**Ôn tập spring boot**

1. Spring Boot Starter: Spring Boot cung cấp các "starter" dễ dàng giúp tích hợp các thành phần cần thiết cho ứng dụng. Ví dụ: spring-boot-starter-web sẽ tự động bổ sung các dependency cho ứng dụng web.
2. Tự động cấu hình (Auto-Configuration): Spring Boot sẽ tự động cấu hình và khởi tạo các bean dựa trên các dependency trong classpath. Ví dụ, nếu bạn có một cơ sở dữ liệu H2 trong classpath, Spring Boot sẽ tự động cấu hình kết nối với cơ sở dữ liệu H2.
3. Embedded Servlet Container: Spring Boot hỗ trợ tích hợp các Servlet container như Tomcat, Jetty, và Undertow. Điều này giúp bạn không cần cài đặt, cấu hình riêng biệt Servlet container.
4. Spring Boot Actuator: Cung cấp các REST endpoint để giám sát và quản lý ứng dụng Spring Boot. Gồm các thông tin về bộ nhớ, health status, các beans, environment properties.
5. Spring Boot CLI: Command Line Interface giúp bạn nhanh chóng tạo ra và thử nghiệm ứng dụng Spring Boot bằng Groovy.
6. Template Engines: Spring Boot hỗ trợ nhiều template engine như Thymeleaf, FreeMarker, và Mustache. Giúp lập trình viên dễ dàng tạo ra view cho ứng dụng web.
7. Packaging và Deployment: Spring Boot cho phép tạo ứng dụng dạng JAR hoặc WAR. Điều này giúp dễ dàng triển khai ứng dụng trên môi trường sản xuất.
8. Embedded server: Spring Boot sử dụng một embedded server để chạy ứng dụng thay vì sử dụng một web server bên ngoài. Điều này giúp giảm thiểu thời gian cài đặt và triển khai ứng dụng.
9. cấu trúc 3 layer:
   * presentation: @RestController(tiếp nhận request)
   * business logic: @Service(xử lý logic)
   * data access: @Responsitory(giao tiếp database)
10. @requestMapping: là một annotation dùng để ánh xạ các yêu cầu http(get, post, put, delete,...), nó giúp xác định url của yêu cầu http và kiểu http đc sử dụng, cùng với các tham số khác như headers và và phương thức trả về.
11. một số yêu cầu http:
    * get: yêu cầu lấy dữ liệu từ phía ng dùng.
    * post: yêu cầu thêm mới dữ liệu từ phía ng dùng.
    * put: yêu cầu cập nhật dữ liệu từ phía ng dùng.
    * delete: yêu cầu xóa dữ liệu từ phía ng dùng.
12. @Bean:
    * @Bean trong Spring Boot là một đối tượng được quản lý bởi Spring container. Một bean là một đối tượng được khởi tạo, cấu hình và quản lý bởi Spring Framework. Bean được sử dụng để tạo ra các thành phần độc lập có thể tái sử dụng trong một ứng dụng mà không cần biết chi tiết về cách thức tạo ra, cấu hình và quản lý chúng.
    * Mỗi bean có một ID (hoặc tên) duy nhất để có thể được nhận dạng bởi container. Spring Boot sử dụng tên của phương thức hoặc tên của lớp (với chữ đầu tiên viết thường) làm ID của bean, nhưng bạn cũng có thể chỉ định một tên rõ ràng hơn.
13. định nghĩa một @bean bằng cách sử dụng cả hai cách sau:
    * Sử dụng annotation `@Bean`: Dùng `@Bean` để đánh dấu một phương thức trong một lớp cấu hình. Phương thức này sẽ tạo ra một đối tượng bean và Spring container sẽ quản lý đối tượng này.
    * Sử dụng annotation `@Component`: Dùng `@Component` để đánh dấu lớp của bean. Spring container sẽ tự động quét và tìm các lớp được đánh dấu bằng `@Component` và tạo ra các đối tượng bean tương ứng.
14. @Autowired: được sử dụng để tự động gắn kết các đối tượng. khi sử dụng @Autowired trên 1 thuộc tính hoặc 1 phương thức khởi tạo của 1 bean thì spring boot sẽ tự động tìm kiếm và gán giá trị cho thuộc tính hoặc tham số của phương thức khởi tạo đó.
15. ưu tiên bean:
    * @primary: sử dụng trên bean để chỉ định bean đó là primary
    * @qualifier: chỉ định 1 bean mà bạn muốn sử dụng trong trường hợp có nh bean cùng loại. bạn cần chỉ định tên hoặc giá trị của bean.