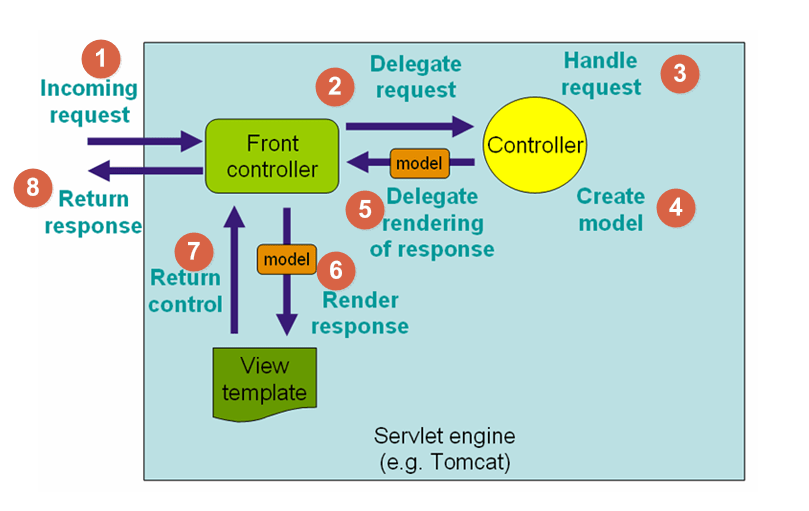
**Audit module 4**

1. **Vẽ và trình bày cách Spring MVC xử lý request?**



1. Front-controller sẽ là nơi đầu tiên nhận tất cả request người dùng gửi đến
2. Front-controller sẽ sử dụng Handle mapping để biết controller nào sẽ xử lý request đó
3. Controller nhận request, gọi các class service xử lý liên quan
4. Sau khi xử lý xong, Controller nhận được model từ tầng Service
5. Controller gửi model vừa nhận được tới Front-controller
6. Front-controller sẽ sử dụng các mẫu view, sử dụng View Resolver và truyền model vào nó
7. View template, view page, model sẽ được gửi lại Front-controller
8. Front-controller gửi một page view để trình duyệt hiển thị ra cho người dùng
9. **Trình bày cơ chế Dependency Injection trong Spring**

* DI là cơ chế để loại bỏ sự phụ thuộc trong chương trình, giúp cho việc quản lý và kiểm thử phần mềm dễ dàng hơn. DI làm cho mã chương trình ít bị phụ thuộc vào nhau hơn

DI là một dạng thực hiện của IoC, bằng cách tiêm (inject) module vào một module khác cần nó. Ví dụ module cấp cao Car cần có module Engine, thì dependency injection sẽ thực hiện theo các bước như sau:

* Tìm và tạo module tương ứng với Engine (class nào có implements Engine interface)
* Tạo tiếp module Car, do Car phụ thuộc vào Engine nên IoC tìm và lấy ra đối tượng Engine đã tạo trước đó và inject vào bên trong Car.
* Xong

Mọi module trong IoC đều gọi là dependency, mặc dù có những module không bị phụ thuộc bởi module nào khác. Khi chương trình chạy, IoC sẽ quét tất cả class đánh dấu dependency, tạo một đối tượng duy nhất (singleton), và bỏ vào cái túi gọi là IoC container, lúc nào cần thì lấy ra sử dụng. Do đó, các module đảm bảo được IoC tạo ra duy nhất một object, giúp tiết kiệm bộ nhớ và quản lý cũng dễ hơn.

Nếu khi tạo module nào đó, mà module đó cần một module khác phụ thuộc, thì IoC sẽ tìm trong IoC container xem có không, nếu có thì inject vào, nếu chưa thì tạo mới, bỏ vào container và inject vào. Việc inject tự động các dependency (module) như thế được gọi là **Dependency injection**.

1. **Có bao nhiêu cách để thực hiện Dependency Injection? Trình bày?**

Có 3 loại cách:

* Constructor-based injection
* Setter-based injection
* Field injection

|  |  |
| --- | --- |
| **Constructor-based injection** | **Setter-based injection** |
| Inject những module bắt buộc | Inject module tùy chọn |
| Có thể inject nhiều dependency cùng lúc | Chỉ inject mỗi dependency mỗi setter |

1. **Framework là gì? Framework khác Library chỗ nào?**

* Framework là các đoạn code được viết sẵn, cấu thành nên một bộ khung và thư viện lập trình được đóng gói. Chúng cung cấp các tính năng có sẵn như mô hình, API và các yếu tố khác để tối giản cho việc lập trình web.
* Library: là tập hợp các chức năng, các lớp được viết sẵn để có thể tái sử dụng. Mỗi function, class phục vụ cho một công việc cụ thể nào đó.

|  |  |
| --- | --- |
| **Framework** | **Library** |
| * Cung cấp các tính năng có sẵn để có thể tái sử dụng | |
| * LTV phải tuân theo cấu trúc, quy tắc của framewok | * LTV sử dụng các tính năng của library để thêm vào dự án của mình |

1. **Spring framework là gì?**

* Spring framework là một mã nguồn mở dùng để phát triển các ứng dụng Java, giúp LTV phát triển ứng dụng có hiệu suất cao, dễ kiểm thử và tái sử dụng.

Các thành phần của Spring:

* Spring core: core, bean, context, SpEL
* Testing: cung cấp các module để test
* Spring Data Acess: module cung cấp cách tương tác DL
* Web Servlet: cung cấp mô hình ứng dụng web
* Web Reactive
* Intergration
* Language: Java, Kotlin, Groovy

1. **Lợi ích của Spring framework?**

* Tăng tốc độ phát triển
* Giúp LTV tập trung xử lý nghiệp vụ
* Cung cấp môi trường làm việc tiêu chuẩn
* Cộng đồng lớn mạnh

1. **Nguyên lý đảo ngược điều khiển(Inversion of Control) là gì?**

* IoC: đảo ngược điều khiển, giúp thay đổi luồng điều khiển của chương trình một cách linh hoạt. IoC container là thành phần thực hiện IoC
* Trong Spring, IoC container sẽ tạo các đối tượng, lắp ráp chúng lại với nhau, cấu hình các đối tượng và quản lý vòng đời của chúng
* Spring container sử dụng DI để quản lý các thành phần, đối tượng để tạo nên 1 ứng dụng. Các thành phần, đối tượng này được gọi là Spring Bean.
* Bean Factory và ApplicationContext đều là các interface để thực hiện Spring Container. ApplicationContext có nhiều chức năng hơn: AOP, xử lý message, context cho web application.

1. **Bean là gì?**

* Bean là 1 đối tượng Java được tạo ra và quản lý bởi Spring container.

1. **Trong Spring có bao nhiêu Bean Scope?**

Có 6 bean scope trong Spring:

* Singleton scope(mặc định): IoC container sẽ khởi tạo duy nhất 1 instance cho bean đó và nó được sử dụng trong tất cả các yêu cầu đến bean này.
* Prototype scope: sẽ trả về 1 instance mới khi có yêu cầu đến IoC container
* Request scope: khởi tạo 1 bean instance cho một Http request
* Session scope: khởi tạo 1 bean instance cho 1 Http session
* Application scope: khởi tạo 1 bean instance cho 1 vòng đời của ServletContext
* Websocket scope: khởi tạo cho 1 websocket session

1. **@Autowire là gì?**

* @Autowire cho phép Spring tự động tìm kiếm và inject các bean tương ứng mà chúng ta đã khai báo trong class.

Có 3 cách để sử dụng autowire:

* Autowire trực tiếp trên thuộc tính
* Autowire setter
* Autowire constructor

1. **@Component nghĩa là gì?**

* @Component là một annotation đánh dấu trên 1 class để Spring biết nó là 1 Bean. Khi chạy Spring boot gặp 1 class đánh dấu là @Component thì nó sẽ tạo ra 1 instance để đưa vào ApplicationContext quản lý.

1. **Trình bày ý nghĩa của Controller?**

* Controller là 1 thành phần trong mô hình MVC, controller nhận request từ người dùng, xử lý request, xây dựng dữ liệu cho view(model) và chọn view để trả về kết quả cho người dùng

1. **Trình bày ý nghĩa của ModelAndView Interface?**

* Là sự kết hợp giữa Model và View, giúp chúng ta có thể gửi model và view chỉ trong 1 dòng lệnh

1. **ModelMap interface là gì?**

* Là đối tượng implement của Map interface, chúng ta có thể dùng put để đưa dữ liệu vào, ngoài ra nó còn hỗ trợ một số phương thức khác để đưa dữ liệu vào.

1. **Trình bày ý nghĩa của ViewResolver interface?**

* Là các đối tượng chịu trách nhiệm lấy các đối tượng View cho một hành động và vị trí cụ thể. Thông thường, Controller yêu cầu ViewResolvers chuyển tiếp đến một view với một tên cụ thể(một chuỗi được trả về phương thức). Sau đó tất cả ViewResolvers trong ứng dụng thực thi theo thứ tự cho đến khi một trong số đó phân giải view trong trường hợp một đối tượng View trả về và điều khiển được truyền tới nó cho việc render HTML.

1. **Phân biệt POST và PUT thường để làm gì?**

|  |  |
| --- | --- |
| **POST** | **PUT** |
| * Unsafe: có thể thay đổi trạng thái của server, nó có thể chỉnh sửa, tạo mới, xóa thông tin | |
| * POST: gửi thông tin đến server thông qua form * POST: ko idempotent, tức là khi thực hiện cùng 1 nội dung giống nhau 2 lần thì sẽ bị lỗi | * PUT: ghi đè tất cả thông tin của đối tượng gửi lên * PUT là imdempotent |

1. **@RequestMapping để làm gì?**

* @RequestMapping để map tới class hoặc method xử lý request đó.

1. **Trình bày cơ chế DataBinding trong Spring?**

* DataBingding là cơ chế liên kết đối tượng đầu vào(hoặc đầu ra) với các đối tượng model. Sử dụng Databinding các form đều được liên kết với các đối tượng phía sau dữ liệu
* Databinding hỗ trợ chuyển đổi dữ liệu(data conversion) và validate dữ liệu

1. Thuộc tính consumes trong các RequestMapping là gì?

* Chỉ chấp nhận các request có kiểu dữ liệu được khai báo trong consumes

1. Thymeleaf là gì?

* Thymeleaf là 1 Java template engine dùng để tạo và xử lý Html, CSS, XML, text và JavaScript.

Lợi ích của Thymeleaf:

* Ta có thể sử dụng file HTML để hiển thị tất cả mọi thứ
* Chỉ sử dụng các thẻ HTML thuần
* Vì là các thẻ HTML nên có thể xem file view mà ko cần khởi chạy server
* Thymeleaf hỗ trợ cơ chế cache, do đó có thể cache dữ liệu hoặc custom dữ liệu mà ko cần phải khởi chạy lại server

|  |  |
| --- | --- |
| Thymeleaf | JSP |
| * Sử dụng thẻ HTML thuần | * Nhúng code Java, hiển thị bằng JSTL |

1. Sử dụng lặp trong Thymeleaf thế nào? Sử dụng điều kiện trong Thymeleaf như thế nào?

* Vòng lặp: th:each, chấp nhận các loại dữ liệu như Iterable, Map, Arrays, thường sử dụng kèm thẻ th:block
* Điều kiện: th:if, th:unless(ngược lại với th:if)

Một biểu thức hoặc biến được đánh giá là false nếu giá trị của nó là null, false, 0, “false”, “off”, “no”.

* Th:swich, th:case: tương tự như trong Java

1. Formatter là gì? Converter là gì?

* Formatter và Converter dùng để chuyển đổi kiểu dữ liệu

|  |  |
| --- | --- |
| Formatter | Converter |
| * Kiểu dữ liệu nguồn String * Phù hợp để sử dụng ở tầng web | * Có thể chuyển đổi từ bất kì kiểu dữ liệu nào * Sử dụng được trong mọi tầng của ứng dụng web |

1. ORM là gì?

* ORM(Object Relational Mapping) là một kĩ thuật/cơ chế ánh xạ CSDL sang các đối tượng trong ngôn ngữ lập trình hướng đối tượng(các table tương ứng với các class)
* Ưu điểm:

+ ORM giúp LTV tập trung vào lập trình hướng đối tượng

+ Tính độc lập: làm việc với nhiều loại database, nhiều kiểu DL khác nhau. Dễ dàng thay đổi loại database hơn. Các câu lệnh SQL ko phụ thuộc vào loại database.

+ Đơn giản, dễ sử dụng: hỗ trợ HSQL, cung cấp nhiều API truy vấn.

+ Năng suất hơn: viết code ít hơn, dễ hiểu hơn. Phù hợp các case CRUD

+ Tái sử dụng code

* Nhược điểm:

+ Khả năng truy vấn bị hạn chế, nhiều trường hợp phải dùng native SQL để truy vấn database.

+ Khó tối ưu câu lệnh SQL(do câu lệnh SQL được ORM tự động sinh ra)

1. JPA là gì? Spring Data JPA là gì?

* JPA(Java Persistence API) là 1 giao diện lập trình ứng dụng Java, nó mô tả cách quản lý các mối quan hệ dữ liệu trong ứng dụng sử dụng Java platform, JPA cung cấp mô hình POJO persistence cho phép ánh xạ các table/mối quan hệ giữa các table trong database sang các class/mối quan hệ giữa các object.
* Entity: là các đối tượng thê hiện tương ứng 1 table trong CSDL. Khi lập trình, entity thường là các class POJO đơn giản.
* EntityManager: là 1 interface cung cấp các API cho việc tương tác với các Entity như Persist(lưu), merge(cập nhật), remove(xóa).
* EntityManagerFactory: dùng để tạo 1 thể hiện của EntityManager.
* Spring Data JPA: là 1 module trong Spring, được xây dựng dựa trên JPA giúp cho việc triển khai ứng dụng dễ dàng hơn bằng việc cung cấp nhiều chức năng cơ bản.

1. Hibernate là gì? Hibernate làm gì với database?

* Hibernate là 1 ORM framework: thực hiện mapping cơ sở dữ liệu quan hệ sang các object trong lập trình hướng đối tượng
* Hibernate giúp lưu trữ và truy vấn dữ liệu quan hệ mạnh mẽ và nhanh. Hibernate cho phép ban truy vấn dữ liệu thông qua JPA hoặc native SQL hoặc HQL

1. Một số annotation của Hibernate?

* @Entity: để chú thích một class là một Entity
* @Table: chú thích tên bảng thông qua thuộc tính name
* @Column: chỉ định thông tin chi tiết của cột mà một field của entity sẽ được ánh xạ với 1 column trong database
* @Transient: thông báo thuộc tính/phương thức này ko liên quan tới một cột nào dưới database
* @Temporal: chú thích cho cột dữ liệu ngày tháng và thời gian

+ TemporalType.DATE: ngày tháng năm

+ TemporalType.TIME: giờ phút giây

+ TemporalType.TIMESTAMP: cả ngày tháng và thời gian

* @Id: khóa chính của table trong database
* @GeneratedValue: tự động tạo giá trị và gán vào cột
* UUID: là một lớp của Java cho phép bạn tạo ra một chuỗi 36 kí tự ngẫu nhiên
* @Lob: thường chú thích cùng @Column để nói rằng cột đó có kiểu BLOB hoặc CLOB
* @ManyToOne, @OneToMany, @OneToOne, @ManyToMany
* @OrderBy: sắp xếp danh sách, thường sử dụng cùng @OneToMany, @ManyToMany

1. Entity là gì?

* Entity là một POJO để khai báo với Spring đó là 1 bean, phải có Id và ít nhất 1 contructor ko tham số

1. PersistenceContext và EntityManager là gì?

* Persistence Context là tập các thể hiện của entity được quản lý, tồn tại trong một kho dữ liệu
* Entity Manager là đối tượng quản lý các entity. Interface EntityManager thực hiện các công việc sau:

+ Khai báo các phương thức để tương tác với persistence context

+ Tạo hoặc xóa các thể hiện của entity

+ Tìm kiếm entity theo khóa chính

+ Thực thi các câu lệnh truy vấn lên entity

1. Câu lệnh truy vấn động là gì? Câu lệnh truy vấn tĩnh là gì?

* SQL động có thể truyền tham số trong lúc thực thi câu lệnh
* SQL tính ko thể truyền tham số trong lúc thực thi câu lệnh

1. Trạng thái của Entity bao gồm những gì?

Có 4 trạng thái:

* New(mới tạo ra, chưa lưu xuống database): object vừa mới được tạo ra mà chưa được kết nối với Hibernate và cũng chưa được ánh xạ trong bất kì bảng nào của database🡪Persistence(method persist() của EntityManager hoặc cơ chế bắc cầu)
* Persistent(đc ghi xuống database và quản lí trong Persistence Context):
* Detached(tạm thời tách ra khỏi Persistence Context)
* Removed()

1. Có bao nhiêu loại mapping trong Hibernate?

Có 3 loại mapping quan trọng:

* Collections Mappings: nếu entity hoặc lớp có một tập hợp trong một biến Collection cụ thể thì chúng ta có thể ánh xạ các giá trị đó bằng bất kì một trong các interface collections có sẵn trong Java
* Association Mappings: ánh xạ liên kết giữa các lớp entity và các mối quan hệ giữa các bảng: OneToOne, OneToMany,…
* Component Mappings: một Entity có thể tham chiếu đến một lớp khác như 1 biến thành viên

1. SessionFactory là gì?

* Là 1 interface giúp tạo ra session kết nối đến database bằng cách đọc cấu hình trong Hibernate configuration
* SessionFactory thường được tạo ra trong quá trình khởi động ứng dụng và lưu trữ để sử dụng sau này
* SessionFactory là 1 đối tượng luồng an toàn(Thread-safe) và được sử dụng bởi tất cả các luồng của một ứng dụng
* Mỗi một database phải có 1 session factory

1. Valition dữ liệu là gì?

* Là việc kiểm tra dữ liệu trước khi xử lý nhằm đảm bảo dữ liệu đáp ứng các yêu cầu đặt ra

1. Trình bày cách triển khai validate dữ liệu trong Spring?

Validation có thể thực hiện ở các tầng khác nhau của ứng dụng:

* Tầng giao diện: sử dụng JS
* Tầng back-end: validate ở controller/service
* Tầng CSDL: sử dụng rang buộc trong CSDL như notnull, unique,…

Có 2 cách để validate trong Spring: triển khai interface Validators và sử dụng Annotation

1. Binding Result là gì?

* Binding Result là nơi chứa các lỗi khi người dùng nhập dữ liệu vào

1. AOP là gì? AOP gồm những yếu tố nào?

* AOP(Aspect Oriented Programming) lập trình hướng khía cạnh là một kĩ thuật lập trình phân tách chương trình thành những module riêng, ko phụ thuộc nhau khi hoạt động sẽ kết hợp các module lại với nhau để thực hiện các chức năng nhưng khi sửa đổi 1 chức năng thì chỉ cần sửa 1 module
* AOP ko phải dùng để thay thế OOP mà để bổ sung cho OOP.
* Các yếu tố trong AOP:

+ Core concern: các method chính của chương trình

+ Crosscutting concerns: nhưng chức năng khác của chức trình(như ghi log)

+ Join points: một điểm của chương trình để chèn crosscutting concern

+ Poincut: cách để xác định join points

+ Advice: những xử lý phụ được thêm vào xử lí chính

* Có 4 loại advice:

+ Before advice: chạy trước khi method được thực thi

+ After returning advice: chạy sau khi method trả về 1 kết quả

+ After throwing advice: chạy khi method ném ra một exception

+ Around advice: chạy khi method đc thực thi(bao gồm cả 3 loại advice trên)

37. Nêu cách triển khai AOP mà bạn dùng trong dự án?

- Khai báo @Aspect trên class crosscutting-corcern

- @After, @AftherThrowing,.. trên phương thức

- Xác định joinpoint

1. Xử lý ngoại lệ trong Spring web MVC như thế nào?

* Sử dụng @ExceptionHandler kèm theo tên của Exception được ném ra

1. Session là gì? Cookie là gì? Phân biệt session và cookie?

* Session: là một phiên làm việc được tạo ra và lưu trữ trên server ghi lại hoạt động của người dùng
* Cookie: là một file ghi thông tin được server tạo ra và lưu trữ trên trình duyệt máy người dùng

|  |  |
| --- | --- |
| Cookie | Session |
| * Lưu trữ trên trình duyệt người dùng * Tính bảo mật thấp * Có sẵn trong trình duyệt cho đến khi hết hạn * Dung lượng thấp khoảng 4kB | * Lưu trữ tại server * Có tính bảo mật cao * Mất khi đóng trình duyệt hoặc hết hạn * Session có sessionID được lưu trữ trên cookie người dùng |

1. Nêu cách triên khai Session bạn dùng trong dự án?

* Sử dụng SessionAttributes, SessionAttribute và ModelAttribute

1. Webservice là gì? Ví dụ

* Webservice sử dụng để tích hợp các ứng dụng được viết bằng ngôn ngữ khác nhau, chạy trên nhiều nền tảng khác nhau. Xây dựng ra các chuẩn mở và sử dụng các giao thức để giao tiếp với nhau
* Hoạt động theo mô hình client-server
* DL đầu ra: JSON và XML
* Ưu điểm:

+ Hoạt động trên nhiều nền tảng, ngôn ngữ, hệ điều hành khác nhau

+ Tái sử dụng cao

+ Mối quan hệ tương tác mềm dẻo trong hệ thống phần mềm

+ Giảm sự phức tạp của hệt thống

+ Giảm thời gian phát triển, chi phí

* Nhược điểm:

+ Khi service chết thì toàn hệ thống bị ảnh hưởng

+ Phải chú ý đến tính bảo mật

1. Phân biệt Website và Webservice?

|  |  |
| --- | --- |
| Website | Webservice |
| * Có giao diện | * Ko có giao diện |
| * Người dùng sử dung | * Ứng dụng sử dụng |
| * Hoạt động đa nền tảng | * Độc lập nền tảng và sử dụng giao thức chuẩn |
| * Sử dụng button, form, textarea | * PUT, GET, POST, DELETE,.. |

1. SOAP và RESTful là gì?

* SOAP(Simple Object Access Protocol) là 1 giao thức dựa trên XML để truy cập Webservice
* Ưu điểm:

+ WS Security: định nghĩa bảo mật riêng của nó gọi là WS security

+ Có thể được viết bằng bất cứ ngôn ngữ nào và thực thi bất kì nền tảng nào

* Nhược điểm:

+ SOAP sử dụng định dàng phải được phân tích cú pháp, các ứng dụng SOAP phải tuân theo nhiều tiêu chuẩn. Vì vậy nó chậm và chiếm nhiều bang thông

+ Phụ thuộc WSDL

* REST(Representational State Transfer): là 1 kiểu kiến trúc ko phải 1 giao thức
* Ưu điểm:

+ Nhanh: nhanh hơn SOAP vì ko có nhiều tiêu chuẩn và chiếm ít bang thông và tài nguyên hơn

+ Viết bằng mọi ngôn ngữ, chạy mọi nền tảng

+ Có thể sử dụng SOAP khi thực hiện

+ Cho phép nhiều định dạng: plain text, HTML, XML, JSON

1. jQuery là gì? Ajax là gì?

* jQuery là 1 thư viện của JS, giúp xây dựng các tính năng JS dễ dàng, nhanh, tiện lợi
* Ajax(Asynchoronous JavaScript and XML) là một nhóm các công nghệ cho phép load dữ liệu mà ko cần tải lại trang

1. @RequestHeader và @ResponseHeader là gì?
2. I18n và L10n là gì?

* I18n(quốc tế hóa): là 1 kĩ thuật cho phép chúng ta tạo ra các ứng dụng thích ứng với nhiều ngôn ngữ và khu vực khác nhau
* L10n(nội địa hóa): là 1 kĩ thuật giúp tạo ra ứng dụng tương thích với 1 vùng miền cụ thể

1. Nêu cách triển khai I18n và L10n trong Spring?

B1: Thêm thư viện Spring boot

B2: Cấu hình LocaleResolver(thiết lập ngôn ngữ mặc định)

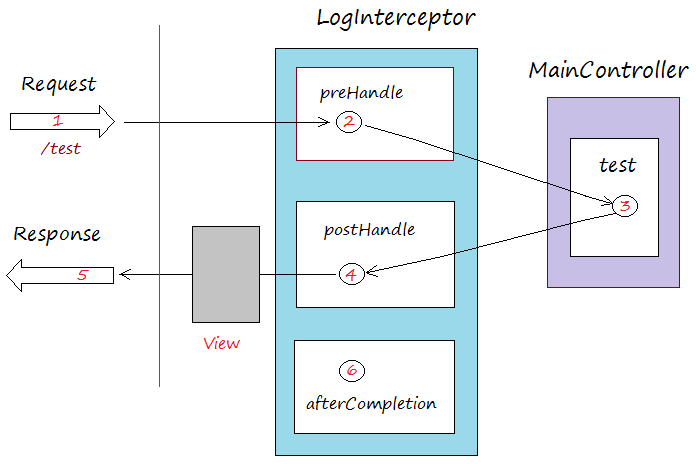
B3: Cấu hình LocaleChangeInterceptor: chuyển đổi qua lại giữa các ngôn ngữ

B4: Định nghĩa nội dung trong file Message Sources(file message\_XX.properties)

B5: Tạo Controller và trang HTML

1. Interceptor là gì?

* Trong Spring, khi 1 request được gửi đến controller, trước khi request được xử lý bởi Controller nó phải vượt qua các Interceptor(0 hoặc nhiều)
* Bạn có thể sử dụng Interceptor để làm một số việc, chẳng hạn như ghi Log, thêm hoặc cập nhật cấu hình trước khi request được xử lí bởi Controller,…



Các bước triển khai Interceptor:

* Khai báo interceptor, mapping, exclude
* Implement HandlerInterceptor: gồm có 3 phương thức: preHandle, postHandle, afterCompletion

1. Bảo mật là gì? Cơ chế bảo mật trong Spring?

* Bảo mật là bảo vệ thông tin, dữ liệu cá nhân, tổ chức nhằm tránh khỏi sự đánh cắp bởi tin tặc
* Spring Security thực sự chỉ là một loạt các bộ lọc servlet giúp bạn thêm authentication và authorization vào ứng dụng web của mình

1. Authentication và Authorization là gì?

* Authentication: xác thực xem có phải là người dùng của hệ thống hay ko
* Authorization: phân quyền cho người dùng được phép truy cập vào tài nguyên nào của hệ thống

1. Liệt kê các cơ chế xác thực?

Có 6 cơ chế xác thực cơ bản:

* Http Basic: cung cấp username và password cho server khi gửi request
* Cookie: xác thực qua cookie của người dùng
* One-time password: mật khẩu 1 lần
* Oauth2: xác thực bằng bên thứ 3 như Google, Facebook, …
* Signature(chữ kí số): thiết bị được mã hóa tất cả dữ liệu, thông tin dùng thay cho chữ kí.
* Token: JWT JSON web tokens thường được sử dụng trong mô hình webservice

1. CSRF là gì?

* CSRF(Cross Site Request Forgery) là kĩ thuật tấn công bằng cách sử dụng quyền chứng thực của người dùng đối với 1 website để gửi đi các mã độc hoặc link đến website mà người dùng đã xác thực
* Phòng tránh CSRF:

+ Phía user: thoát khỏi các website khi thực hiện xong giao dịch, ko lưu mật khẩu

1. CORS là gì?

* CORS(Cross Origin Resource Sharing) là 1 tính năng mới được tích hợp trong HTML5, thêm vào các HTTP headers chỉ dẫn cho trình duyệt web sử dụng và quản lý nội dung cross-domain, cho phép lấy dữ liệu từ một trang khác thông qua XMLHttpRequest.
* Khi trình duyệt gửi 1 request đến 1 domain khác để yêu cầu làm 1 việc gì đó, những request này sẽ được gắn thêm 1 header có tên là origin để xác định là origin của client, giá trị này được thêm tự động bời trình duyệt và ko ai có thể thay đổi được. Header này đại diện cho nguồn gốc truy vấn
* Origin được cấu tạo dựa trên 3 phần:

+ Protocol: Http/Https

+ Host: server/domain

+ Port: cổng, mặc định là 80 thì ko cần

* Nếu origin hợp lệ, response trả về header Access-Control-Allow-Origin

1. Spring boot là gì?

* Spring boot là 1 module trong Spring framework, giúp tạo dự án mà ko phải cấu hình quá nhiều

1. Sự khác nhau Spring boot và Spring MVC?

|  |  |
| --- | --- |
| Spring Boot | Spring MVC |
| * 1 module trong Spring * Cấu hình ít * Giảm thời gian phát triển, tang năng suất | * 1 mô hình Spring * Cần cầu hình nhiều * Tốn nhiều thời gian và năng suất |

1. Cấu hình cho Spring Boot để tìm kiếm các Bean ở nhiều package bằng cách nào?

Có 2 cách để cấu hình Spring boot tìm kiếm bean:

* Cách 1: @ComponentScan
* Cách 2: @SpringBootApplication(scanBasePackages = “…”)

1. Trình bày Spring Security? @EnableWebSecurity để làm gì?

* @EnableWebSecurity để kích hoạt tính năng Spring Security trên ứng dụng web của mình
* Spring Security là 1 module trong Spring framework, cung cấp dịch vụ bảo mật toàn diện thông qua 2 cơ chế cơ bản: authentication và authorization
* Các thành phần cốt lõi:

+ SecurityContext: là interface cốt lõi của Spring framework, lưu trữ tất cả các chi tiết liên quan đến bảo mật trong ứng dụng. Khi kích hoạt Spring Security thì SecurityContext được kích hoạt theo. SecurityContextHolder là class lưu trữ Spring Context hiện tại của class

+ UserDetail và UserDetailService: là interface đại điện cho người dùng đã xác thực. UserDetailService là triển khai của UserDetail chỉ có 1 phương thức duy nhất là loadUserbyUsername.

1. Spring Boot Jpa là gì? Cài đặt?

* Spring Boot JPA là 1 bảng ghi chi tiết của Java để quản lí dữ liệu quan hệ trong các ứng dụng Java. Nó cho phép chúng ta truy cập và lưu trữ dữ liệu giữa các object/class Java và database quan hệ. Nó cung cấp EntityManager để quản lí query và giao dịch trên database

1. Để sử dụng JpaRepository cần cung cấp những thông tin nào?

* JpaRepository thừa kế PagingAndSortingRepository và có thêm 1 số chức năng mở rộng

1. @ConfigurationProperties là gì?

* @ConfigurationProperties cho phép truy xuất các thuộc tính được đặt trong các tệp cấu hình một cách tự động và mạnh mẽ.