**CÂU HỎI AUDIT MODULE 4**

|  |  |
| --- | --- |
| **STT** | **Q&A** |
| 1 | **Q:** Vẽ và trình bày cách Spring MVC xử lý request  🡺 **A:**  - Request từ người dùng gửi tới ứng dụng web, DispatcherServlet sẽ là đối tượng nhận request. Điều hướng request tới handler phù hợp, các Controllers sẽ xử lý request. Chuẩn bị model và chọn view hiển thị. Trả về kết quả xử lý request cho DispatcherServlet. DispatcherServlet sẽ gọi View Template phù hợp để xử lý việc hiển thị trên giao diện bằng cách sử dụng model. View template trả kết quả về cho DispatcherServlet. Trả response về cho người dùng. |
| **2** | **Q:** Trình bày cơ chế Dependency Injection trong Spring  🡺 **A:**  - Các container hoặc framework tạo các đối tượng và cung cấp cho những nơi cần chúng |
| 3 | **Q:** Có bao nhiêu cách để thực hiện Dependency Injection? Trình bày?  🡺 **A:**  - Constructor-Based: Container sẽ gọi một constructor với các tham số, mỗi tham số đại diện cho một dependency  - Setter-Based: Container gọi các phương thức setter để khởi tạo bean  - Field-Based: Sử dụng annotation @Autowired để tiêm các dependency |
| 4 | **Q:** Framework là gì ? Framework khác Library chỗ nào ?  🡺 **A:**  - Là phần mềm có tính trừu tượng cao, giúp người dùng có thể xác định các ứng dụng 1 cách nhanh chóng, có thể tùy biến để tạo ra các ứng dụng khác nhau  - Đặc điểm khác biệt giữa Framework và Library đó là cơ chế IoC. Framework sẽ là người nắm quyền điều khiển của chương trình, còn đối với Library thì Library sẽ được gọi bởi code của người dùng và chịu sự điều khiển của luồng code đó |
| 5 | **Q:** Spring Framework là gì ?  🡺 **A:**  - Là 1 framework dùng để xác định các ứng dụng Web = ngôn ngữ Java hoặc Kotlin, thông thường được sử dụng cho các dự án lớn |
| 6 | **Q:** Lợi ích của Spring Framework?  🡺 **A:**  - Chỉ cần quan tâm đến những gì cần làm  - Dễ dàng kiểm tra và thử nghiệm.  - Cung cấp API giúp thuận tiện trong việc dịch các ngoại lệ  - Spring framework được xây dựng theo mô hình MVC nên các tinh năng thay thế các web framework khác đều được cung cấp đầy đủ |
| 7 | **Q:** Nguyên lý đảo ngược quyền điều khiển (Inversion of Control) là gì ?  🡺 **A:**  - Là một nguyên lý thiết kế trong công nghệ phần mềm trong đó các thành phần dựa vào nó để làm việc bị đảo ngược quyền điều khiển khi so sánh với lập trình hướng thủ thục truyền thống. |
| 8 | **Q:** Bean là gì?  🡺 **A:**  - Là các đối tượng tạo thành xương sống của ứng dụng và được quản lý bởi Spring IoC container. Một bean là một đối tượng được khởi tạo, lắp ráp, và được quản lý bởi một Spring IoC container. Các bean này được tạo ra bằng siêu dữ liệu cấu hình mà bạn cung cấp cho container |
| 9 | **Q:** Trong Spring có bao nhiêu Bean Scope?  🡺 **A:**  **-** Singleton:Là scope mặc định, chỉ duy nhất một thể hiện của bean sẽ được tạo cho mỗi container.  - 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. |
| 10 | **Q:** @Autowire là gì?  🡺 **A:**  - Là một annotation của Spring, nó đánh dấu rằng biến này sẽ được init bởi bean container của Spring nếu có phần tử tương ứng. |
| **11** | **Q:** @Component có ý nghĩa gì?  🡺 **A:**  - Được sử dụng để định nghĩa một khuôn mẫu chung cho toàn bộ dự án. Class được đánh dấu với annotation @Component khi dự án Spring được chạy nó sẽ quét qua và tạo 1 bean cho class đó. |
| 12 | **Q:** Trình bày ý nghĩa của Controller  🡺 **A:**  - Là một thành phần trong mô hình MVC dùng để điều hướng và xử lý request, gọi model để xử lý nghiệp vụ, kết nối dữ liệu và gửi dữ liệu lại cho view |
| 13 | **Q:** Trình bày ý nghĩa của ModelAndView Interface  🡺 **A:**  - Là sự kết hợp của 2 mục đích ModelMap and View. Nó cho phép controller trả về 1 giá trị bao gồm Model và View. |
| 14 | **Q:** Trình bày ý nghĩa của ModelMap  🡺 **A:**  - Được sử dụng để truyền dữ liệu từ Controller sang View để hiển thị. Có thể sử dụng ModelMap như một tham số trong method của Controller. ModelMap cài đặt lớp Map interface nên nó thêm các phương thức của Map. |
| 15 | **Q:** Trình bày ý nghĩa của ViewResolver Interface  🡺 **A:**  - Controller do người dùng định nghĩa trả về tên của view và các thông tin chi tiết khác tới Front Controller. Tên của view được trả về có kiểu là String, nên cần xử lý tiếp bởi ViewResolver. ViewResolvers có thể sử dụng để trả lại view. |
| 16 | **Q:** Phân biệt POST với PUT thường sử dụng để làm gì?  🡺 **A:**   |  |  | | --- | --- | | POST | PUT | | - Để tạo dữ liệu  - Khi ấn submit bao nhiêu lần đi chăng nữa thì dữ liệu cũng chỉ tính như submit 1 lần.  - Hệ thống tự tạo id cho dữ liệu | - Để cập nhật lại dữ liệu  - Khi submit quá nhiều lần thì có thể nó sẽ cho ra những kết quả khác nhau  - Khi tạo dữ liệu mới sẽ đi kèm theo với id để hệ thống sẽ tạo id của dữ liệu theo id đó dễ gây trùng lặp dẫn đến dữ liệu bị thay đổi | |
| 17 | **Q:** @RequestMapping làm gì?  🡺 **A:**  - Là annotation dùng để quy định cho class và method được sử dụng để ánh xạ đến các request. |
| 18 | **Q:** Thuộc tính consumes trong các Request Mapping là gì ?  🡺 **A:**  - consumes: chỉ chấp nhận các request có content-type giống với giá trị khai báo bên trong consumes. Có thể sử dụng đối tượng với Media Type được yêu cầu bằng cách sử dụng phần tử consumes của @RequestMapping kết hợp với annotation @RequestBody. |
| 19 | **Q:** Thuộc tính produces trong các Request Mapping là gì ?  🡺 **A:**  - Produces: kiểu dữ liệu trả về, thường dùng với các REST-API. Để tạo đối tượng trong Media Type được yêu cầu, sử dụng phần tử produces của @RequestMapping kết hợp với annotation @ResponseBody. |
| 20 | **Q:** Trình bày cơ chế Data Binding trong Spring  🡺 **A:**  - Là cơ chế liên kết dữ liệu đầu vào (đầu ra) với đối tượng model cho phép tương tác với dữ liệu dễ dàng hơn. Các form đều được liên kết với một đối tượng biểu diễn dữ liệu ở phía sau. Khi thay đổi dữ liệu của đối tượng thì dữ liệu trên các điều khiển cũng thay đổi theo.  **\*Chú ý:** Dữ liệu do người dùng nhập vào thông qua DataBinder trả về kết quả là BindingResult. DataBinder bao gồm các bộ xử lý PropertyEditors, bộ định dạng Fomatters, bộ lọc dữ liệu Validators và bộ chuyển đổi ConversionService |
| 21 | **Q:** So sánh RequestParam và PathVariable ?  🡺 **A:**   |  |  | | --- | --- | | RequestParam | PathVariable | | - Có thể lấy dữ liệu từ form  - Lấy dữ liệu theo kiểu key - value | - Chỉ có thể lấy dữ liệu trên URL  - Chỉ lấy dữ liệu theo value | |
| 22 | **Q:** Thymeleaf là gì?  🡺 **A:**  - Là bộ công cụ dùng để xử lý view(template engine) dựa trên ngôn ngữ Java |
| 23 | **Q:** Sử dụng lặp trong Thymeleaf như thế nào? Sử dụng điều kiện trong Thymeleaf như thể nào?  🡺 **A:**  - Cú pháp sử dụng vòng lặp trong Thymeleaf:  <**someHtmlTag** th:each="item : ${items}">  ....  </**someHtmlTag**>  - Cú pháp sử dụng điều kiện trong Thymaleaf:  <**someHtmlTag** th:if="condition">  ......  </**someHtmlTag**> |
| 24 | **Q:** Formatter là gì ? Converter là gì ?  🡺 **A:**  - Formatter là một đối tượng được dùng để chuyển đổi kiểu dữ liệu này sang kiểu dữ liệu khác, chỉ được thiết kế sử dụng ở tầng web  - Converter là một đối tượng được dùng để chuyển đổi kiểu dữ liệu này sang kiểu dữ liệu khác, được sử dụng cho toàn bộ hệ thống, sử dụng ở bất kì tầng nào của ứng dụng |
| 25 | **Q:** ORM là gì ?  🡺 **A:**  - (Object Relational Mapping) là kỹ thuật liên kết giữa các đối tượng trong lập trình với đối tượng trong CSDL |
| 26 | **Q:** JPA là gì? Spring Data JPA là gì ?  🡺 **A:**  - JPA (Java Presistance API) là tập hợp các interface dùng cho lập trình ứng dụng Java  - Spring Data JPA là framework giúp cho việc sử dụng JPA trong ứng dụng Spring dễ dàng hơn. Cung cấp các chức năng cơ bản như: CRUD, phân trang, search, sort |
| 27 | **Q:** Hibernate là gì ? Hibernate làm gì với database  🡺 **A:**  - Là ORM Framework triển khai những gì mà JPA đã thiết kế  - Hibernate cho phép truy vẫn dữ liệu thông qua Java Persistence API (JPA) hoặc bằng ngôn ngữ SQL mở rộng của Hibernate (HQL) hoặc bằng SQL thuần (Native SQL). |
| 28 | **Q:** Liệt kê một số annotation của hibernate?  🡺 **A:**  - @Entity được sử dụng để chú thích một class là một Entity.  - @Table cho phép chú thích tên bảng thông qua thuộc tính name  - @Column được sử dụng để chỉ định thông tin chi tiết của cột  - @Id được sử dụng để mô tả đây là Id (Identity) của Entity, nó tương đương với cột đó là khóa chính (Primary Key) của table trong database.  - @GeneratedValue được sử dụng để Hibernate tự động tạo ra giá trị và gán vào cho một cột  - @ManyToOne mô tả một quan hệ  **N-1**  **-** @OneToMany mô tả quan hệ **1-N**  **-** @OneToOne mô tả quan hệ **1-1**  **-** @ManyToMany mô tả quan hệ **N-N** |
| 29 | **Q:** Entity là gì?  🡺 **A:**  - Là đối tượng đại diện cho dữ liệu trong ứng dụng. Một entity thông thường sẽ là 1 POJO (Plain Old Java Object) |
| 30 | **Q:** Persistence Context & Entity Manager là gì?  🡺 **A:**   |  |  | | --- | --- | | Persistence Context | Entity Manager | | - Là tập các thể hiện của entity được quản lý, tồn tại trong một kho dữ liệu | - Là một interface cung cấp các API cho việc tương tác với các Entity. | |
| 31 | **Q:** Câu lệnh truy vấn động là gì? Câu lệnh truy vấn tĩnh là gì?  🡺 **A:**  - Câu lệnh truy vấn động hay còn gọi là HQL. EntityManager cung cấp phương thức createQuery() để tạo các câu lệnh truy vấn động  - Câu lệnh truy vấn tĩnh hay còn gọi là Native SQL (SQL thuần). Sử dụng phương thức createNamedQuery của EntityManager để tạo câu truy vấn tĩnh. |
| 32 | **Q:** Trạng thái của Entity bao gồm những gì?  🡺 **A:**  - Added: Thực thể được đánh dấu là đã thêm  - Deleted: Thực thể được đánh dấu là đã xóa  - Modified: Thực thể đã được sửa đổi  - Unchanged: Thực thể chưa được sửa đổi  - Detached: Thực thể không được theo dõi |
| 33 | **Q:** Có bao nhiêu loại mapping trong Hibernate?  🡺 **A:**  - Collections Mappings, Association Mappings, Component Mappings |
| 34 | **Q:** Để sử dụng interface JpaRepository cần cung cấp những thông tin nào?  🡺 **A:**  - Cung cấp đối tượng Entity và kiểu dữ liệu trường khóa chính của đối tượng |
| 35 | **Q:** Validation dữ liệu là gì?  🡺 **A:**  - Là quá trình xác minh dữ liệu từ người dùng, đảm bảo dữ liệu là đúng, là hợp lệ |
| 36 | **Q:** Trình bày cách triển khai validate dữ liệu trong Spring  🡺 **A:**  - Có 3 cách:  + Sử dụng annotation được cung cấp sẵn  + Custom validate  + Tự tạo ra các annotation |
| 37 | **Q:** Binding Result là gì?  🡺 **A:**  - Là một giao diện chỉ định cách đối tượng lưu trữ, kết quả xác thực sẽ lưu trữ và truy xuất kết quả xác thực (lỗi, cố gắng liên kết với các trường không được phép, ...) |
| 38 | **Q:** AOP là gì? AOP gồm những yếu tố nào?  🡺 **A:**  - AOP (Aspect Oriented Programming) lập trình hướng khía cạnh, là phương pháp lâp trình nhằm phân tách chương trình thành các module riêng lẻ, không phụ thuộc vào nhau  - Trong AOP sẽ chia chương trình thành 2 loại concern khác nhau:  + Core\_concern / Primary Concern: đây là requirement là nghiệp vụ chính của chương trình  + Cross\_cutting\_corcern: là những logic xử lý phụ cần được thực hiện của chương trình khi core\_concern được gọi như: bảo mật, ghi log, theo dõi server,... |
| 39 | **Q:** Các loại Advice?  🡺 **A:**  - Có 5 loại:  + Before Advice:  + After Advice:  + Around Advice: thêm cả trước và sau  + After Returning Advice: thêm ngay khi phương thức được return  + After Throwing Advice: thêm ngay khi ngoại lệ được ném ra |
| 40 | **Q:** JoinPoint là gì?  🡺 **A:**  - Là một điểm trong chương trình, là những nơi có thể chèn cross-cutting-concern |
| 41 | **Q:** Nêu cách triển khai AOP mà bạn dùng trong dự án  🡺 **A:**  - Khởi chạy Spring Initializr, chọn Spring Web, Thymeleaf, Spring Data JPA, MySQL Driver, Validation |
| 42 | **Q:** Mục đích xử lý ngoại lệ trong Spring Web MVC ?  🡺 **A:**  - Khi một ngoại lệ xảy ra chương trình sẽ bị ngắt. Khi đó, tất cả các nguồn tài nguyên mà hệ thống đã cấp không được giải phóng. Điều này gây lãng phí tài nguyên. |
| 43 | **Q:** Các cách xử lý ngoại lệ trong Spring Web MVC ?  🡺 **A:**  **-** Sử dụng try-catch để xử lý các ngoại lệ tại request  - Sử dụng @ExceptionHandler để ủy quyền cho front controller xử lý ngoại lệ.  - Sử dụng HandlerExceptionResolver để xử lý ngoại lệ tại servlet.  - Sử dụng trang báo lỗi để bắt các fatal error hay các ngoại lệ sinh ra ngoài phạm vi của Spring. |
| 44 | **Q:** Cookie là gì?  🡺 **A:**  - Là tệp được tạo ra khi người dùng truy cập vào các website. Cookie được lưu trữ vào Client nhằm tăng tính trải nghiệm người dùng, ghi nhớ thông tin đăng nhập, lưu thông tin người dùng |
| 45 | **Q:** Phân biệt Session và Cookie  🡺 **A:**   |  |  |  | | --- | --- | --- | |  | Session | Cookie | | Giống nhau | - Đều tạo ra dữ liệu trong quá trình người dùng sử dụng | | | Khác nhau | - Được lưu ở server  - Thời gian sống phụ thuộc vào phiên làm việc  - Có thể lưu trữ các kiểu dữ liệu của server  - Hoạt động dựa vào Cookie  - Dung lượng vô hạn | - Được lưu ở Client  - Thời gian sống phụ thuộc vào sự cài đặt của LTV  - Chỉ lưu trữ được dữ liệu text  - Dung lượng tối đa 4KB | |
| 46 | **Q:** Session là gì ? Cho ví dụ về session ?  🡺 **A:**  - Là phiên làm việc giữa client và server. Một session bắt đầu khi client thực hiện request đầu tiên lên server và kết thúc khi client dừng làm việc với server  - Ví dụ: Thông thường sử dụng session để lưu thông tin đăng nhập, giỏ hàng hoặc những dữ liệu mang tinh tạm thời |
| 47 | **Q:** Nêu cách triển khai Session mà bạn dùng trong dự án  🡺 **A:**  - Phương pháp 1: Inject trong HttpSession nơi được yêu cầu – Sử dụng annotation @Autowired để tiêm phụ thuộc đối tượng httpSession. Ví dụ: Để hiện thị student trong httpsession, chúng ta có GetMapping với Request là viewSession, lấy student từ httpsession ra sẽ có Student student = httpSession.getAttibute(attribute tên là student), lúc đó sẽ lấy ra được một object là student và cần phải ép kiểu, thêm student vào ModelAndView  - Phương pháp 2: Sử dụng như một tham số - Thêm HttpSession vào trong @RequestMapping  - Phương pháp 3: Tạo một bean và giới hạn cho session làm việc  - Phương pháp 4: Sử dụng annotation @SessionAttributes |
| 48 | **Q:** Vì sao Session hoạt động dựa vào Cookie  🡺 **A:**  - Khi truy cập trang web, Server sẽ tạo ngẫu nhiên 1 ID định danh cho phiên hoạt động và đồng thời máy cũng tự sinh 1 cookie lưu giá trị của Session ID đó. Viêc này giúp Server quản lý được các request gửi lên để có thể trả respone về cho đúng client đã request. |
| 49 | **Q:** Web Service là gì? Lấy ví dụ về web service  🡺 **A:**  - Là các thành phần ứng dụng được hiển thị dưới dạng dịch vụ, có thể được tích hợp với các ứng dụng viết bằng cách ngôn ngữ khác nhau  - Ví dụ: hệ thống xử lý đơn đặt hàng và quản lý tài khoản. Nhân viên sử dụng ứng dụng khách hàng để tạo tài khoản mới và nhập các đơn đặt hàng của khách hàng  mới. Logic xử lý cho hệ thống này được viết bằng CSDL để lưu trữ thông tin. |
| 50 | **Q:** Phân biệt Web Service và Web truyền thống (Website)  🡺 **A:**   |  |  | | --- | --- | | Website | Web Service | | - Có giao diện  - Con người sử dụng  - Sử dụng được đa nền tảng  - Tương tác bằng button, input, form...  - Ứng dụng đầu cuối | - Không có giao diện  - Được sử dụng bởi các ứng dụng khác  - Độc lập về nền tảng  - Thông qua các giao thức HTTP: GET, POST, PUT, DELETE  - Được sử dụng để làm back-end, xử lý nghiệp vụ và các phần chức năng cho front-end | |
| 51 | **Q:** SOAP là gì? RESTful là gì?  🡺 **A:**  - SOAP(Simple Object Access Protocol) là giao thức sử dụng XML để định nghĩa và giao tiếp dữ liệu thông qua HTTP  - RESTful là các webservice được xây dựng dựa trên REST |
| 52 | **Q:** jQuery là gì? Ajax là gì?  🡺 **A:**  - jQuery là thư viện của JS, slogan “write less, do more”, giúp cho việc phát triển ứng dụng bằng JS nhanh hơn và code ngắn gọn hơn  - AJAX (Asynchrous Javascript and XML) là một nhóm các công nghệ cho LTV phát triển các ứng dụng web có thể cập nhập được dữ liệu mà không cần load lại trang |
| 53 | **Q:** Nêu ví dụ về cách bạn triển khai Ajax trong dự án của bạn  🡺 **A:**  - Vào build.gradle để thêm dependences jackson-core và jackson-databind và cấu hình thư viện sử dụng ajax vào file HTML |
| 54 | **Q:** @RequestHeader và @ResponseHeader có ý nghĩa gì?  🡺 **A:**  - @RequestHeader là các thông số kèm theo khi gửi yêu cầu lên server  - @ResponseHeader là các thông số kèm theo khi server gửi kết quả về cho client. |
| 55 | **Q:** I18n và L10n là gì?  🡺 **A:** |
| 56 | **Q:** Nêu cách triển khai I18N và I10N trong dự án Spring  🡺 **A:** |
| 57 | **Q:** Interceptor là gì?  🡺 **A:** |
| 58 | **Q:** Bảo mật là gì? Cơ chế bảo mật trong Spring như thế nào?  🡺 **A:** |
| 59 | **Q:** Authentication là gì ? Authorization là gì ?  🡺 **A:**  - Authentication hay còn gọi là xác thực là quá trình chứng thực thông tin của người dùng là đáng tin cậy. Xác thực cũng có nghĩa là chúng ta công nhận nguồn gốc của đối tượng  - Authorization hay còn gọi là phân quyền, là quá trình xác định xem một người dùng nào đó có được sử dụng những tính năng và quyền truy cập mà ứng dụng cho phép. Việc cấp quyền không thể được sử dụng mà không có sự xác thực |
| 60 | **Q:** Liệt kê các cơ chế xác thực  🡺 **A:**  - HTTP Basic, Cookies, One Time Password, JWT: Json Web Token, Signatures: chữ kí số, OAuth2: thông qua bên thứ 3 |
| 61 | **Q:** CSRF là gì  🡺 **A:** |
| 62 | **Q:** CORS là gì  🡺 **A:** |
| 63 | **Q:** Spring Boot là gì?  🡺 **A:**  - Là một trong số các module của [Spring framework](https://james.codegym.vn/mod/page/view.php?id=1746), cung cấp tính năng RAD (Rapid Application Development), được sử dụng để tạo ra một ứng dụng độc lập dựa trên Spring có thể chạy ngay với rất ít cấu hình cần thêm. |
| 64 | **Q:** Sự khác nhau giữa SpringBoot và SpringMVC là gì?  🡺 **A:**   |  |  | | --- | --- | | Spring Boot | Spring MVC | | - Là một module của Spring để đóng gói ứng dụng dựa trên Spring với các giá trị mặc định  - Cung cấp các cấu hình mặc định  - Không có yêu cầu đối với các file mô tả khi triển khai  - Tránh mã viết sẵn và kết hợp các dependencies lại với nhau trong một đơn vị duy nhất  - Giảm thời gian phát triển và tăng năng suất | - Là based web framework dựa trên Spring  - Cung cấp các tính năng có sẵn  - Cần có nhiều file mô tả triển khai  - Chỉ định từng dependency riêng biệt  - Cần nhiều thời gian để đạt được năng suất cao | |
| 65 | **Q:** Cấu hình cho Spring Boot Tìm kiếm các Bean ở nhiều package khác nhau bằng cách nào?  🡺 **A:** |