* 1. 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 lý Request, nếu cần thì 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 ViewResolver.
* Cuối cùng kết quả sẽ được gửi lại FrontController và phản hồi về cho người dung.
* 2. Trình bày cơ chế Dependency Injection trong Spring.
* Dependency injection là một cơ chế để triển khai IoC(inversion of control).
* Các dependency được cung cấp và điềukhiển bởi container hoặc framework
* Thao tác “tiêm” các đối tượng vào trong đối tượng khác được thực hiện bởi container hoặc framework
* 3. Có bao nhiêu cách để thực hiện Dependency Injection? Trình bày?
* **Có 3 cách để chúng ta có thể Dependency Injection (nhúng một object vào một object khác).**
* **Thông qua constructor:**Các dependency (biến phụ thuộc) được cung cấp thông qua constructor
* **Thông qua setter :** Các dependency (biến phụ thuộc) sẽ được truyền vào 1 class thông qua các setter method
* **Thông qua interface:** Các client phải triển khai một Interface mà có một setter method dành cho việc nhận dependency.
* **4.** **Framework là gì ? Framework khác Library chỗ nào ?**

|  |  |
| --- | --- |
| **Framework** | **Library** |
| – Framework và Library đều cung cấp các tính năng (functions) được viết sẵn để chúng ta có thể tái sử dụng. | |
| – Framework lớn hơn và phức tạp hơn Library. |  |
| – Sử dụng Framework bạn phải thay đổi cấu trúc code của dự án theo các quy tắc của framework đó để có thể sử dụng được các functions mà framework đó cung cấp. | – Chúng ta có thể sử dụng các functions của Library một cách trực tiếp mà không cần thay đổi cấu trúc code của dự án. |
| – Framework có thể hiểu là một khung chương trình, người dùng bổ sung code và tuân theo quy tắc để tạo ra ứng dụng. | – Library chỉ cung cấp các chức năng tiện ích hay các class để sử dụng trong quá trình xây dựng ứng dụng. |
| – Framework hoạt động chủ động. Nghĩa là nó có thể đưa ra các quyết định gọi hoặc bị gọi bởi các Library hay ứng dụng nào đó. | – Library hoạt động bị động. Nghĩa là nó chỉ được gọi khi nào chúng ta cần dùng nó. |

* **5.** **Spring Framework là gì ?**
* Spring là một Framework phát triển các ứng dụng Java được sử dụng bởi hàng triệu lập trình viên. Nó giúp tạo các ứng dụng có hiệu năng cao, dễ kiểm thử, sử dụng lại code…
* Spring Framework được xây dựng dựa trên 2 nguyên tắc design chính là: Dependency Injection và Aspect Oriented Programming(AOP).
* 6. Lợi ích của Spring Framework?
* sử dụng để phát triển Java Desktop, ứng dụng mobile, Java Web. Mục tiêu chính của Spring là giúp phát triển các ứng dụng J2EE một cách dễ dàng hơn.
* 7. Nguyên lý đảo ngược quyền điều khiển (Inversion of Control) là gì ?
* Nguyên lý thiết kế mà framework nắm quyền kiểm soát code của chúng ta.
* Các object dependency sẽ do IOC container quản lý không còn do ứng dụng quản lý.
* 8. Bean là gì?
* bean là những module chính của chương trình, được tạo ra và quản lý bởi Spring IoC container.
* Các bean có thể phụ thuộc lẫn nhau, . Sự phụ thuộc này được mô tả cho IoC biết nhờ cơ chế Dependency injection.
* 9. Trong Spring có bao nhiêu Bean Scope?
* Có 5 scope được định nghĩa cho Spring Bean:
* **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.
* 10. @Autowire là gì?
* @Autowired **đánh dấu cho Spring biết rằng sẽ tự động inject bean tương ứng vào vị trí được đánh dấu**.
* 11. @Component có ý nghĩa gì?
* là một annotation của class. Nó dùng để đánh dấu class **Java**là một bean.
* 12. Trình bày ý nghĩa của Controller?
* Là nơi trung gian dung để giao tiếp giữa người dung và ứng dụng thông qua request sau đó thông báo tới View và Model. Sau đó liền gửi các lệnh tới Model để thay đổi trạng thái của Model
* 13. Trình bày ý nghĩa của ModelAndView Interface.
* Truyền và render dữ liệu từ controller về view
* Là 1 class khai báo 1 object khi sử dụng và cần khai báo thêm tên của view.
* 14. Trình bày ý nghĩa của ModelMap.
* Truyền và render dữ liệu từ controller lên view
* Là 1 class được khai báo như 1 tham số và có them các phương thức của Map.
* 15. Trình bày ý nghĩa của ViewResolver Interface.
* Là 1 cơ chế giúp frontController render dữ liệu sang view.
* 16. Phân biệt POST với PUT thường sử dụng để làm gì?

|  |  |
| --- | --- |
| PUT | POST |
| Put dung để cập nhật dữ liệu | post dung để tạo mới dữ liệu. |
| PUT method nên kèm theo một định danh (thường là ID) cùng với một entity. Nếu một tài nguyên được tìm thấy bởi mã định danh kèm theo thì tài nguyên này sẽ được thay thế bởi các giá trị trong entity kèm theo. | POST method sẽ tạo một tài nguyên dựa trên entity đã cung cấp. |
| nếu chúng ta gọi PUT method nhiều lần thì sẽ nó tạo hoặc cập nhật cùng một tài nguyên (dựa vào mã định danh). | POST thì việc chúng ta thực thi nhiều lần với cùng một enity thì nó sẽ tạo ra nhiều tài nguyên ở phía server có giá trị tương tự nhau. |

* 17. @RequestMapping làm gì?
* Là 1 annotation dung để ánh xạ 1 request đến method ở controller được đặt ở trên đầu class controller
* Gồm có 5 biến thể: GET, POST, PUT, PATH, DELETE.
* 18. Thuộc tính consumes trong các Request Mapping là gì ?
* Để quy định định dạng dữ liệu mà một request có thể process khi người dùng request tới,
* 19. Thuộc tính produces trong các Request Mapping là gì ?
* produces định nghĩa định dạng dữ liệu sẽ trả về cho user
* 20. Trình bày cơ chế Data Binding trong Spring.
* Là kĩ thuật liên kết dữ liệu đầu vào/ đầu ra với các object model thông qua cơ chế databinder với 3 nhiệm vụ
  + - Property Editor(liên kết với các thuộc tính).
    - Validator (kiểm tra tính hợp lệ của dữ liệu).
    - Formatter/ Converter(định dạng/ chuyển đổi dữ liệu).
* 21. So sánh RequestParam và PathVariable ?

|  |  |
| --- | --- |
| RequestParameter | PathVariable |
| Dung để trích xuất dữ liệu theo đường dẫn có dạng query (key,value) | Dung để trích xuất dữ liệu theo đường dẫn theo dạng URL chỉ chứa value |
| Kém bảo mật | Bảo mật hơn |

* 22. Thymeleaf là gì?
* Là 1 java template engine dung để xử lý và tạo HTML , XML, JS, CSS, TEXT.
* Là bộ công cụ dung để xử lý View.
* 23. Sử dụng lặp trong Thymeleaf như thế nào? Sử dụng điều kiện trong Thymeleaf như thế nào?
* Sử dụng lặp : th:each=”variableName, status:${items}”;
* Sự dụng điều kiện: th:if , th:unless, th:switch.
* 24. Formatter là gì ? Converter là gì ?

|  |  |
| --- | --- |
| Formatter | Converter |
| Chuyển đổi 1 kiểu dữ liệu này sang 1 kiểu dữ liệu khác | |
| Kiểu dữ liệu nguồn phải là kiểu String | Làm việc với bất kì kiểu dữ liệu nguồn nào |
| Phù hợp làm việc ở tầng web | Làm việc với bất kì tầng nào |

* 25. ORM là gì ?
* Là (object relational mapping) kĩ thuật để liên kết các object trong lập trình với các object trong Database,

giúp ánh xạ các class đến các bảng và các đối tượng đến các Record trong các bảng.

* 26. JPA là gì? Spring Data JPA là gì ?
* JPA là bộ đặc tả ( bộ khung quy chuẩn ) bao gồm các interface được thiết kế từ ý tưởng ORM .
* Spring Data JPA : là cái tiến tiêu chuẩn của JPA ,

Đơn giản hóa tầng truy xuất dữ liệu ,

Có thể tự tạo Repo ,

Truy vấn CSDL .

* 27.Hibernate là gì ?
* Là một thư viện ORM mã nguồn mở giúp lập trình viên viết ứng dụng JAVA có thể map các OBJ(POJO) với hệ quản trị CSDL quan hệ ,

Và hổ trợ thực hiện các khái niệm lập trình OOP với CSDL quan hệ .

* 28.Liệt kê một số Annotation của hibernate
* @Entity : Định nghĩa class đó là 1 entity .
* @Table :
* @Column :
* @Transient :
* @Temporal :
* @ID
* @GeneratedValue
* @Lob
* @ManyToOne
* @OneToMany
* @OneToOne
* @ManyToMany
* @OrderBy
* 29.Entity là gì ?
* Là 1 JAVA CLASS tương ứng với 1 table trong DB , mỗi Entity OBJ tương ứng với một dòng trong bảng .
  + JPA yêu cầu một entity class phải cung cấp đầy đủ các metadata để ánh xạ giữa các entity class và table như

(tên cột , kiểu dữ liệu , khóa chinh , khóa ngoại ) .

* 30. Persistence Context & Entity Manager là gì ?
* Persistence context : được coi như 1 container hoặc là cache cấp đầu tiên cho tất cả các đối tượng mà chúng ta đã load hoặc lưu vào CSDL trong 1 session
* Entity Manager : Là một interface cung cấp các API cho việc tương tác giữa ứng dụng với DB
  + Một số chức năng cơ bản của EM :
* Persist : tạo mới một thực thể vào CSDL
* Merge : Cập nhật 1 thực thể trong CSDL
* Remove : Xóa một thực thể trong CSDL
* 31. Câu lệnh truy vấn động là gì ? Câu lệnh truy vấn tĩnh là gì ?
* EntityManager cung cấp phương thức createQuery() để tạo các câu lệnh truy vấn động
* Trong các lớp entity chúng ta có thể định nghĩa sẵn các câu lệnh truy vấn, sau đó sử dụng phương thức createNameQuery() của entityManager để gọi đến câu truy vấn đó.
* 32. Trạng thái của Entity bao gồm những gì?
  + **New (Mới tạo ra, chưa được lưu xuống CSDL):**
* Một object vừa mới được tạo ra mà chưa được kết nối với Hibernate và cũng chưa được ánh xạ vào bất cứ bảng nào trong database thì sẽ được coi như là ở trạng thái New (Transient)

### Persistent (được ghi xuống CSDL và quản lý trong Persistence Context):

### Detached ( tạm thời tách ra khỏi persistence contex)

### Một khi Persistence Context đang chạy đã đóng lại, mọi entity được liên kết trước đó sẽ bị tách ra. Các thay đổi tiếp theo sau đó sẽ không còn được theo dõi và động bộ hóa vào trong database nữa

* 33. Có bao nhiêu loại mapping trong Hibernate?
* có 4 loại mapping trong hibernate:
* oneToOne mối quan hệ 1 – 1.
* oneToMany mối quan hệ 1-n.
* manyToOne mối quan hệ n-1.
* manyToMany mối quan hệ n-n.
* 34. Để sử dụng interface JpaRepository cần cung cấp những thông tin nào?
* Cần implement interface JpaRepository và cần khai báo kiểu dữ liệu trong repository triển khai là tên của entity và kiểu dữ liệu khóa chính của entity đó.
* 35. Validation dữ liệu là gì?
* Validation là quá trình đánh giá/xác minh tính hợp lệ của dữ liệu đầu vào trước khi lưu vào Database.
* 36. Trình bày cách triển khai validate dữ liệu trong Spring.
* Có 2 cách để triển khai validate trong spring:
  + Validate bằng các annotation có sẵn của spring
  + Custom validate bằng cách implement interface Validator và triển khai thực hiện validate các trường của object trong phương thức validate của interface này.
* 37. Binding Result là gì?
* Binding result là đối tượng của spring, nơi lưu giữ kết quả rang buộc và các lỗi có thể xảy ra trong quá trình kiểm tra validate dữ liệu.
* 38. AOP là gì? AOP gồm những yếu tố nào?
* AOP (aspect oriented programing) là kĩ thuật lập trình hướng khía cạnh, phân tách chương trình thành các module riêng rẽ, phân biệt và ko phụ thuộc lẫn nhau.
* AOP không thay thế cho OOP mà sẽ tách chỉ hỗ trợ phân tách các module nhỏ để dễ dàng quản lý hơn.
* AOP gồm những yếu tố:

- Aspect: là 1 khối độc lập với OOP. Bổ trợ cho OOP.

- Advice: là các hoạt động mà các aspect dung để quan sát đến OOP.

- PoinCut: tập hợp các điểm có khả năng xảy ra advice.

- JoinPoin: các điểm có khả năng xảy ra advice.

* 39. Các loại Advice?
* Trong Spring thì AOP hỗ trợ 4 loại advice:
  + Before advice: chạy trước khi method thực thi.
  + After returning advice: chạy sau khi method trả về 1 kết quả.
  + After throwing advice: chạy sau khi method ném ra exception.
  + Around advice: chạy khi method thực thi(bao gồm cả 3 advice trên).
* 40. JoinPoint là gì?
* Là điểm bất kì nào có thể xác định được khi chạy chương trình.
* Một số loại joinpoin chính trong AOP spring:
  + Tại constructor.
  + Tại các method.
  + Tại các điểm truy cập thuộc tính.
  + Tại các điểm điều khiển ngoại lệ.
* 41. Nêu cách triển khai AOP mà bạn dùng trong dự án.
* Chia nhỏ chương trình thành nhiều module riêng rẽ, bớt phụ thuộc lẫn nhau.
* Khi hoạt động các chương trình sẽ kết hợp các module lại để thực hiện các chức năng. Khi chỉnh sửa thì không cần sửa tất cả module.
* 42. Mục đích xử lý ngoại lệ trong Spring Web MVC ?
* Khi xảy ra 1 exception thì hệ thống sẽ báo lỗi, thay vì hiển thị hết các lỗi đó khiến người dung khó hiểu thì chúng ta sẽ che giấu và xử lý exception đó và trả về cho client 1 thông điệp thực tế hơn ở giao diện web.
* 43. Các cách xử lý ngoại lệ trong Spring Web MVC ?
* Dung try catch để xử lý tại chỗ.
* Dung throw, throws để ném ra ngoài cho nơi gọi xử lý.
* Sử dụng handlerException của spring.
* 44. Cookie là gì?
* Là thành phần lưu trữ dữ liệu lâu dài, chỉ kết thúc khi hết thời gian setMaxAge hoặc khi bị người dung xóa trên browser.
* 45. Phân biệt Session và Cookie.

|  |  |  |
| --- | --- | --- |
|  | Session | Cookie |
| Kiểu dữ liệu | * Bất kì kiểu dữ liệu nào | Chỉ hoạt động với String (dạng text) |
| Vòng đời | * + Tắt sever   + Tắt trình duyệt   + Hết thời gian TimeOut | * + Hết thời gian setMaxage()   + Người dung xóa cookie ở browser |
| Nơi lưu trữ | * + Trên Server | * + Client (browser). |
| Độ bảo mật | * + Bảo mật tốt hơn do được lưu tại server | * + Kém bảo mật hơn do được lưu tại browser và có thể thay đổi. |

* 46. Session là gì ? Cho ví dụ về session ?
* Session là thành phần lưu trữ dữ liệu trong 1 phiên làm việc , chỉ kết thúc khi tắt trình duyệt , tắt server hoặc hết timeOut.
* Ví dụ: ứng dụng giỏ hang trên shoppe khi them 1 sản phẩm vào giỏ hang………………………………………………………
* 47. Nêu cách triển khai Session mà bạn dùng trong dự án.
* B1: Sử dụng @SessionAttributes để khai báo Session  
  B2: Sử dụng @ModelAttribute bên ngoài handle method để khởi tạo Session  
  B3: Sử dụng @ModelAttribute trong tham số của handle method của Controller  
  tạo ra nó để tương tác với Session  
  B4: Sử dụng @SessionAttribute trong tham số của handle method của Controller  
  không tạo ra Session đó để tương tác với Session
* 48. Web Service là gì? Lấy ví dụ về web service.
* Là thành phần bao gồm các tiêu chuẩn hoặc giao thức để trao đổi dữ liệu giữa các ứng dụng với nhau.
* Các ứng dụng phần mềm được viết bằng các ngôn ngữ lập trình khác nhau và  
  chạy trên nhiều nền tảng khác nhau, có thể sử dụng web service để trao đổi dữ  
  liệu qua mạng máy tính
* Web service hoạt động một cách độc lập không phụ thuộc bất kì ngôn ngữ nào.  
  Các ứng dụng java, .net hoặc PHP... có thể giao tiếp với các ứng dụng khác  
  thông qua web service
* VD: Chúng ta ứng dụng có FE phát triển từ Angular thực hiện việc tiếp nhận  
  request. Sau đó gởi dữ liệu về Web service, web service sẽ gởi request về BE
* 49. Phân biệt Web Service và Web truyền thông.

|  |  |
| --- | --- |
| WebService | WebSite |
| * + Không có giao diện | * + Có giao diện |
| * + Được sử dụng bởi các ứng dụng khác | * + Được sử dụng bởi người dung. |
| * + Độc lập nền tảng , sử dụng giao thức chuẩn để giao tiếp. | * + Hoạt động đa nền tảng vì cần được tinh chỉnh phù hợp với các ngôn ngữ hay hệ điều hành khác nhau. |
| * + Truy cập thông qua ô input thao tác sự kiện với button | * + Truy cập thông qua các phương thức: post, get, put, path, delete. |
| * + Website là ứng dụng đầu cuối, người dùng truy cập website qua URL, qua  đó nhận được những dữ liệu như text,  hình ảnh, âm thanh có thể dễ dàng  hiểu được | * + Là 1 khái niệm rộng hơn, dữ liệu trả  về từ web service, người dùng khó có  thể hiểu được. Dữ liệu đó được các  ứng dụng khác sử dụng và chế biến  thành các dữ liệu mà người dùng có  thể đọc được |

* 50. SOAP là gì? RESTful là gì?

|  |  |
| --- | --- |
| SOAP | RESTful |
| * + SOAP (stands for Simple Object Access Protocol) và RESTful (REpresentational State Transfer) là 2 kiểu thực hiện web service khác nhau. | |
| * + SOAP chỉ trả về dữ liệu dạng XML | * + RESful trả về dữ liệu ở nhiều định dạng khác nhau: Plain Text, HTML, XML and JSON… |
| * + Tốc độ của soap chậm hơn restful | * + Tốc độ nhanh hơn |
| * + Soap có tính bảo mật và toàn vẹn hơn restful | * + Bảo mật kém hơn |
| * + SOAP là một kiểu giao thức (protocol) na ná như HTTP, nó cũng có phần header, body… để định nghĩa dữ liệu gửi về | * + RESTFul giống như 1 kiểu kiến trúc gửi/ nhận (client gửi request/dữ liệu tới, server trả dữ liệu tương ứng về) |

* 51. jQuery là gì? Ajax là gì?
* Jquery là thư viện được viết bằng Javascript, giúp xây dựng các chức năng bằng JS nhanh và dễ dàng với với nhiều tính năng.
* Ajax(Asynchronous JS and XML) là nhóm các công nghệ phát triển web được sử dụng để tạo các trang web giàu tính tương tác, nhanh hơn ,mượt hơn với sự hỗ trợ của XML, HTML,CSS, JS thông qua cơ chế bất đồng bộ.
* 52. Nêu ví dụ về cách bạn triển khai Ajax trong dự án của bạn.
* Khi click vào 1 button thì 1 đoạn văn bản sẽ được ẩn đi mà không cần chuyển hướng hay load lại trang:
* <button type=”button” id=”but”>click me</button>

<p>lồn mẹ</p>

<script>

$(“#but”).click(function(){

$(“p”).hide();

});

</script>

* 53. @RequestHeader và @ResponseHeader có ý nghĩa gì.

|  |  |
| --- | --- |
| @RequestHeader | @ResponseHeader |
| * + Được sử dụng khi muốn lấy dữ liệu được truyền bằng Header của một  request (yêu cầu từ client) | * + Được sử dụng khi muốn trả về thêm  dữ liệu cho client ở phần trên cùng của mỗi response |

* 54. I18n và L10n là gì?

|  |  |
| --- | --- |
| L18n | L10n |
| * + I18n là một kỹ thuật để phát triển ứng dụng hỗ trợ nhiều định dạng ngôn ngữ và dữ liệu mà không cần phải viết lại logic lập trình. | * + là kỹ thuật thích ứng với một ứng dụng quốc tế hóa để hỗ trợ một địa phương cụ thể. |

* 55. Nêu cách triển khai I18N và I10N trong dự án Spring.
* Để có được l18n và l10n ứng dụng, yêu cầu phải:
  + Tách các thành phần văn bản thành tệp properties.
  + Làm cho ứng dụng có khả năng chọn và đọc chính xác các trường từ tệp thuộc tính.
* 56. Interceptor là gì?
* Là thành phần được đính vào trong một phần vòng đời của request.
* Có khả năng xử lý tiền/hậu kỳ các request.
* Được dùng để them hoặc cập nhập cấu hình trước khi request được xử lý bởi controller.
* 57. 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 hoạt động dựa trên Authentication và Authorization
* 58. 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 là đáng tin cậy.
* Authorization (Cấp quyền/phân quyền) là quá trình xác định xem một người dùng có quyền truy cập một tài nguyên cụ thể để thực hiện một số hành động hay không.

.

* 59. Các cơ chế xác thực ?
* OTP (one time password) mật khẩu được tạo ra 1 lần.
* Signature (chữ kí điện tử)
* Token : thường sử dụng JWT bao gồm 3 phần >> header , payload, signature.
* HTTP basic : mã hóa chuỗi username + password dựa trên thuật toán base 64 hay MD5.
* Cookie : việc xác thực sẽ dựa vào các thông tin được lưu trữ ở cookie.
* 60. CSRF là gì?
* **CSRF** hay còn gọi là kỹ thuật tấn công “**Cross-Site Request Forgery**“, nghĩa là kỹ **thuật tấn công giả mạo** chính chủ thể của nó.
* **CSRF** nói đến việc tấn công vào chứng thực request trên web thông qua việc sử dụng Cookies, nơi mà các hacker có khả năng sử dụng thủ thuật để tạo request mà bạn không hề biết. Vì vậy, một CSRF là hacker lạm dụng sự tin tưởng của một ứng dụng web trên trình duyệt của nạn nhân.
* 61. CORS là gì?
* Cross-Origin Resource Sharing (viết tắt là CORS) là một tiêu chuẩn để truy cập tài nguyên web trên các tên miền khác nhau.
* CORS cho phép các web scripts tương tác cởi mở hơn với nội dung bên ngoài tên miền gốc, dẫn đến sự tích hợp tốt hơn giữa các dịch vụ web.
* 62. Spring Boot là gì?
* Spring Boot là một trong số các module của Spring framework, cung cấp tính năng RAD (Rapid Application Development). 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
* 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.
* 63. Sự khác nhau giữa SpringBoot và SpringMVC là gì?

|  |  |
| --- | --- |
| Spring MVC | Spring Boot |
| * + **Spring MVC** model view controller-based web framework dựa trên Spring. | * + **Spring Boot** là một module của Spring để đóng gói ứng dụng dựa trên Spring với các giá trị mặc định hợp lý. |
| * + Cần có nhiều file mô tả triển khai. | * + Không có yêu cầu đối với các file mô tả khi triển khai. |
| * + Nó chỉ định từng dependency riêng biệt. | * + Nó tránh mã viết sẵn và kết hợp các dependencies lại với nhau trong một đơn vị duy nhất. |
| * + Cần nhiều thời gian hơn để đạt được điều tương tự. | * + Nó làm giảm thời gian phát triển và tăng năng suất. |

* 64. Cấu hình cho Spring Boot Tìm kiếm các Bean ở nhiều package khác nhau bằng cách nào?
* Đối với các ứng dụng Spring Boot, @SpringBootApplication annotation được sử dụng ở main class là sự kết hợp của cả 3 annotation:
  + @Configuration
  + @EnableAutoConfiguration
  + @ComponentScan
* mặc định Spring Boot sẽ quét tất cả các class ở cùng package ở main class và tất cả các sub-package của nó để tìm và khởi tạo các bean tương ứng đã được định nghĩa.
* 65. Trình bày Spring Security? @EnableWebSecurity làm gì?
* Spring Security là một loạt các bộ lọc servlet giúp thêm các authentication và authorization vào ứng dụng web
* hêm @EnableWebSecurity vào lớp @Configuration để xác định cấu hình Spring Security trong bất kỳ WebSecurityConfigurer nào hoặc nhiều khả năng  
  hơn bằng cách mở rộng lớp cơ sở WebSecurityConfigurerAdapter và ghi đè từng  
  phương thức riêng lẻ
* 66. Trình bày Spring Boot JPA? Cài đặt?
* Spring Boot JPA giúp sử dụng JPA trong Spring một cách dễ dàng hơn.
* Có thể sử dụng với các framework hibernate, OpenJPA, EclipseLink.
* Cải tiến JPA tiêu chuẩn, đơn giản hóa tầng truy xuất, tự tạo repository, tự tạo ra các truy vấn thông qua hàm ghi log, phân trang