**Câu 1: Vẽ và trình bày cách Spring MVC xử lý request**

- Khi có 1 request đi vào Spring MVC thì FrontController sẽ đón nhận và sử dụng cơ chế URL handler Mapping để ánh xạ đến controller tương ứng

- Controller sẽ xử lí request, nếu cần sẽ dựa vào Model xử lý nghiệp vụ và thao tác dữ liệu Database, kết quả sẽ được gửi ngược lại cho Controller

- Controller sẽ render dữ liệu lên view theo cơ chế ViewRosolver

- Cuối cùng kết quả sẽ được gửi lại FrontController và phản hồi về cho người dùng

**Câu 2: Trình bày cơ chế Dependency Injection trong Spring**

DI là một Design Pattern cho phép xóa bỏ sự phụ thuộc

DI là một cơ chế để triển khai IoC

Các dependency được cung cấp và điều khiển bởi container hoặc framework

Thao tác tiêm vào các đối tượng khác được thực hiện với container hoặc framework

**Câu 3: Có bao nhiêu cách để thực hiện Dependency Injection? Trình bày?**

Trả lời :

Triển khai DI có 3 cách: Interface, Constructor, Setter

+ Contructor: Các dependency (biến phụ thuộc) được cung cấp thông qua constructor

+ Setter: Các dependency (biến phụ thuộc) sẽ được truyền vào 1 class thông qua các setter method

+ Interface: Các client phải triển khai một Interface mà có một setter method dành cho việc nhận dependency.

+ Interface: Các client phải triển khai một Interface mà có một setter method dành cho việc nhận dependency.

**Câu 4 Framework là gì ? Framework khác Library chỗ nào ?**

Framework là một phần mềm có tính trừu tượng cao, cung cấp các chức năng chung - thông dụng nhất và có khả năng tùy biến

Điểm khác biệt so với Library:

+ Library: lập trình viên sẽ nắm quyền điều khiển (Code của mình sẽ gọi Code thư viện)

+ Framework: Framework sẽ nắm quyền ứng dụng (Code của Framework sẽ gọi Code của mình)

+ Khi code với framework thì cần tuân theo những nguyên tắc có sẵn

=> Điểm khác biệt chính giữa Framework và Library là cơ chế IoC

**Câu 5 Spring Framework là gì ?**

Spring Framework là một framework dùng để phát triển các ứng dụng Java

Spring Framework được xây dựng dựa trên 2 nguyên tắc design chính là Dependency Injection và Aspect Oriendted Programming ( AOP )

**Câu 6 Lợi ích của Spring Framework?**

Nhẹ, giúp các ứng dụng có hiệu năng cao, dễ kiểm thử, tái sử dụng code,...

**Câu 7 Nguyên lý đảo ngược quyền điều khiển (Inversion of Control) là gì ?**

IoC là nguyên lý phát triển phần mềm. Trong đó, việc điều khiển các đối tượng hoặc thành phần của hệ thống sẽ được thực hiện bởi Framework hoặc các Container

**Câu 8 Bean là gì?**

Bean là đối tượng được Framework và Container quản lý

Trong Spring, Bean là một đối tượng được khởi tạo, lắp ráp và quản lý bởi Spring IoC Container

**Câu 9 Trong Spring có bao nhiêu Bean Scope?**

Trong Spring có 5 Bean Scope: singleton, prototype, request, session, global session

+ Singleton: Chỉ duy nhất một thể hiện của bean sẽ được tạo cho mỗi container. Đây là scope mặc định

cho spring bean.

+ Prototype: Một thể hiện của bean sẽ được tạo cho mỗi lần được yêu cầu(request)

+ Request: giống với prototype scope, tuy nhiên nó dùng cho ứng dụng web, một thể hiện của bean sẽ được tạo cho mỗi HTTP request

+ Session: Mỗi thể hiện của bean sẽ được tạo cho mỗi HTTP Session

+ Global-Session: Được sử dụng để tạo global sesion bean cho các ứng dụng Portlet

|  |
| --- |
| **Câu 10:@Autowire là gì?** |
| @Autowire đánh dấu cho Spring biết rằng sẽ tự động inject bean tương ứng vào vị trí được đánh dấu   |  | | --- | | **Câu 11:@Component có ý nghĩa gì?** | | @Component dùng để đánh dấu một class Java là mộtBean | |
| **Câu 12: Trình bày ý nghĩa của Controller** |
| Controller là file class, xử lý nghiệp vụ, logic |
| **Câu 13: Trình bày ý nghĩa của ModelAndView Interface** |
| ModelAndView Interface là sự kết hợp của ModelMap và View. Nó cho phép Controller trả về một giá trị bao gồm Model và View |
| **Câu 14:Trình bày ý nghĩa của ModelMap** |
| ModelMap cho phép truyền dữ liệu từ Controller sang View để hiển thị. ModelMap có các tính chất của Map |
| **Câu 15: Trình bày ý nghĩa của ViewResolver Interface**  ViewResolver Interface cho phép ánh xạ "tên" của View sang đối tượng của View tương ứng  Câu 16 chưa xoạn  **Câu 17: Phân biệt POST với PUT thường sử dụng để làm gì?**  POST: thêm mới dữ liệu  PUT: chỉnh sửa dữ liệu  **Câu 18: @RequestMapping làm gì?**  @RequestMapping được dùng để ánh xạ các request tới Controller tương ứng  **Câu 19: Trình bày cơ chế Data Binding trong Spring**  Data Binding là cơ chế liên kết dữ liệu đầu vào và đầu ra với một đối tượng Model  + Các form đều được liên kết với một đối tượng biểu diễn dữ liệu phía sau  => Dữ liệu trên form sẽ được tự động chuyển đổi thành các thuộc tính của đối tượng liên kết với form  + Data binding hỗ trợ chuyển đổi dữ liệu và validate  + Data binding hoạt động dựa trên binder  **Câu 19: Thuộc tính consumes trong các Request Mapping là gì ?**  Thuộc tính consumes được dùng để định dạng dữ liệu khi client request lên server |
| **Câu 20: Thuộc tính consumes trong các Request Mapping là gì ?**  Thuộc tính consumes được dùng để định dạng dữ liệu khi client request lên server  **Câu 21: Thymeleaf là gì?**  Thymeleaf là một bộ xử lý view, được sử dụng cho các ứng dụng web và các ứng dụng độc lập  Được xây dựng phù hợp với tiêu chuẩn web, hỗ trợ HTML5  Thymeleaf cho phép xử lý các loại templates HTML, TEXT, XML, JAVASCRIPT, CSS,...   |  | | --- | | - Ưu điểm: | | + Thân thiện, gần gũi, dễ sử dụng, vì sử dụng cú pháp HTML5 | | + Cú pháp của thymeleaf là một atribute ( thuộc tính ) của thẻ HTML và bắt đầu bằng th: | |

**Câu 22: Sử dụng lặp trong Thymeleaf như thế nào? Sử dụng điều kiện trong Thymeleaf như thế nào?**

th:each="loop : ${loopList}";

th:if="${condition}";

Câu 23: chưa xoạn bài

**Câu 24: ORM là gì ?**

ORM (Object Relational Mapping) là kỹ thuật liên kết đối tượng trong lập trình và đối tượng trong CSDL

=> Là quá trình biến đổi giữa một record trong Database và một Object trong ngôn ngữ lập trình

- Một số ORM Framework

+ Hibernate

+ EclipseLink

+ Ibatic

- ORM có rất nhiều implementation, trong đó có hibernate xem là ORM framework nổi tiếng nhất và sử dụng rộng rãi trong java

**Câu 25: JPA là gì? Spring Data JPA là gì ?**

|  |
| --- |
| * JPA (Java Persistence API) là một bộ đặc tả tiêu chuẩn của Java, trong đó mô tả các thao tác quản lý dữ liệu quan hệ trong ứng dụng Java. Cung cấp mô hình cho phép ánh xạ các bảng trong CSDL với các lớp |
| * Spring Data JPA là một module của Spring Framework giúp cải tiến JPA tiêu chuẩn, đơn giản hóa tầng truy xuất dữ liệu, tự tạo ra Repository, tạo các truy vấn JPA thông qua tên phương thức |

**Câu 26: Hibernate là gì ? Hibernate làm gì với database**

|  |
| --- |
| * Hibernate là một thư viện ORM mã nguồn mở giúp lập trình viên viết ứng dụng Java có thể map các Object với hệ CSDL, và hỗ trợ thực hiện các khái niệm OOP với CSDL |
| * Hibernate có thể hỗ trợ tạo bảng và các dữ liệu thông qua ngôn ngữ Java |

**Câu 27: Liệt kê một số annotation của hibernate?**

@Entity, @Table, @Id, @GenerateValue, @Column, @OneToOne, @OneToMany, @ManyToOne, @ManyToMany

**Câu 28: Entity là gì?**

|  |
| --- |
| Entity là đối tượng đại diện cho dữ liệu trong ứng dụng |
| Mỗi Entity sẽ được mapping với một table trong CSDL |
| Mỗi Entity cần tuân thủ: |
| + Được gắn annotation @Entity |
| + Có constructor public rỗng |
| + Không Được khai báo final |
| + Thuộc tính của đối tượng Có access modifier là private, protected hoặc ở mức package private |

**Câu 29 : Persistence Context & Entity Manager là gì?**

|  |
| --- |
| Persistence Context là một tập các thể hiện của Entity, được dùng quản lý Entity trong một kho dữ liệu |
| Entity Manager là một interface cung cấp các phương thức cho việc tương tác với các Entity |
| Các phương thức thông dụng: |
| + persit |
| + merger |
| + remove |
| + find |

**Câu 30 : Câu lệnh truy vấn động là gì? Câu lệnh truy vấn tĩnh là gì?**

|  |
| --- |
| Câu lệnh truy vấn động là câu truy vấn có các tham số thay đổi |
| Câu lệnh truy vấn tĩnh là câu truy vấn có các tham số cố định |

**Câu 31 : Trạng thái của Entity bao gồm những gì?**

|  |  |
| --- | --- |
| + New: tạo mới, chưa được lưu xuống CSDL | |
| + Persistent: Được ghi xuống CSDL và quản lý trong Persistence Context | |
| + Detached: tạm thời tách ra khỏi Persistence Context | |
| + Removed: xóa những Detached Entity | |
| **Câu 32 : Có bao nhiêu loại mapping trong Hibernate?**  Có 3 loại mapping trong Hibernate: |
| - Entity Mapping |
| + OneToOne |
| + OneToMany |
| + ManyToOne |
| + ManyToMany |
| - Collection Mapping: Set, SortedSet, List, Collection, Map, SortedMap |
| - Component Mapping |

**Câu 33 : SessionFactory là gì?**

Là một interface giúp tạo ra Session kết nối đến Databate bằng cách đọc các cấu hình trong Hibernate Configuration

**Câu 34 : Validation dữ liệu là gì?**

Validation là thực hiện đánh giá/xác minh tính hợp lệ của dữ liệu đầu vào

**Câu 35 : Trình bày cách triển khai validate dữ liệu trong Spring**

|  |
| --- |
| 1. Implement interface Validator vào model cần Validate |
| 2. Overide các phương thức của Interface theo hình thức regex dữ liệu |
| 3. Sử dụng các annotation Validate có sẵn: @NotNull, @NotEmpty, @NotBlank,… |
| 4. Sử dụng @Validated, @Valid, BindingResult ở Controller để thực hiện trả về kết quả khi có lỗi |

**Câu 36 : Binding Result là gì?**

Binding Result là một đối tượng chứa các lỗi trong quá trình Validate

**Câu 37 : AOP là gì? AOP gồm những yếu tố nào?**

|  |
| --- |
| AOP (Aspect Oriented Programming) là lập trình hướng khía cạnh, là các đoạn chương trình độc lập với nghiệp vụ dự án (bổ trợ OOP) |
| AOP gồm các yếu tố: |
| + Aspect: dùng để khai báo cho các đoạn chương trình độc lập, không liên quan gì đến nghiệp vụ dự án |
| + Advice: là các hành động mà Aspect dùng để quan sát OOP |
| + Pointcut: là tập hợp các Joinpoint có khả năng xảy ra các hoạt động Advice |
| + Joinpoint: những điểm xảy ra Advice |

**Câu 38: Các loại Advice?**

|  |
| --- |
| Có 5 loại: |
| + Before: hoạt động trước khi Joinpoint thực thi |
| + AfterReturning: hoạt động sau khi Joinpoint thực thi và trả về giá trị |
| + AfterThrowing: hoạt động sau khi Joinpoint thực thi và ném ngoại lệ |
| + After: hoạt động khi Joinpoint hoàn thành |
| + AroundAdvice: thực thi cả trước và sau Joinpoint |

**Câu 39: JoinPoint là gì?**

Joinpoint (điểm gia nhập ): một vị trí nào đó trong luồn thực thi của chương trình mà advice sẽ được đan ( weave ) vào chương trình chính

**Câu 40: Nêu cách triển khai AOP mà bạn dùng trong dự án**

|  |
| --- |
| 1. Khai báo class AOP và đánh dấu bằng annotation @Aspect, @Component |
| 2. Khai báo cách Advice với các annotation: @Before, @AfterReturning, @AfterThrowing, @After, @Around |
| 3. Khai báo Pointcut bằng từ khóa "excution" |
| 4. Viết phương thức Advice khi Joinpoint xảy ra |

**Câu 41: Xử lý ngoại lệ trong Spring Web MVC như thế nào?**

Sử dụng annotion @ExceptionHandler hoặc Try/Catch để xử lý các ngoại lệ

**Câu 42: Cookie là gì?**

Cookie là những tệp được trang web người dùng truy cập tạo ra. Cookie giúp trải nghiệm trực tuyến dễ dàng hơn

**Câu 43: Phân biệt Session và Cookie**

Session :Session chứa thông tin người dùng và được lưu trữ ở Server

* Dữ liệu Session khó sửa đổi
* Một Session kết thúc khi đóng trình duyệt
* Một Session phụ thuộc vào Cookie (JSESSIONID)

Cookie :Cookie lưu thông tin người dùng ở phía client

* Dữ liệu Cookie dễ sửa đổi hoặc dễ bị đánh cắp
* Kết thúc tùy thuộc vào thời gian cài đặt (setMaxAge())
* Một Cookie không phụ thuộc vào Session

**Câu 44 : Nêu cách triển khai Session mà bạn dùng trong dự án**

|  |
| --- |
| 1. Khởi tạo đối tượng Session cùng annotation @ModelAttribute |
| 2. Dùng annotation @SessionAttributes để lưu trữ thông tin của Model Attribute |
| 3. Sử dụng annotation @CookieValue để ràng buộc giá trị của cookie HTTP với tham số phương thức trong Controller |
| 4. Tạo Cookie và trả về cho Client |

**Câu 45 : Web Service là gì? Lấy ví dụ về web service**

|  |
| --- |
| * Web Service là một dịch vụ web cung cấp nền tảng chung cho nhiều ứng dụng. Những ứng dụng này có thể được viết bằng nhiều nền tảng công nghệ khác nhau nhưng có thể giao tiếp với nhau được |
| * Hoạt động dựa trên mô hình Client-Server, thông qua giao thức HTTP, HTTPs |
| * Kiểu dữ liệu dùng để giao tiếp là XML và JSON |

**Câu 46 : Phân biệt Web Service và Web truyền thông**

Web service

* Không có giao diện
* Kết quả trả về chỉ có data (theo format XML hoặc JSON) => User khó đọc
* Tương tác giữa các ứng dụng thông qua các phương thức HTTP: GET, POST, PUT, DELETE,... và HTTP code
* Thông thường chỉ có các ứng dụng giao tiếp với nhau trong môi trường Internet

Web application

* Có giao diện
* Kết quả trả về cho người dùng là view
* Tương tác giữa User và ứng dụng thông qua form, link, image, button,…
* Thông thường được tương tác giữa User & ứng dụng

**Câu 47 : SOAP là gì? RESTful là gì?**

|  |
| --- |
| * SOAP (Simple Object Access Portal) là giao thức Internet cho phép các chương trình trong các hệ điều hành khác nhau hoặc trên các máy chủ riêng biệt liên kết với nhau qua Internet. SOAP sử dụng giao thức XML |
| * RESTful (Representational State Transfer) là một tiêu chuẩn trong thiết kế API cho các ứng dụng web để quản lý resource. |

**Câu 48 : jQuery là gì? Ajax là gì?**

|  |
| --- |
| * jQuery là một thư viện được xây dựng từ JavaScript |
| * Ajax (Asynchronous JavaScript & XML) là một kỹ thuật giúp phát triển các trang web động mà không cần tải lại trang |

**Câu 49 : Nêu ví dụ về cách bạn triển khai Ajax trong dự án của bạn**

|  |
| --- |
| 1. Sử dụng jQuery $(document).ready để thông báo sử dụng jQuery khi HTML load xong |
| 2. Dùng jQuery để bắt sự kiện |
| 3. Dùng $.ajax để triển khai Ajax thông qua các thuộc tính: url, dataType, type, success |

**Câu 50**

**Câu 51 : I18n và L10n là gì?**

I18n (Internationalization): Quốc tế hóa

L10n (Localization): Nội địa hóa

**Câu 52 : Nêu cách triển khai I18N và I10N trong dự án Spring**

1. Cấu hình Java: LocalResolver, LocalChangeInterceptor, InterceptorRegistry, MessageSource

2. Tạo file Resource Bundle để khai báo message bằng các ngôn ngữ khác nhau

3. Sử dụng Thymeleaf để gán các message vào HTML

**Câu 53 : Interceptor là gì?**

Interceptor được xem như là một chốt chặn giữa các request gửi đến Controller

**Câu 54 : Bảo mật là gì? Cơ chế bảo mật trong Spring như thế nào?**

|  |
| --- |
| Bảo mật là sự hạn chế khả năng lạm dụng tài nguyên và tài sản |
| Cơ chế bảo mật trong Spring: |
| + Authentication (xác thực): là một hành động nhằm thiết lập hoặc chứng thực một thông điệp hoặc đối tượng nào đó đáng tin cậy |
| + Authorization (phân quyền): là quá trình xác định xem một đối tượng có quyền truy cập một tài nguyên cụ thể để thực hiện một số hành động hay không |

**Câu 55 : Authentication là gì ? Authorization là gì ?**

|  |
| --- |
| + Authentication (xác thực): là một hành động nhằm thiết lập hoặc chứng thực một thông điệp hoặc đối tượng nào đó đáng tin cậy |
| + Authorization (phân quyền): là quá trình xác định xem một đối tượng có quyền truy cập một tài nguyên cụ thể để thực hiện một số hành động hay không |

**Câu 56 : Liệt kê các cơ chế xác thực**

|  |
| --- |
| + HTTP Basic: yêu cầu phía Client cung cấp Username & Password để xác thực |
| + Cookies |
| + Tockens: là một tiêu chuẩn mở định nghĩa cách truyền tin an toàn giữa các thành viên bằng một đối tượng JSON |
| + Signature |
| + One-time Password (OTP) |
| + OAuth2: xác thực thông qua ứng dụng thứ 3 |

**Câu 57 : CSRF là gì**

CSRF (Cross-site Request Forgery) là một kỹ thuật tấn công giả mạo chính chủ thể của nó

------------------------------------------------------------------------------------------------------------

**Câu 58 : CORS là gì**

CORS (Cross-origin Resource Sharing) là một cơ chế cho phép nhiều tài nguyên khác nhau của một trang web có thể truy cấp từ một domain khác với domain của trang

**Câu 59 : Spring Boot là gì?**

|  |
| --- |
| 1. Spring Boot là một trong số các module của Spring framework, cung cấp tính năng RAD (Rapid Application Development). |
| 1. Spring Boot được sử dụng để tạo ra một ứng dụng độc lập dựa trên Spring mà có thể chạy ngay với rất ít cấu hình cần thêm. |
| 1. Spring Boot không cần đến bất kỳ cấu hình xml nào, mà nó sử dụng quy ước về mô hình thiết kế phần mềm theo cấu hình, có nghĩa là nó giảm tải công việc cho các lập trình viên. |

**Câu 60 : Sự khác nhau giữa SpringBoot và SpringMVC là gì?**

SpringBoot

1. Spring Boot là một module của Spring Framework để đóng gói ứng dụng dựa trên Spring với các giá trị mặc định hợp lý
2. Nó cung cấp các cấu hình mặc định để xây dựng Spring-powered Framework
3. Không có yêu cầu với các file mô tả khi triển khai
4. Nó tránh mã viết sẵn và kết hợp các dependencies lại với nhau trong một đơn vị duy nhất

SpringMVC

1. Spring MVC dựa trên Spring Framework
2. Nó cung cấp các tính năng có sẵn để xây dựng một ứng dụng web
3. Cần có nhiều file mô tả triển khai
4. Nó chỉ định từng dependency riêng biệt
5. Cần nhiều thời gian để phát triển

Câu 61 : Cấu hình cho Spring Boot Tìm kiếm các Bean ở nhiều package khác nhau bằng cách nào?

**Câu 62 : Trình bày Spring Security? @EnableWebSecurity làm gì?**

|  |
| --- |
| Spring Security là một Framework của Spring, dành riêng cho việc thiết lập bảo mật ứng dụng bao gồm Authentication và Authorization |
| @EnableWebSecurity để kích hoạt SpringSecurity trong dự án |

Câu 63 :

**Câu 64 : Để sử dụng interface JpaRepository cần cung cấp những thông tin nào?**

Interface JpaRepository cần cung cấp thông tin: Đối tượng và Primary Key

**Câu 65 : Bạn có biết annotation @ConfigurationProperties? Trình bày?**

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