coggle
made for free at coggle.it

Module 4

- 1. Tổng quan Spring MVC
  - Phân biệt Framework và library
    - Phân biệt framework và library:
      - Framework là một ứng dụng phần mềm có tính trừu tượng cao, cung cấp các tính năng chung thông dụng nhất và có khả năng tùy biến.(thích hợp với dự án lớn)
      - Framework sẽ nắm quyền điều khiển, code của framework sẽ gọi code của lập trình viên(cơ chế IoC)
      - Library là công cụ, tập hợp các chức năng, class được viết sẵn để có thể tái sử dụng
      - Ứng dụng nắm quyền điều khiển, code của lập trình viên sẽ gọi code của library.
  - Các thành phần:
    - Front controller: Là nơi xử lý, lắng nghe toàn bộ request từ bên ngoài ứng dụng, thực hiện điều hướng và trả về data hiển thị cho người dùng.
    - DispatcherServlet: Đây là thành phần của front controller, nơi đón nhận và xử lý request của người dùng
    - Controller: Là file class xử lý logic, nghiệp vụ. Annotation `@Request` Mapping ánh xạ một URL đến một phương thức của controller
    - ViewResolver(trình phân giải view): Là cơ chế ánh xạ tên của view sang đối tượng view tương ứng.
    - Cách gửi data từ controller sang view: model, modelMap, ModelAndView.
  - Cơ chế Inversion of Control:
    - IoC là nguyên lí trong phát triển phần mềm, trong đóng việc điều khiển các đối tượng hoặc các thành phần trong hệ thống được thực hiện bởi framework hoặc các container. Với IoC thì framework nắm quyền điều khiển luồng thực thi của hệ thống.
    - Triển khai IoC có nhiều design pattern trong đó nổi bật là DI(Dependency Injection), lợi ích của IoC:
      -Tách rời việc thực thi (execution) và việc triển khai (implementation)
      -Dễ chuyển đổi giữa các implementation
      -Dễ phân tách module hơn
      -Dễ kiểm thử hơn, bằng cách tách rời các thành phần riêng lẻ.
    - Có 3 cách thực hiện DI:
      - constructor
      - setter
      - interface
- 2 Spring controller
  - `@Controller:` dung để đánh dấu một class là spring mvc controller(dùng cho web application)
    `@RestController:` dùng để đánh dấu một class restfull controller dùng cho web service.
    cần khai báo conponent-scan để phát hiện controller.
  - `@RequestMapping` dùng để ánh xạ các request tới các action tương ứng của controller (có thể dùng được cho class và phương thức)
    Các thuộc tính của `@RequestMapping:`
    - URL(value) có thể 1 hoặc nhiều url pattern
    - Method: GET,POST,DELETE,PUT,PATCH(update 1 phần)
    - Consumes: định dạng dữ liệu khi user request tới server(ánh xạ tới content-type của request header)
    - Produces: định nghĩa kiểu dữ liệu sẽ trả về cho user (ánh xạ tới accept của resuest header)
    - Header: lấy toàn bộ thông tin của header
    - Các biến thể của `@RequestMapping:` `@GetMapping` `@PostMapping` `@PutMapping` `@DeleteMapping` `@PatchMapping`
  - Các tham số truyền vào handle method: HttpServletRequest, HttpServletResponse, HttpSesion, `@RequestParam,` `@PathVariable,` `@ModelAttribute,` Model, ModelMap
  - RequestParam: dùng để truy cập một tham số của URI (thuộc tính value, require, defaultValue) lưu ý khi viết tắt thì tên param và thuộc tính value phải trùng nhau.
  - PathVariable dùng để lấy tham số tĩnh trên uri.
- 3. Databinding và form
  - Databinding là cơ chế liên kết dữ liệu đầu vào hoặc đầu ra với model, giúp cho việc tương tác với dữ liệu dễ dàng hơn.
    khi tương tác với form thì dữ liệu trên form sẽ tự động chuyển đổi thành các thuộc tính của đối tượng liên kết với form đó
    data binding hỗ trợ chuyển đổi dữ liệu và validate dữ liệu
  - Thư viện spring-form.tld: hỗ trợ tạo form trong spring, bổ sung thuộc tính modelAttribute cho form các thuộc tính quan trọng:
    - path: tên thuộc tính liên kết với trường hiện tại
    - item: danh sách các hạng mục của trường như checkboxs, radiobuttons.
- 4.Thymeleaf
  - • Thymeleaf là một bộ xử lý view(Java template engine) được sử dụng cho các ứng dụng web và các ứng dụng độc lập.(dùng để lấy dữ liệu từ controller sang view)
    • Được xây dựng phù hợp với các tiêu chuẩn của web, đặc biệt là HTML5
    • Thymeleaf cho phép xử lý 6 loại template: HTML, TEXT, XML, JAVA, SCRIPT, CSS, RAW
    • Ưu điểm so với jsp: thân thiện, sử dụng html thuần, dễ sử dụng.
  - • Các biểu thức của thymeleaf:
    - o Biểu thức với biến: ${}
    - o Biểu thức với thuộc tính *{}
    - o Biểu thức với message #{}
    - o Biểu thức với URL `@{}`
    - o Biểu thức với phân đoạn: ~{}
- 5. ORM
  - Entity
    - • Entity: là đối tượng đại diện cho dữ liệu trong ứng dụng, entity thường là Pojo(Plain Old Java Object) là lớp đơn giản, chỉ dùng để mô tả dữ liệu chứ không xử lí logic nghiệp vụ
      • Mỗi entity sẽ map với một table trong CSDL
      • Mỗi entity cần tuân thủ: gắn annotation `@Entity,` có constructor public ko tham số, không được khai báo final,thuộc tính của đối tượng có access modifier là private, protected hoặc ở mức pagkage-private.
  - • Object Relational Mapping là kỹ thuật liên kết giữa đối tượng trong lập trình với đối tượng trong CSDL.(là 1 nguyên lý)
    • Cụ thể là quá trình biến đổi giữa 1 record trong database và object trong ngôn ngữ lập trình hướng đối tượng
    • ORM có rất nhiều inplementation. Phổ biến là Hibernate framework, được sử dụng rộng rãi
  - • Persistence context là 1 tập các thể hiện của entity được quản lí, tồn tại trong 1 kho dữ liệu
    • Entity manager là interface cung cấp các phương thức để tương tác với persistence:
    - o Tạo các thể hiện của persistence
    - o Tìm kiếm entity theo khóa chính
    - o Thực hiện các câu lệnh truy vấn
    - o Phương thức createQuery() tạo các câu lệnh truy vấn động
    - o Phương thức createNameQuery() giúp tạo ra câu lệnh truy vấn tĩnh
- 6.JPA
  - • JPA là Java Persistence API, là một bộ đặc tả tiêu chuẩn của Java, mô tả các thao tác quản lí dữ liệu quan hệ trong các ứng dụng sử dụng java. Cung cấp mô hình cho phép ánh xạ các bảng trong CSDL và các class
    • Spring Data JPA là một module của spring framework giúp cho việc sử dụng JPA trong ứng dụng Spring trở lên dễ dàng hơn.
    • Spring Data JPA cải tiến JPA tiêu chuẩn, đơn giản hóa tầng truy xuất dữ liệu, tự tạo repository, tạo các truy vấn JPA thông qua tên phương thức
    - • Các interface trong Spring Data JPA:
      - o Repository
      - o CrudRepository
      - o PagingAndSortingRepository
      - o JpaRepository
- 7.Spring data JPA và Spring boot
  - Sử dụng `@PageableDefault` để cấu hình phân trang tại controller, các phương thức thông dụng là hasNext(), hasPrevious(), getPageNumber(), getPageSize(), getTotalPage().
  - spring boot: là một trong số các module của spring framework. được sử dụng để tạo ra một ứng dụng độc lập có thể chạy ngay với rất ít cấu hình thêm.
    ưu điểm:
    - tạo ứng dụng độc lập, chạy ngay bằng lệnh java
    - nhúng sẵn tomcat
    - cung cấp các thư viện starter
    - tự động cấu hình spring khi có thể không yêu cầu cấu hình xml
- 8 validation
  - Cách validate:
    • Dùng annotation có sẵn của spring, thuộc tính message dùng để tùy chỉnh lại nội dung thông báo lỗi.
    • Ở controller cần khai báo `@Valid` hoặc `@Validated` (đứng trước `@ModelAttribute` rồi đến bindingResult
    • Dùng custom validate: tại Dto cần implement Validator, overide 2 phương thức, trong đó có phương thức validate: thêm các điều kiện để validate, sau đó gọi phương thức validate ở phía controller.
  - • Để tránh các mối quan hệ giữa các entity, sử dụng Dto để validate, ngoài ra còn đảm bảo s trong solid
  - • Validate là thực hiện đánh giá, xác minh tính hợp lệ của dữ liệu đầu vào.
    - Có thể validate ở 3 tầng:
      - o UI Layer(phía client): dùng javascript, jquery, angular
      - o Business layer (phía server) dùng Java, PHP, C#
      - o Data Layer(phía DB) dùng SQL để tạo các constrain
- 9 AOP
  - • AOP là Aspect Oriented Programming – lập trình hướng khía cạnh: là một kĩ thuật lập trình nhằm phân tách chương trình thành các module riêng rẽ, phân biệt, không phụ thuộc lẫn nhau
    • Các chức năng chính của chương trình: sử dụng OOP, các chức năng phụ sử dụng AOP, vd ghi log, số lần login thành công, xử lí exception, bộ lọc filter, sercutity.. bổ trợ thêm cho OOP.
    • Những tính năng phụ trên gọi là khía cạnh, code để thực hiện aspect là advice, là các hành động cụ thể
  - • Các thuật ngữ trong AOP:
    - Core concern/ primary concern: là những xử lí logic chính của chương trình(thường là requiment)
    - Cross – cutting concern: mối quan tâm xuyên suốt, là những logic xử lí phụ cần được thực hiện của chương trình khi core concern được gọi như logging, tracing, monitoring
    - Aspect tương tự như một class java, một aspect đóng gói toàn bộ cross-cutting concern
  - advice
    - After returning advice: thực thi nếu như join poin kết thúc không có loại lệ
    - Before advice: thực thi trước khi vào join point
    - After throwing advice: thực thi nếu như join point kết thúc có ngoại lệ
    - Around advice: thực thi trước và sau khi join point
    - After (finally) advice: thực thi sau khi join poin hoàn thành dù thành công hay thất bại
  - • JoinPoint là điểm gia nhập(thường là method). Một vị trí nào đó trong luồng thực thi chương trình mà advice sẽ được đan (weave) vào chương trình chính
    • PointCut là 1 tập các join point để ccoong cụ AOP đan các advice vào nghiệp vụ chính(within, excution)
    • Weave (đan) là quá trình liên kết thành phần aspect và non-aspect của chương trình để tạo ra đầu ra mong muốn(hiển thị có log)
  - xử lí ngoại lệ
    - o Sử dụng try-catch, throw- throws
      o Sử dụng@exception
- Sesion Cookie
  - • Session được gọi là một phiên làm việc giữa client và server
    • Một session bắt đầu khi client thực hiện request đầu tiên đến server và nó sẽ kết thúc khi client dừng làm việc với server (đóng trình duyệt, server bị tắt)
    • ứng dụng: lưu thông tin đăng nhập, giỏ hàng… hoặc những dữ liệu mang tính tạm thời (HttpSession)
    - Triển khai sesion:
      - Dùng `@SessionAttributes("nameSession")` trên controller
      - Tạo phương thức khởi tạo đối tượng, trả về đối tượng, với `@ModelAttribute("nam")` có tên trùng với session.
      - Trong tham số của các phương thức, dùng `@modelAttribute` hoặc `@SessionAttribute` để gửi model xuống view để hiển thị.
      - Tại view html, sử dụng th:text=${tên model} để hiển thị model từ trên controller trả về
  - • Cookie là các tệp được các trang web người dùng truy cập tạo ra, được lưu trữ trong trình duyệt
    • ứng dụng cookie để duy trì trạng thái đăng nhập dù tắt trình duyệt, thông tin mà người dùng đã nhaapjm cung cấp nội dung phù hợp với vị trí, tiêu chí tìm kiếm của người dùng.
    - Triển khai Cookie
      - Dùng `@CookieValue(value="tenCookie",` defaultvalue) String tênCookie, Model model).
      - Tạo một cookie: Cookie cookie = new Cookie("tenCookie", valueCookie).
      - Tại phương thức trong controller, xét điều kiện để tạo cookie, tạo mới đối tượng cookie với tên và giá trị.
      - setMaxAge và setPath, response.addCookie(cookie)
      - tạo mảng cookie = request.getCookie, duyệt mảng, nếu tên cookie trùng tên cookie cần tìm thì trả xuống view tên cookieValue.
- webservice
  - • Webservice là 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(vd: login bằng facebook,bảng tin giá vàng…)
    • Hoạt động dựa trên mô hình client-server thông qua giao thức HTTP/HTTPS(không có tầng view, chỉ trả về data).
    • Kiểu dữ liệu dùng để giao tiếp thường là Json hoặc XML
    - ưu điểm:
      Hoạt động trên nhiều nền tảng, nhiều ứng dụng với các ngôn ngữ khác nhau
      Khả năng tái sử dụng cao
      Thời gian triển khai nhanh
      Giảm sự phức tạp của hệ thống(một số chức năng có thể mua từ nơi khác) giảm thời gian phát triển, giảm giá thành
    - nhược điểm
      Khi webservice bị chết/dừng hoạt động thì sẽ ảnh hưởng đến nhiều hệ thống khác
      Lý do bảo mật nên webservice cần được bảo mật nghiêm ngặt, chặt chẽ
  - • Rest là một kiểu cấu trúc cung cấp các quy tắc để xây dựng webservice. Sử dụng giao thức HTTP toàn bộ các tài nguyên được biểu thị dưới dạng uri(chuỗi kí tự xác định tài nguyên trong môi trường mạng)
    - các quy tắc
      - o PATCH: update 1 phần tài nguyên
      - o DELETE: xóa tài nguyên
      - o PUT: update tài nguyên
      - o POST: tạo tài nguyên trên máy chủ
      - o GET: lấy tài nguyên
      - o Restfull là phi trạng thái: ứng dụng không lưu thông tun, phiên làm việc của client.(vì bảo mật)
  - Phân biệt webservice và webapp truyền thống
    - webservice:
      Không có giao diện
      Tương tác giữa các ứng dụng web với nhau, qua các phương thức HTTP GET, POST, PUT, DELETE, PATCH, HTTP status code
      Kết quả trả về chỉ có data theo format XML hoặc json, người dùng khó đọc được
      Thông thường chỉ có các ứng dụng giao tiếp với nhau trong môi trường internet
    - web truyền thống:
      Có giao diện (view)
      Tương tác giữa user và ứng dụng thông qua form, link
      Kết quả trả về cho người dùng là view, hình ảnh, âm thanh
      Thông thường được tương tác giữa user và ứng dụng
- jQuery Ajax
  - • DOM(Document Object Model) là mô hình các đối tượng trong tài liệu HTML
    • DOM sẽ cung cấp cách truy cập hoặc thay đổi nội dung của tài liệu HTML
    • Javascript truy cập thành phần của DOM thông qua đối tượng có sẵn là document
    • jQuery là một thư viện được xây dựng từ javascript(write less, do more)
    - các tính năng
      - o Thao tác với DOM dễ dàng, nhanh, cách truy cập gần giống với css
      - o Xử lí anomation tạo hiệu ứng chuyển động: phóng lớn, di chuyển element.
      - o Hỗ trợ hầu hết các trình duyệt
      - o Hỗ trợ ajax
      - o Gọn nhẹ, chỉ tích hợp trong 1 file js
      - o Xử lí sự kiện
  - • Đồng bộ (Synchronous) là xử lí mà chương trình chạy theo từng bước, nghĩa là đoạn code trên thực hiện xong thì đoạn code dưới mới thực hiện, code chạy từ trên xuống dưới, trái qua phải.
    • Bất đồng bộ (Asynchronous) chương trình có thể nhảy cóc, đoạn code phía dưới có thể thực hiện mà không cần function, code trên thực hiện xong.
  - • Ajax (Asynchrounous Javascript and XML) là một kĩ thuật giúp phát triển các trang web động mà không cần tải lại trang
    o Tạo ra các trang web nhanh hơn, mượt hơn
    o Cập nhật nội dung của 1 phần trang web mà không cần tải lại trang
- I18N
  - • I18N là internationlization, quốc tế hóa, là quá trình thiết kế một ứng dụng phần mềm có thể đáp ứng được nhiều ngôn ngữ và khu vực khác nhau mà không cần thay đổi kĩ thuật
    - o Cấu hình LocalResolver: cấu hình nơi lưu, vd session, cấu hình ngôn ngữ mặc định: setDefaultLocale()
      o Cấu hình Interceptor: lọc request. Tham số làm tên lang sẽ thay đổi ngôn ngữ, tạo đối tượng interceptor, gọi phương thức setParameter("lang"), rồi addInterceptor.
      o Cấu hình messageSource để chứa các message cho từng ngôn ngữ, nên có 1 file message mặc định không có hậu tốt, hậu tố vd _vi, _ja…, cài đường dẫn thư mục message qua phương thức setBasename, cài bảng mã mặc định setDefaultEncoding("UTF-8")
  - • L10N localization, là quá trình điều chỉnh phần mềm đã quốc tế hóa về một ngôn ngữ địa phương/ khu vực nào đó
- Security
  - • Authentication (xác thực): là hành động nhằm thiết lập hoặc chứng thực một thông điệp/ đối tượng là đáng tin cậy
    • Authorization (phân quyền) là quá trình xác định xem đối tượng/ người dùng có quyền truy cập một tài nguyên cụ thể để thực hiện một số hành động CRUD hay không
  - cơ chế xác thực:
    - • HTTP basic: yêu cầu client cung cấp username và password để thực hiện xác thực
      • Cookie
      • Token: JWT, là ứng dụng webservice tiêu chuẩn mở, định nghĩa cách truyền tin an toàn giữa các thành viên bằng một đối tượng json:
      • Signature
      • One Time Password
      • Oauth 2: xác thực qua ứng dụng bên thứ 3
  - các thành phần
    o SecurityContext: là interface lưu trữ tất cả các thông tin chi tiết liên qua đến bảo mật ứng dụng.
    o SecurityContextHolder: cách truy cập SecurityContext
    o UserDetails: là một interface, đại diện cho một principal
    o Đối tượng Authentication: biểu diễn thông tin của principal
    o GrantedAuthority: là quyền được cấp cho principal, các quyền đều có tiền tố "ROLE_