
@AllArgsConstructor
@Builder
@Data
public class Staff {
  private String id;
  private String fullname;
  @Default
  private String photo = "photo.jpg";
  @Default
  private Boolean gender;
  @Default
  @DateTimeFormat(
  pattern=”MM/dd/yyyy”)
  private Date birthday = **new** Date();
  @Default
  private double salary = 12345.6789;
  @Default
  private Integer level = 0;
	Vui lòng nhập thông tin nhân viên!
	Email:
	Họ và tên:
	Giới tính: ○ Nam ○ Nữ
	Ngày sinh:
	Hình ảnh: Choose File No file chosen
	Cấp bậc: Úy
	Salary:
	Create

}	

Hướng dẫn:

Bước 1: Tạo bean Staff

Bước 2: Xây dựng form staff-create.html

```
<i th:text="${message}"></i>
<form th:object="${staff}" action="/staff/create/save"
                    method="post" enctype="multipart/form-data">
        Email: <input th:field="*{id}"><hr>

        Họ và tên: <input th:field="*{fullname}"><hr>

        Giới tính:
        <input type="radio" th:field="*{gender}" value="true">Nam
        <input type="radio" th:field="*{gender}" value="false">Nữ<hr>

        Ngày sinh: <input name="birthday"
                    th:value="*{#dates.format(birthday, 'MM/dd/yyyy')}"><hr>
        Hình ảnh:
        <input th:field="*{photo}" type="hidden">
        <input name="photo_file" type="file"><hr>

        Cấp bậc:
        <select th:field="*{level}">
            <option value="0">Úy</option>
            <option value="1">Tá</option>
            <option value="2">Tướng</option>
        </select><hr>

        Salary: <input th:field="*{salary}"><hr>
        <button>Create</button>
</form>
```

Chú ý:

- Sử dụng th:object để chọn bean staff (trong Model)
- Sử dụng th:field để buộc các thuộc tính của bean vào các trường trên form

- Với ngày sinh chúng ta cần định dạng dữ liệu nên phải dùng th:value để hiển thị dữ liệu và thuộc tính name phải sử dụng tên của thuộc tính bean
- Với hình ảnh là file upload nên cần sử dụng trường file và đặt tên riêng để xử lý đồng thời buộc thuộc tính với trường ẩn để giữ dữ liệu của thuộc tính bean.
  Chú ý: form upload nên method phải là post, enctype phải là multipart/form-data.

Bước 3: Xây dựng controller StaffController

```
@RequestMapping("/staff/create/form")
public String createForm(Model model, @ModelAttribute("staff") Staff staff) {
        model.addAttribute("message", "Vui lòng nhập thông tin nhân viên!");
        return "/demo/staff-create";
}

@RequestMapping("/staff/create/save")
public String createSave(Model model, @ModelAttribute("staff") Staff staff,
                                @RequestPart("photo_file") MultipartFile photoFile) {
        // Gán tên file upload cho thuộc tính photo của bean nếu có upload file
        if(!photoFile.isEmpty()) {
                staff.setPhoto(photoFile.getName());
        }
        model.addAttribute("message", "Xin chào " + staff.getFullname());
        return "/demo/staff-create";
}
```

Chú ý:

- @/staff/create/form: hiển thị form
- @/staff/create/save: tiếp nhận và xử lý dữ liệu form, địa chỉ url này được gắng với thuộc tính action của <form>
- Đối số: @ModelAttribute("staff") Staff staff thực hiện 2 công việc
  - Tạo bean mới
  - Đọc dữ liệu từ các tham số form và gán cho các thuộc tính cùng tên của bean
  - Bổ sung bean mới tạo vào Model với tên là staff

Bài 2: Form validation (2 điểm)

Hãy tiến hành kiểm tra dữ liệu nhập vào form (validation) theo yêu cầu sau

Form	Yêu cầu kiểm tra
Vui lòng sửa các lỗi sau! Email: *Chưa nhập email* Họ và tên: *Chưa nhập họ và tên* Giới tính: Nam Nữ *Chưa chọn giới tính* Ngày sinh: *Chưa nhập ngày sinh* Hình ảnh: Choose File No file chosen Cấp bậc: Úy Salary: *Chưa nhập lương* Validate	**Email**: *+ Không để trống* *+ Đúng định dạng email* **Fullname**: *+ Không để trống* **Gender**: *+ Phải chọn* **Birthday**: *+ Không để trống* *+ Phải là ngày trong quá khứ* **Salary**: *+ Không để trống, trên 1000*

Hướng dẫn:

Bước 1: Khai báo thư viện phụ thuộc (nếu chưa khai báo)

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

Bước 2: Khai báo bổ sung các annotation vào bean class

```
@NoArgsConstructor
@AllArgsConstructor
@Builder
@Data
public class Staff {
        @NotBlank(message = "Chưa nhập email")
        @Email(message = "Email không đúng định dạng")
        private String id;

        @NotBlank(message = "Chưa nhập họ và tên")
        private String fullname;
```

```
        @Default
        private String photo = "photo.jpg";

        @NotNull(message = "Chưa chọn giới tính")
        private Boolean gender;

        @NotNull(message = "Chưa nhập ngày sinh")
        @Past(message = "Ngày sinh không hợp lệ")
        @DateTimeFormat(pattern = "MM/dd/yyyy")
        private Date birthday;

        @Min(value = 1000, message = "Lương tối thiểu phải là 1000")
        @NotNull(message = "Chưa nhập lương")
        private Double salary;

        private Integer level;
}
```

Chú ý:

- @NotNull áp dụng cho mọi kiểu dữ liệu (trừ các kiểu nguyên thủy)
- @NotBlank áp dụng cho chuỗi
- @NotEmpty áp dụng cho chuỗi và tập hợp (Collection, List, Set)

Bước 3: Hiển thị thông báo lỗi lên form

```
<i th:text="${message}"></i>
<form th:object="${staff}" action="/staff/create/save" method="post"
enctype="multipart/form-data">
        Email:
        <input th:field="*{id}">
        <i th:errors="*{id}"></i><hr>
        Họ và tên:
        <input th:field="*{fullname}">
        <i th:errors="*{fullname}"></i><hr>
        Giới tính:
        <input type="radio" th:field="*{gender}" value="true">Nam
        <input type="radio" th:field="*{gender}" value="false">Nữ
        <i th:errors="*{gender}"></i><hr>
```

```
        Ngày sinh:
        <input name="birthday"
                        th:value="*{#dates.format(birthday, 'MM/dd/yyyy')}">
        <i th:errors="*{birthday}"></i><hr>
        Hình ảnh:
        <input th:field="*{photo}" type="hidden">
        <input name="photo_file" type="file"><hr>
        Cấp bậc:
        <select th:field="*{level}">
                <option value="0">Úy</option>
                <option value="1">Tá</option>
                <option value="2">Tướng</option>
        </select><hr>
        Salary:
        <input th:field="*{salary}">
        <i th:errors="*{salary}"></i><hr>
        <button>Validate</button>
</form>
<style>
        i{color:red;}
</style>
```

Chú ý:

- Sử dụng th:errors để hiển thị lỗi của thuộc tính bean. Bạn có thể đặt bất kỳ đâu tùy thích (trong bài này đặt bên phải các trường buộc với các field)

Bước 4: Hiệu chỉnh controller để thực hiện kiểm tra dữ liệu của bean nhận được từ form

```
@RequestMapping("/staff/create/save")
public String createSave(Model model,
                @RequestPart("photo_file") MultipartFile photoFile,
                @Valid @ModelAttribute("staff") Staff staff, Errors errors) {
        if(!photoFile.isEmpty()) {
                staff.setPhoto(photoFile.getName());
        }
        if(errors.hasErrors()) {
                model.addAttribute("message", "Vui lòng sửa các lỗi sau!");
```

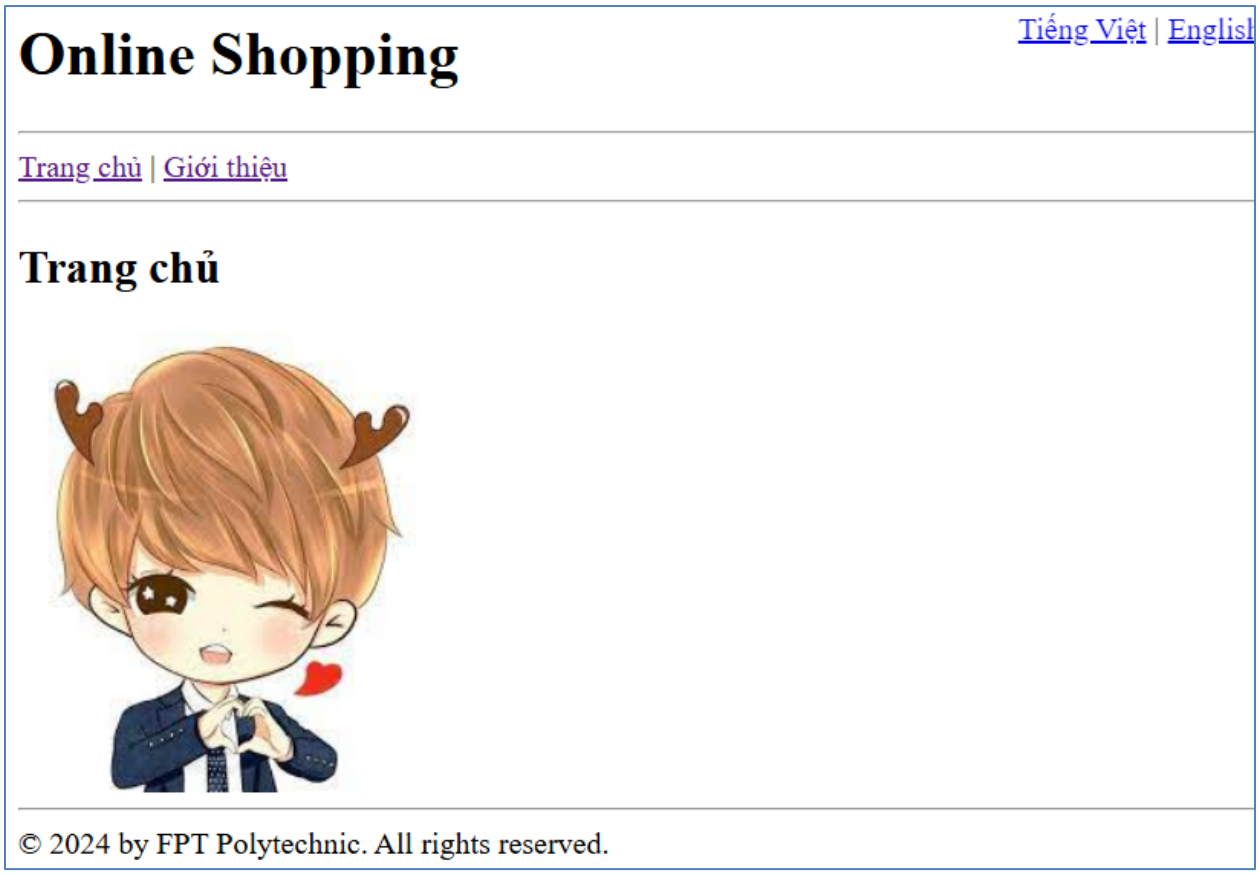
```
        } else {
                model.addAttribute("message", "Dữ liệu đã nhập đúng!");
        }
        return "/demo/staff-validate";
}
```

Bài 3 (1 điểm)

Giảng viên cho thêm

Bài 4 (2 điểm)

Xây dựng một website gồm 2 trang web là trang chủ và trang giới thiệu có bố cục giao diện gồm header, menu, content và footer. Trong đó content sẽ thay đổi theo nội dung của trang, những phần còn lại là cố định. Ngoài ra tiêu đề cửa số của trang web cũng thay đổi theo từng trang khác nhau.



Hướng dẫn:

Bước 1: Thiết kế layout có bố cục chung

Trang layout.html

```
<html th:fragment="view(title, content)">
	<head>
		<title th:replace="${title}">TITLE</title>
	</head>
	<body>
		<header><h1>Online Shopping</h1></header>
		<nav th:replace="~{/shared/menu}">MENU</nav>
		<article th:replace="${content}">CONTENT</article>
		<footer>
			<hr>&copy; 2024 by FPT Polytechnic. All rights reserved.
		</footer>
	</body>
</html>
```

Trang menu.html

```
<hr>
<nav>
	<a href="/home/index">Trang chủ</a> |
	<a href="/home/about">Giới thiệu</a>
	<div style="position: fixed; top: 5px; right: 5px;">
```

```
            <a href="#">Tiếng Việt</a> |
            <a href="#">English</a>
        </div>
</nav>
<hr>
```

Chú ý:

- Sử dụng th:replace để thay thế thẻ bằng fragment (một phân đoạn giao diện)
- ~{/shared/menu} là fragment lấy toàn bộ file menu.html đặt trong đừng dẫn templates/shared
- <html th:fragment=”view(title, content)”> đặt tên fragment cho thẻ <html>. Trong đó view là tên fragment, title và content là 2 đối số.

Bước 2: Thiết kế các trang thành viên (home.html và about.html)

Trang home.html

```
<html th:replace="~{/shared/layout::view(~{::title}, ~{::article})}">
    <head>
        <title>Trang chủ</title>
    </head>
    <body>
        <article>
            <h1>Trang chủ</h1>
            <img src="/photos/photo.jpg">
        </article>
    </body>
</html>
```

Trang about.html

```
<html th:replace="~{/shared/layout::view(~{::title}, ~{::article})}">
    <head>
        <title>Giới thiệu</title>
    </head>
    <body>
        <article>
            <h1>Giới thiệu</h1>
            <ul>
```

```
                        <li><h3>FPT Polytechnic</h3></li>
                        <li><h3>Thực học - Thực nghiệp</h3></li>
                    </ul>
                </article>
            </body>
</html>
```

Chú ý:

- ~{/shared/layout::view(~{::title}, ~{::title})} là fragment có tên là view trong file layout.html đặt trong thư mục templates/shared
- ~{::title} là <title>, ~{::article} là <article> trong cùng trang

Bước 3: Xây dựng controller và chạy thử

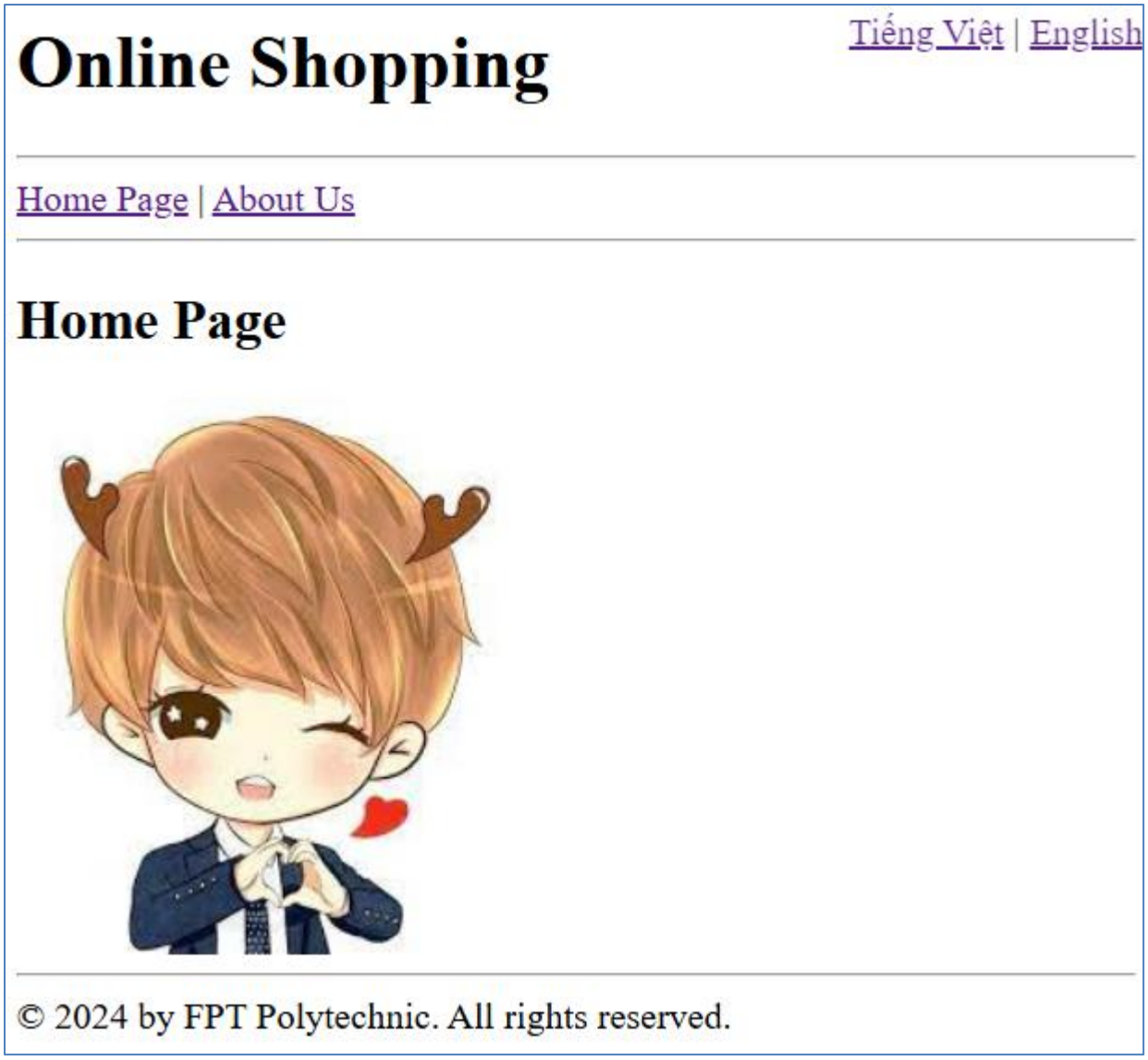
```
@RequestMapping("/home/index")
public String index(Model model) {
        return "/home/index";
}

@RequestMapping("/home/about")
public String about(Model model) {
        return "/home/about";
}
```

Bài 5 (2 điểm)

Chuyển đổi website của bài 4 thành website gồm 2 thứ tiếng (tiếng Anh và tiếng Việt) được mô tả như các hình sau:

Hình 1: Khi chọn tiếng Việt (nhấp link Tiếng Việt)



Hình 2: Khi chọn tiếng Anh (nhấp link English)

Tổng kết các thành phần cần thực hiện đa ngôn ngữ: header, footer, menu và tiêu đề của các trang.

Hướng dẫn:

Bước 1: Tạo và xây dựng các file tài nguyên đa ngôn ngữ

Tạo folder i18n trong thư mục src/main/resources sau đó tạo 2 file tài nguyên layout.properties chứa tài nguyên tiếng Anh và layout.properties chứa tài nguyên tiếng Việt. Soạn nội dung cho các file như sau:

layout.properties	layout_vi.properties
header.name=Online Shopping menu.home=Home Page menu.about=About Us footer.copy=© 2024 by FPT Polytechnic. All rights reserved.	header.name=Mua Sắm Trực Tuyến menu.home=Trang chủ menu.about=Giới thiệu footer.copy=© 2024 bởi FPT Polytechnic. Đã đăng ký bản quyền.

Bước 2: Cấu hình nạp các file tài nguyên đa ngôn ngữ và xử lý lựa chọn ngôn ngữ

Tạo lớp cấu hình MessageConfig.java trong package gốc của dự án và viết mã để câu hình nạp các file tài nguyên đa ngôn ngữ và xử lý lựa chọn ngôn ngữ như sau:

```
@Configuration
public class MessageConfig implements WebMvcConfigurer{
	@Bean("messageSource")
	public MessageSource getMessageSource() {
		ReloadableResourceBundleMessageSource ms
				= new ReloadableResourceBundleMessageSource();
		ms.setBasenames("classpath:i18n/layout");
		ms.setDefaultEncoding("utf-8");
		return ms;
	}

	@Bean("localeResolver")
	public LocaleResolver getLocaleResolver() {
		CookieLocaleResolver localeResolver = new
CookieLocaleResolver();
		localeResolver.setCookiePath("/");
		localeResolver.setCookieMaxAge(Duration.ofDays(30));
		localeResolver.setDefaultLocale(new Locale("vi"));
		return localeResolver;
	}

	@Override
	public void addInterceptors(InterceptorRegistry registry) {
		LocaleChangeInterceptor interceptor
				= new LocaleChangeInterceptor();
		interceptor.setParamName("lang");
		registry.addInterceptor(interceptor);
```

```
        }
}
```

Chú ý:

- i18n/layout là đường dẫn của file layout.properties và layout_vi.properties (chỉ lấy tên cơ sở (gọi là basename) không bao gồm mã ngôn ngữ và phần mở rộng của file).
- @Bean LocaleResolver giúp cấu hình cơ chế duy trì ngôn ngữ được chọn. Trong bài này chúng ta sử dụng Cookie để duy trì trong 30 ngày. Chúng ta cũng thiết lập ngôn ngữ mặc định là tiếng Việt (vi)
- Chúng ta cũng cần implement WebMvcConfigurer để đăng ký xử lý lựa chọn ngôn ngữ với tham số lang.

Bước 3: Hiển thị tài nguyên đa ngôn ngữ lên giao diện và lựa chọn ngôn ngữ

- Trang layout.html

```
<header><h1 th:text="#{header.name}">Online
Shopping</h1></header>
...
<footer>
        <hr>
        <th:block th:utext="#{footer.copy}">
                &copy; 2024 by FPT Polytechnic. All rights reserved.
        </th:block>
</footer>
```

- Trang menu.html

```
<a href="/home/index" th:text="#{menu.home}">Trang chủ</a> |
<a href="/home/about" th:text="#{menu.about}">Giới thiệu</a>
<div style="position: fixed; top: 5px; right: 5px;">
        <a href="?lang=vi">Tiếng Việt</a> |
        <a href="?lang=en">English</a>
</div>
```

- Trang home.html

```
<h1 th:text="#{menu.home}">Trang chủ</h1>
```

- Trang about.html

```
<h1 th:text="#{menu.about}">Giới thiệu</h1>
```

Bài 6 (1 điểm)

Giảng viên cho thêm