

SPRING BOOT SECURITY

Nguyễn Huy Cường - nh.cuong@hutech.edu.vn

06/2024

Nội dung

- 1- Tổng quan
- 2- Các khái niệm
- 3- Cấu hình spring boot security
- 4- Demo
- 5- Security Dialect

1- Tổng quan

- **Spring Boot Security 6:** là một module của Spring Boot, giúp bảo mật ứng dụng web dễ dàng.
- Hoạt động: theo mô hình client-server. Khi một request được gửi đến ứng dụng web, sẽ được chuyển qua một chuỗi các bộ lọc (**filter chain**) do **Spring Security** quản lý. Mỗi bộ lọc có một nhiệm vụ cụ thể, như kiểm tra xác thực, kiểm tra phân quyền, điều hướng đến trang đăng nhập hoặc đăng xuất, xử lý các lỗi bảo mật...
- Các tính năng chính:
 - ❑ Xác thực người dùng (Authentication)
 - ❑ Kiểm soát quyền truy cập, ủy quyền (Authorized)
 - ❑ Bảo vệ API (JWT)
 - ❑ bảo vệ chống lại tấn công CSRF
 - ❑ Password Encoding/ Quản lý Session...

2- Các khái niệm

- Xác thực (**Authentication**): là một quá trình kiểm tra danh tính của một tài khoản đang vào trong hệ thống hiện tại thông qua một hệ thống xác thực.

Xác thực người dùng bằng:

basic: xác thực bằng cách gửi thông tin đăng nhập trong tiêu đề "Authorization".

form-based: đăng nhập bằng cách điền thông tin đăng nhập vào form được cung cấp

token-based: JWT (JSON WEB TOKEN), OAuth2

- Phân quyền (**Authorization**): là quá trình quyết định người dùng có quyền truy cập vào tài nguyên nào trong hệ thống.

Spring Security cung cấp nhiều cách để phân quyền:

❑ Sử dụng các annotation như: **@PreAuthorize** và **@Secured**

Ví dụ

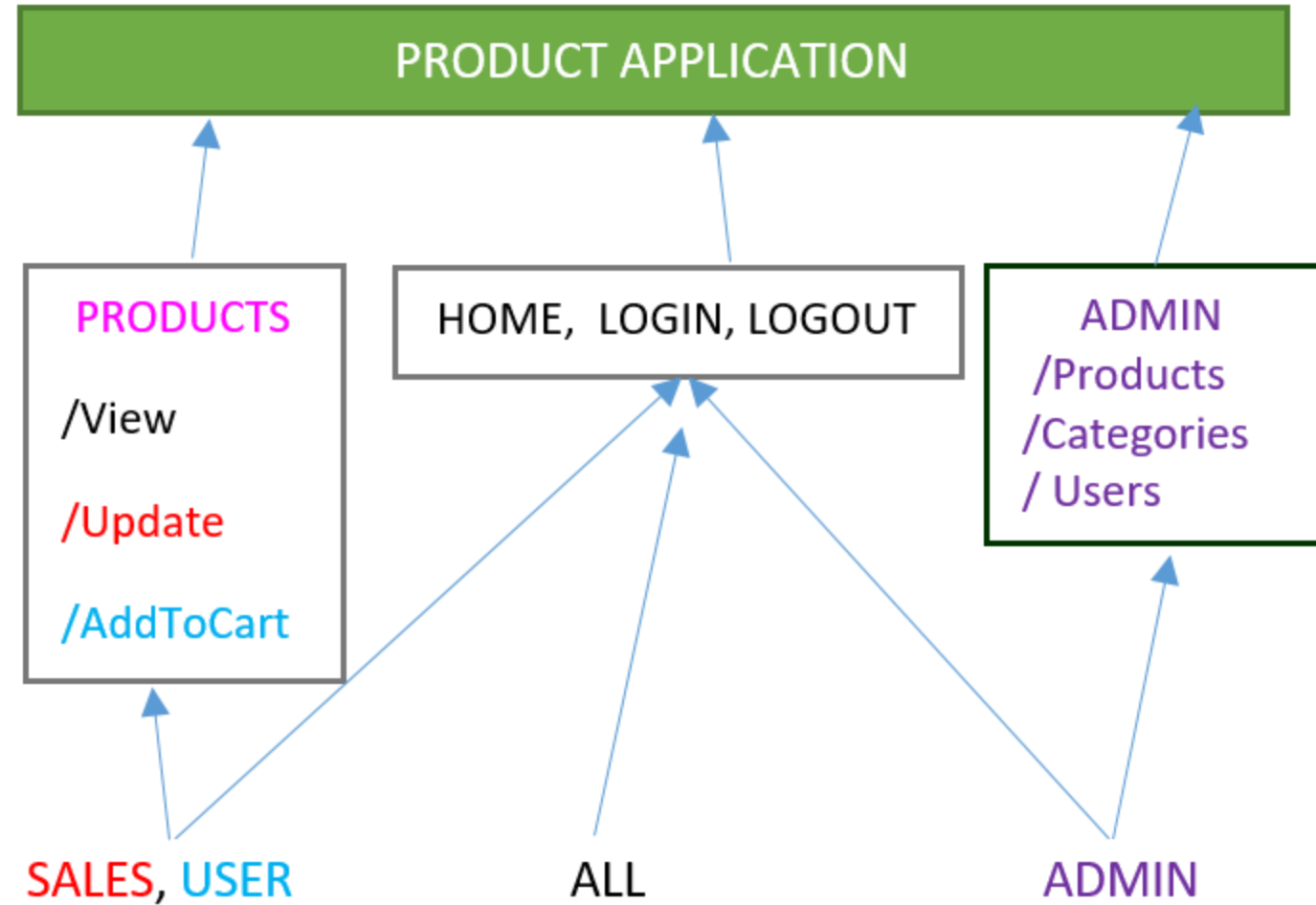
Hệ thống có : ADMIN, SALES, USER

ADMIN: Quản lý sản phẩm, danh mục, quản lý người dùng, đơn hàng...

USER: Xem sản phẩm (Product Viewing), Thêm vào giỏ hàng, quản lý giỏ hàng, đặt hàng..

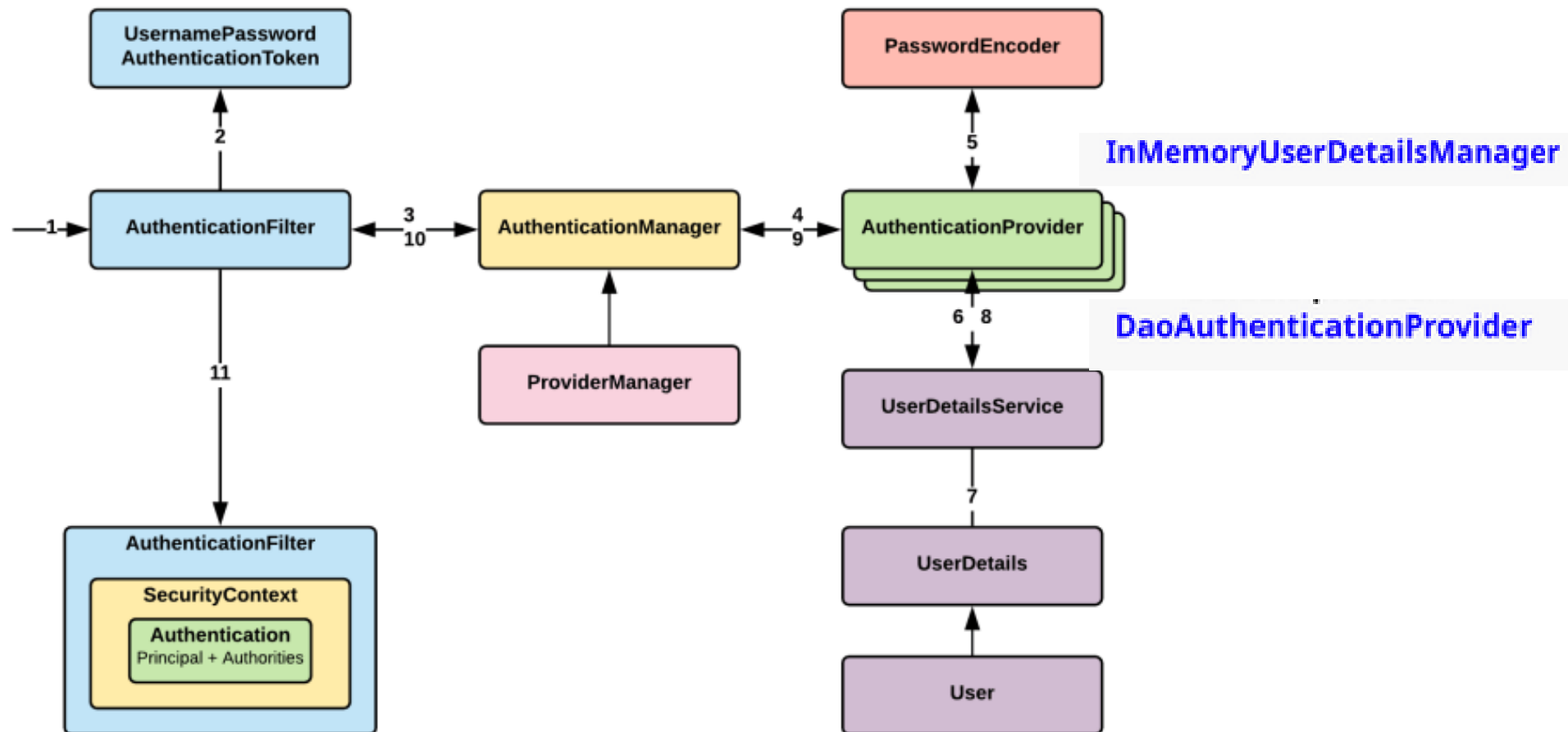
SALES: dành cho nhân viên bán hàng, có trách nhiệm quản lý quá trình bán hàng và chăm sóc khách hàng.

VD: xem sản phẩm, cập nhật sản phẩm, quản lý đơn hàng, tạo giảm giá...



3 - Cấu hình Spring boot Security

- (1) Thêm Spring Boot Security: dependency **spring-boot-starter-security** vào file **pom.xml**.
- (2) Cấu hình bảo mật: **@EnableWebSecurity** để kích hoạt cấu hình bảo mật.
- (3) Cơ chế hoạt động



Spring boot Security

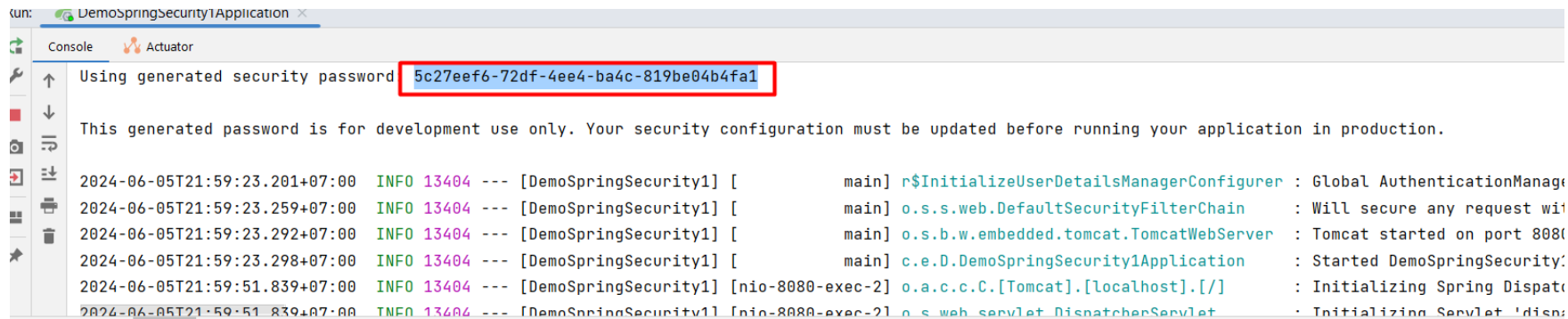
1- Thêm dependency

- Thêm Spring Boot Security vào dự án

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-security</artifactId>
</dependency>
```

Kết quả: Basic Authentication

1. Tự động generated: user/ [pass]



The screenshot shows the IDE console for 'DemoSpringSecurity1Application'. The 'Console' tab is active, displaying the following output:

```
Using generated security password 5c27eef6-72df-4ee4-ba4c-819be04b4fa1

This generated password is for development use only. Your security configuration must be updated before running your application in production.

2024-06-05T21:59:23.201+07:00 INFO 13404 --- [DemoSpringSecurity1] [main] r$InitializeUserDetailsManagerConfigurer : Global AuthenticationManager
2024-06-05T21:59:23.259+07:00 INFO 13404 --- [DemoSpringSecurity1] [main] o.s.s.web.DefaultSecurityFilterChain : Will secure any request with
2024-06-05T21:59:23.292+07:00 INFO 13404 --- [DemoSpringSecurity1] [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8080
2024-06-05T21:59:23.298+07:00 INFO 13404 --- [DemoSpringSecurity1] [main] c.e.D.DemoSpringSecurity1Application : Started DemoSpringSecurity1Application
2024-06-05T21:59:51.839+07:00 INFO 13404 --- [DemoSpringSecurity1] [nio-8080-exec-2] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet
2024-06-05T21:59:51.839+07:00 INFO 13404 --- [DemoSpringSecurity1] [nio-8080-exec-2] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
```

2. Có thể thay đổi tài khoản ở application.properties

```
spring.security.user.name=test
spring.security.user.password=test
```

Dependencies

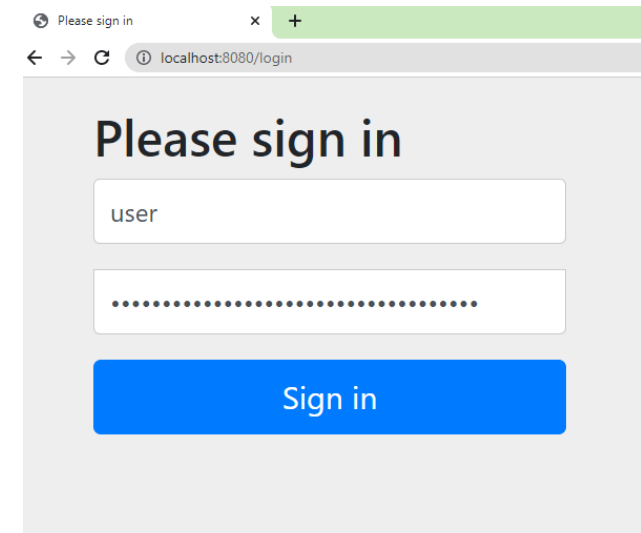
ADD DEPENDENCIES... CTRL + B

Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Security SECURITY

Highly customizable authentication and access-control framework for Spring applications.



Spring boot Security

2- @EnableWebSecurity & SecurityFilterChain

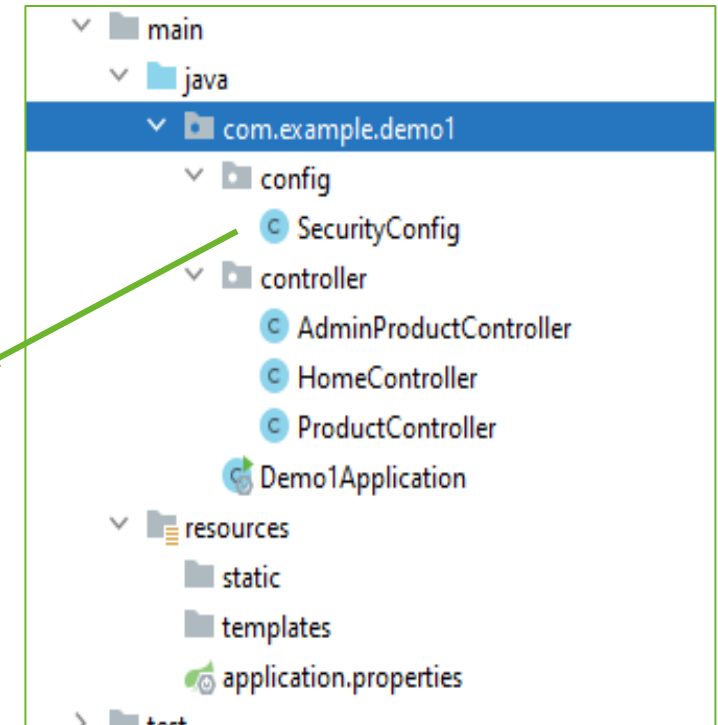
@Configuration

@EnableWebSecurity

public class SecurityConfig {

@Bean

```
public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {  
    http.authorizeRequests(authorize -> authorize  
        .requestMatchers("/home/**", "/login/**", "/logout/**").permitAll()  
        .requestMatchers("/products/**").hasAnyAuthority("USER", "SALES")  
        .requestMatchers("/admin/**").hasAnyAuthority("ADMIN")  
        .anyRequest().authenticated()  
        .formLogin(withDefaults())  
        // .httpBasic(withDefaults())  
        // .authenticationProvider(authenticationProvider())  
        ;  
    return http.build();  
}
```



3- Authentication Provider: In-Memory Authentication

❑ In-Memory Authentication (*phù hợp thử nghiệm ứng dụng*)

Phương pháp xác thực người dùng mà thông tin về người dùng (username, password, roles) được lưu trữ trực tiếp trong bộ nhớ của ứng dụng.

```
@Bean
public PasswordEncoder passwordEncoder() {
    return new BCryptPasswordEncoder();
}

@Bean
public UserDetailsService userDetailsService()
{
    var userAdmin = User.withUsername("admin")
        .password(passwordEncoder().encode("admin"))
        .authorities("ADMIN").build();
    var userSales = User.withUsername("sales")
        .password(passwordEncoder().encode("sales"))
        .authorities("SALES").build();
    var user = User.withUsername("user")
        .password(passwordEncoder().encode("user"))
        .authorities("USER").build();
    return new InMemoryUserDetailsManager(userAdmin, userSales, user);
}
```

3- Authentication Provider: DAO authenticationProvider

- ❑ **DAO authenticationProvider** (được sử dụng để xác thực người dùng dựa trên dữ liệu được lưu trữ trong CSDL hoặc dữ liệu có thể truy xuất thông tin người dùng)
 - Để sử dụng thì ở (2) phải gọi **authenticationProvider()**
 - Các bước cấu hình DAOAuthenticationProvider
- ❑ Tạo lớp triển khai **UserDetailsService** để ghi đè hàm **loadUserByUsername** lấy thông tin người dùng đưa vào đối tượng user trong Spring security.
- ❑ Cấu hình Security

```
@Autowired
private CustomUserDetailsService uds;

1 usage

@Bean
public AuthenticationProvider authenticationProvider() {
    DaoAuthenticationProvider authenticationProvider = new DaoAuthenticationProvider();
    authenticationProvider.setUserDetailsService(uds);
    authenticationProvider.setPasswordEncoder(passwordEncoder());
    return authenticationProvider;
}
```

Spring boot Security

3- Authentication Provider

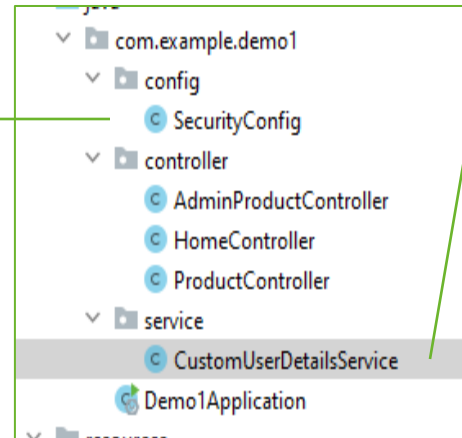
❑ Sử dụng DAO authenticationProvider

```

@Configuration
@EnableWebSecurity
public class SecurityConfig {
    @Autowired
    private CustomUserDetailsService uds;
    @Bean
    public AuthenticationProvider authenticationProvider()
    {
        DaoAuthenticationProvider authenticationProvider
        = new DaoAuthenticationProvider();
        authenticationProvider.setUserDetailsService(uds);

        authenticationProvider.setPasswordEncoder(passwordEncoder());
        return authenticationProvider;
    }
}

```



```

@Configuration
@EnableWebSecurity
public class SecurityConfig {
    @Bean
    public SecurityFilterChain filterChain(HttpSecurity http) throws
    Exception {
        .authenticationProvider(authenticationProvider());
        return http.build();
    }
}

```

```

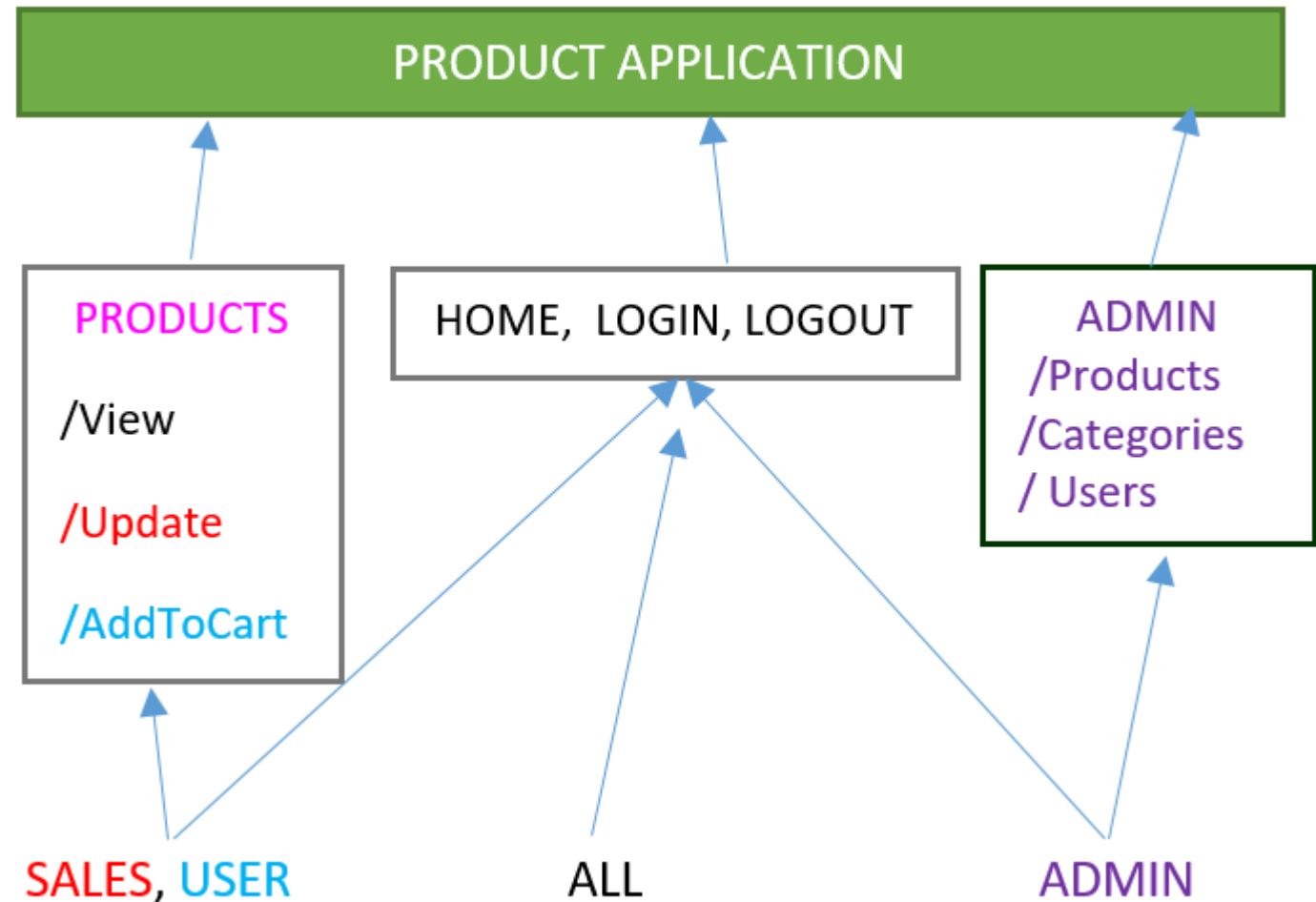
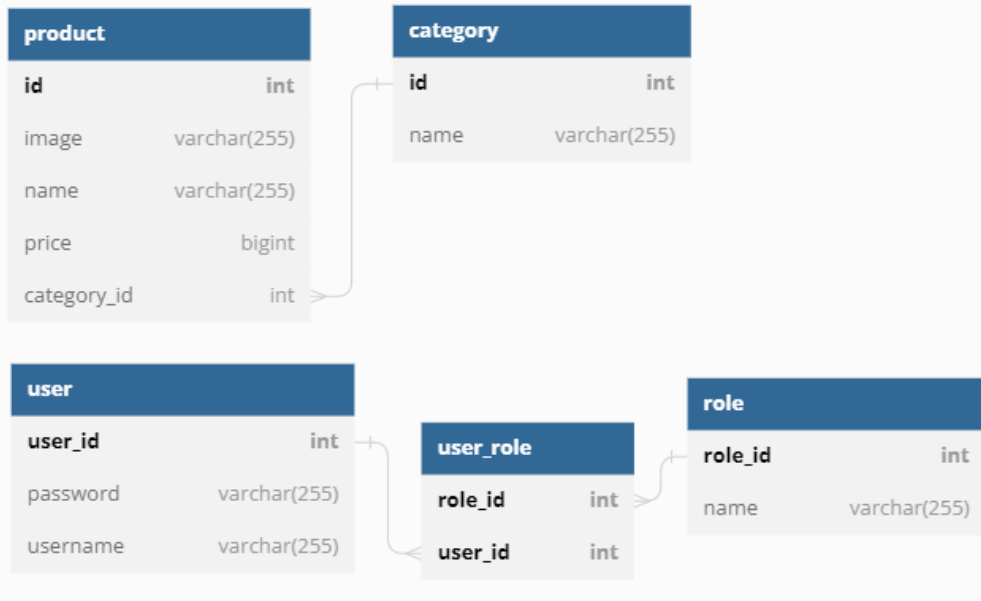
@Service
public class CustomUserDetailsService implements
UserDetailsService {
    @Override
    public UserDetails loadUserByUsername(String username)
    throws UsernameNotFoundException {
        String pass;
        Set<GrantedAuthority> authorities = new HashSet<>();

        if(username == "admin") {
            pass = "admin";
            authorities.add(new SimpleGrantedAuthority("ADMIN"));
        }
        else if(username == "sales") {
            pass = "sales";
            authorities.add(new SimpleGrantedAuthority("SALES"));
        }
        else {
            pass = "user";
            authorities.add(new SimpleGrantedAuthority("USER"));
        }
        return new org.springframework.security.core.userdetails.User(
            username,
            new BCryptPasswordEncoder().encode(pass),
            authorities
        );
    }
}

```



Demo Product Application: JPA + MySQL + Thymeleaf + Spring Boot Security



1. Thêm thư viện, cấu hình
2. Entity & Repository
3. Kích hoạt bảo mật
4. Service
5. **ADMIN**: Controller & Thymeleaf layout
6. Sales & USER: Products (view, update, AddToCart)

Demo

B1: dependencies & db connection

● pom.xml

```
<dependency>
  <groupId>org.thymeleaf.extras</groupId>
  <artifactId>thymeleaf-extras-springsecurity6</artifactId>
</dependency>
```

● application.properties

```
spring.datasource.url=jdbc:mysql://localhost:3306/DemoSpringSecurity
spring.datasource.username=root
spring.datasource.password=123456
#none, create-only, drop, create, create-drop, validate, update
spring.jpa.hibernate.ddl-auto=update
```

Dependencies

ADD DEPENDENCIES... CTRL + B

Spring Security

SECURITY

Highly customizable authentication and access-control framework for Spring applications.

—

Spring Web

WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

—

Spring Data JPA

SQL

Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

—

MySQL Driver

SQL

MySQL JDBC driver.

—

Validation

I/O

Bean Validation with Hibernate validator.

—

Thymeleaf

TEMPLATE ENGINES

A modern server-side Java template engine for both web and standalone environments. Allows HTML to be correctly displayed in browsers and as static prototypes.

—

Lombok

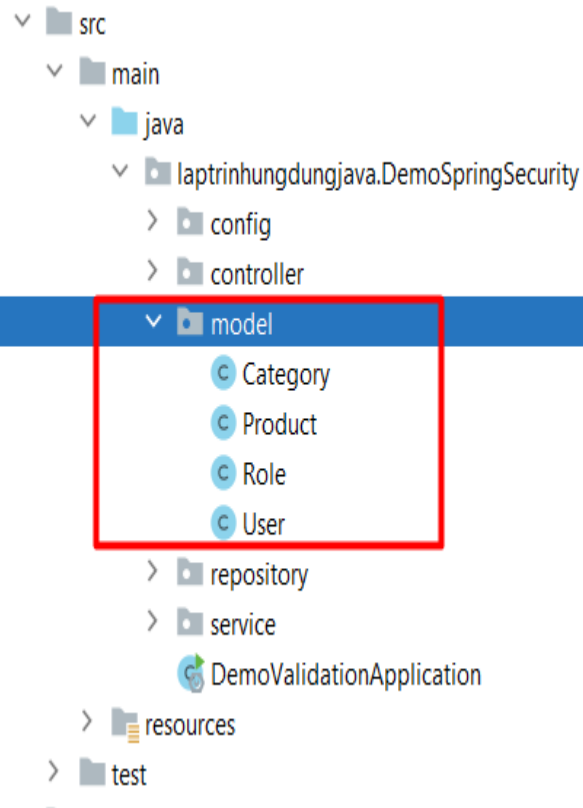
DEVELOPER TOOLS

Java annotation library which helps to reduce boilerplate code.

—

Demo

B2: JPA entity



```

@Entity
@Data
public class Product {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    @NotBlank(message = "Tên sản phẩm không được để trống")
    private String name;
    @Length(min= 0,max = 200, message = "Tên hình ảnh không quá 200 kí tự")
    private String image;
    @NotNull(message = "Giá sản phẩm không được để trống")
    @Min(value = 1, message = "Giá sản phẩm không được nhỏ hơn 1")
    @Max(value = 9999999, message = "Giá sản phẩm không được lớn hơn 9999999")
    private long price;
    @ManyToOne(fetch = FetchType.EAGER)
    @JoinColumn(name="categoryId")
    private Category category;
}

```

```

@Entity
@Data
public class Role {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int roleId;
    private String name;
    @ManyToMany(mappedBy = "roles")
    private Set<User> user = new HashSet<>();
}

```

```

@Entity
@Data
public class Category {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    private String name;
}

```

```

@Entity
@Data
public class User {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int userId;
    private String userName; //login Name
    private String name; //fullName
    private String password;
    @ManyToMany(fetch = FetchType.EAGER)
    @JoinTable(name = "UserRole",
        joinColumns = @JoinColumn(name = "userId"),
        inverseJoinColumns = @JoinColumn(name = "roleId"))
    private Set<Role> roles = new HashSet<>();
}

```

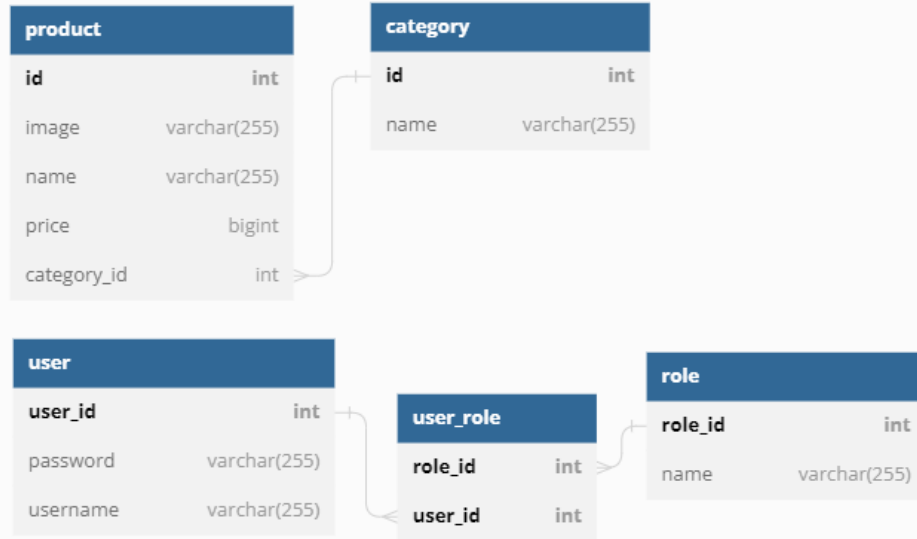
```

@Override
public boolean equals(Object o) {
    if (this == o) return true;
    if (o == null || getClass() != o.getClass()) return false;
    User user = (User) o;
    return userId == user.userId;
}
}

```

Demo

B2: Entity Repository

<https://bcrypt-generator.com/>


productdb.user: 3 rows total (approximately)

	user_id	password	username
	2	\$2a\$10\$JVeKCjBMRWPmg7nhwYrdOVZ3XQcBdx...	sales
	1	\$2a\$10\$O22UKYoAQ81IocycavtXTeS4AbkJm0Kk...	admin
	3	\$2a\$10\$ywRrcvVMRoK4Ma/obKz1qOMpbd7S1SD....	user

productdb.role: 3 rows total (approximately)

	role_id	name
	1	ADMIN
	2	SALES
	3	USER

productdb.user_role: 3 rows total (approximately)

	role_id	user_id
	1	1
	2	2
	3	3

@Repository

```

public interface UserRepository extends JpaRepository<User, Integer> {
    @Query("SELECT u FROM User u WHERE u.userName = :username")
    public User getUserByUsername(@Param("username") String username);
}
  
```

@Repository

```

public interface ProductRepository extends JpaRepository<Product, Integer> {
}
  
```

@Repository

```

public interface CategoryRepository extends JpaRepository<Category, Integer> {
}
  
```

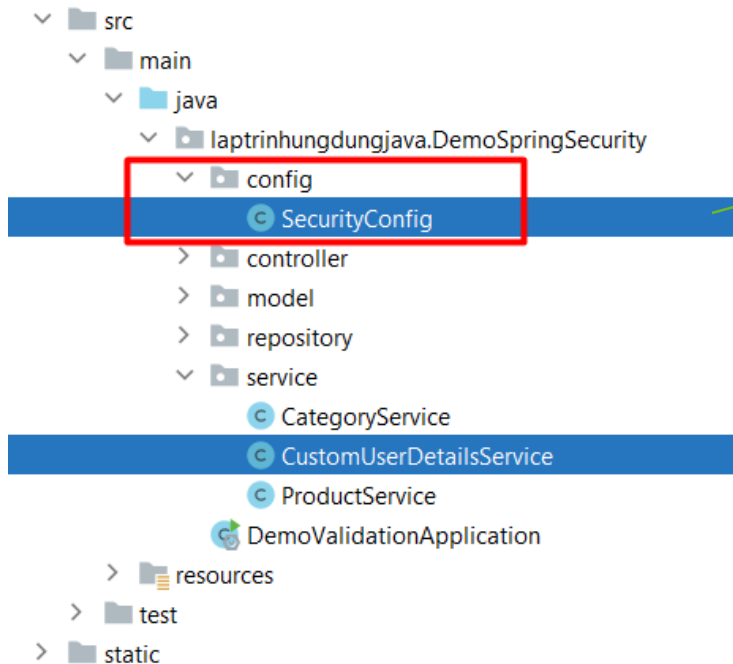

Demo

B3: Kích hoạt bảo mật

Sử dụng **@EnableWebSecurity** và triển khai

- **SecurityFilterChain**
- **AuthenticationProvider**

Sử dụng CustomUserDetailsService được implements ở tầng service (B4)



```
@Configuration
@EnableWebSecurity
public class SecurityConfig {
    @Bean
    public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {
        http.authorizeRequests(authorize -> authorize
            .requestMatchers("/home/**", "/login/**", "/logout/**").permitAll()
            .requestMatchers("/products/**").hasAnyAuthority("USER",
                "SALES")
            .requestMatchers("/admin/**").hasAnyAuthority("ADMIN")
            .anyRequest().authenticated()
        )
        .formLogin(withDefaults())
        .httpBasic(withDefaults())
        .authenticationProvider(authenticationProvider());
        return http.build();
    }

    @Autowired
    private CustomUserDetailsService uds;

    @Bean
    public PasswordEncoder passwordEncoder() {
        return new BCryptPasswordEncoder();
    }

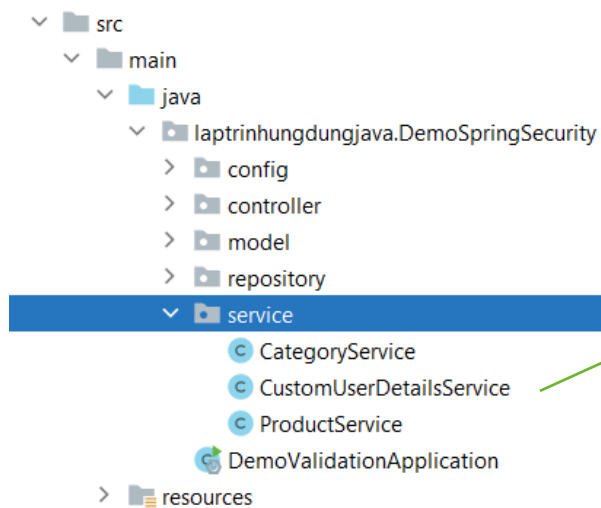
    @Bean
    public AuthenticationProvider authenticationProvider() {
        DaoAuthenticationProvider authenticationProvider = new
        DaoAuthenticationProvider();
        authenticationProvider.setUserDetailsService(uds);
        authenticationProvider.setPasswordEncoder(passwordEncoder());
        return authenticationProvider;
    }
}
```


Demo

B4: Triển khai service

- CategoryService, ProductService
- **CustomUserDetailsService** implements **UserDetailsService**

Tìm user & role từ CSDL để đưa vào **security.core.userdetails.User**

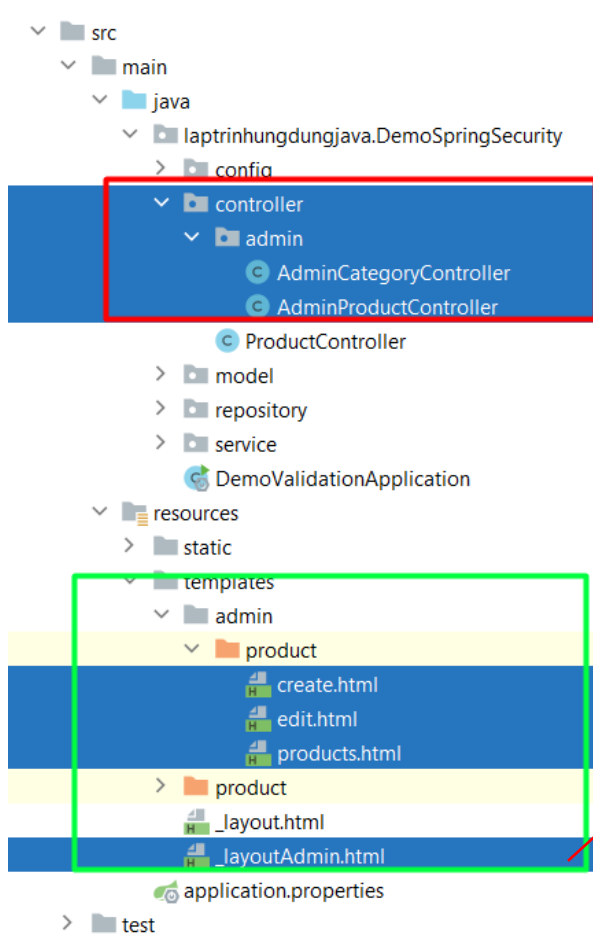


```
@Service
public class CustomUserDetailsService implements
UserDetailsService {
    @Autowired
    private UserRepository userRepository;
    @Override
    public UserDetails loadUserByUsername(String username)
throws UsernameNotFoundException {
        User u = userRepository.getUserByUsername(username);
        if (u == null) {
            throw new UsernameNotFoundException("Could not find
user");
        }
        Set<GrantedAuthority> authorities = u.getRoles().stream()
            .map((role) -> new
SimpleGrantedAuthority(role.getName()))
            .collect(Collectors.toSet());
        return new org.springframework.security.core.userdetails.User(
            username,
            u.getPassword(),
            authorities
        );
    }
}
```

Demo

B5: Admin Controller và template giao diện cho Admin

- **AdminProductController**: Quản lý sản phẩm
- **AdminCategoryController**: Quản lý danh mục



```
<!DOCTYPE html>
<html lang="en"
  xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta name="description" content="Demo - Home page with courses and search functionality">
  <title>Admin Pages</title>
  <!-- Bootstrap CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
    rel="stylesheet"
    integrity="sha384-
QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhY6hW+ALEwIH"
    crossorigin="anonymous">
</head>
<body>
  <nav class="navbar bg-dark border-bottom border-body" data-bs-theme="dark">
    <div style="padding-left: 40px">
      <a class="navbar-brand" th:href="@{/admin/products}">Admin</a>
      <a class="navbar-brand" th:href="@{/admin/products/create}">Products</a>
      <a class="navbar-brand" th:href="@{/admin/categories/create}">Categories</a>
    </div>
  </nav>
  <div layout:fragment="content" class="container body-content">
  </div>
  <div class="card-footer text-center">
    <hr/>
    <p>&copy; Nguyễn Huy Cường - Lập trình ứng dụng với Java</p>
  </div>
</body>
</html>
```

Demo

B5: AdminProductController

```

@Controller
@RequestMapping("/admin/products")
public class AdminProductController {
    @Autowired
    private ProductService productService;
    @Autowired
    private CategoryService categoryService;
    @GetMapping()
    public String Index(Model model)
    {
        model.addAttribute("listproduct", productService.getAll());
        return "admin/product/products";
    }
    @GetMapping("/create")
    public String Create(Model model) {
        model.addAttribute("product", new Product());
        model.addAttribute("categories", categoryService.getAll());
        return "admin/product/create";
    }
    @PostMapping("/create")
    public String Create(@Valid Product newProduct,
        BindingResult result,
        @RequestParam MultipartFile imageProduct,
        Model model) {
        if (result.hasErrors()) {
            model.addAttribute("product", newProduct);
            model.addAttribute("categories", categoryService.getAll());
            return "admin/product/create";
        }
        productService.updateImage(newProduct, imageProduct);
        productService.add(newProduct);
        return "redirect:/admin/products";
    }
}

```

```

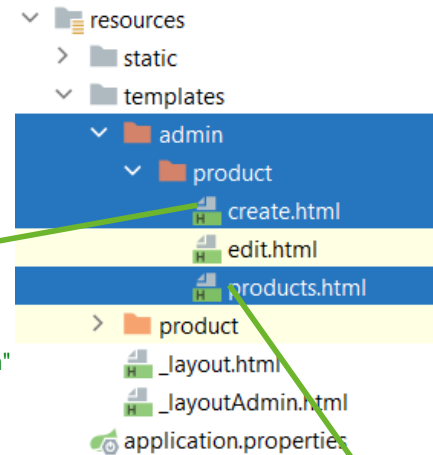
@GetMapping("/edit/{id}")
public String Edit(@PathVariable int id, Model model) {
    Product find = productService.get(id);
    if(find == null)
        throw new IllegalStateException("Product not found with ID: " + id);
    model.addAttribute("product", find);
    model.addAttribute("categories", categoryService.getAll());
    return "admin/product/edit";
}
@PostMapping("/edit")
public String Edit(@Valid Product editProduct,
    BindingResult result,
    @RequestParam MultipartFile imageProduct,
    Model model) {
    if (result.hasErrors()) {
        model.addAttribute("product", editProduct);
        model.addAttribute("categories", categoryService.getAll());
        return "admin/product/edit";
    }
    productService.updateImage(editProduct, imageProduct);
    productService.update(editProduct);
    return "redirect:/admin/products";
}
//Delete , etc..
}

```

Demo

B5: thymeleaf Admin/Products

```
<!DOCTYPE html>
<html lang="en"
  xmlns:th="http://www.thymeleaf.org" xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"
  layout:decorate="_layoutAdmin">
<head>
  <meta charset="UTF-8">
  <title>Create product</title>
</head>
<body>
<div layout:fragment="content" class="container body-content">
  <form th:action="@{admin/products/create}" th:object="${product}" method="post" class="form"
  enctype="multipart/form-data">
    <div class="form-group">
      <label for="name">Name</label>
      <input class="form-control" type="text" th:field="*{name}" id="name" placeholder="Product
Name">
      <div class="alert alert-warning" th:if="${#fields.hasErrors('name')}" th:errors="*{name}"></div>
    </div>
    <div class="form-group">
      <label for="image">Image</label>
      <input class="form-control-file" type="file" id="image" name="imageProduct"
accept="image/png,image/jpeg">
      <div class="alert alert-warning" th:if="${#fields.hasErrors('image')}" th:errors="*{image}"></div>
    </div>
    <div class="form-group">
      <label for="price">Price</label>
      <input class="form-control" type="number" th:field="*{price}" id="price" placeholder="price">
      <div class="alert alert-warning" th:if="${#fields.hasErrors('price')}" th:errors="*{price}"></div>
    </div>
    <div class="form-group">
      <label for="category" class="form-label">Category</label>
      <select th:field="*{category}" class="form-control" id="category">
        <option th:each="category : ${categories}"
          th:value="${category.id}" th:text="${category.name}"></option>
      </select>
      <div class="alert alert-warning" th:if="${#fields.hasErrors('category')}"
th:errors="*{category}"></div>
    </div>
    <input type="submit" class="btn btn-success" value="Add Product">
  </form>
</div></body></html>
```



```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org"
  xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"
  layout:decorate="_layoutAdmin"
  xmlns:custom="http://www.w3.org/1999/xhtml">
<head>
  <meta charset="UTF-8">
  <title>List products</title>
</head>
<body>
<div layout:fragment="content" class="container body-content">
  <a th:href="@{admin/products/create}" class="btn btn-primary">Create New Product</a>
  <table class="table table-striped">
    <thead>
      <tr>
        <th scope="col">#</th>
        <th scope="col">Name</th>
        <th scope="col">Image</th>
        <th scope="col">Price</th>
        <th scope="col">Category Name</th>
        <th scope="col"></th>
      </tr>
    </thead>
    <tbody>
      <tr th:each="product : ${listproduct}" >
        <th scope="row" th:text="${product.id}"></th>
        <td th:text="${product.name}"></td>
        <td>
          
        </td>
        <td th:text="${product.price}"></td>
        <td th:text="${product.category?.name ?: ''}"></td>
        <td>
          <a th:href="@{/admin/products/edit/{id}(id=${product.id})}"
custom:linkMethod="post" class="btn btn-secondary">Edit</a>
          <a th:href="@{/admin/products/delete/{id}(id=${product.id})}"
custom:linkMethod="post" class="btn btn-danger">Delete</a>
        </td>
      </tr>
    </tbody>
  </table>
</div>
</body>
</html>
```

DEMO

Kết quả khi đăng nhập bằng admin

localhost:8080/login

Please sign in

localhost:8080/admin/products

Admin Products Categories

Create New Product

#	Name	Image	Price	Category Name	
1	SP1		111	Laptop	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
2	San pham 2		122	Điện thoại	<input type="button" value="Edit"/> <input type="button" value="Delete"/>

- Tương tự thực hiện các chức năng ở categories

© Nguyễn Huy Cường - Lập trình ứng dụng với Java

localhost:8080/admin/products/edit/2

Admin Products Categories

Name

Image:

Change Image: Không có tệp nào được chọn

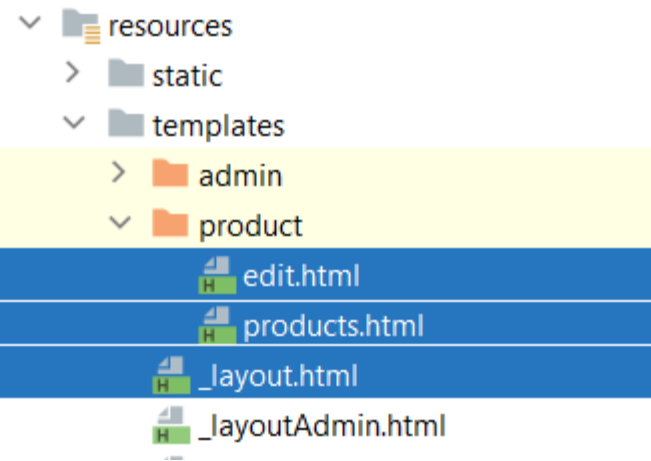
Price:

Category:

Demo

B6: Product Controller và template giao diện cho Sales, User

- **ProductController:** Quản lý sản phẩm





```
<!DOCTYPE html>
<html lang="en"
  xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <title>Demo Products</title>
  <!-- Bootstrap CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
    rel="stylesheet"
    integrity="sha384-
  QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhY6hW+ALEwIH"
    crossorigin="anonymous">
</head>
<body>
  <nav class="navbar bg-dark border-bottom border-body" data-bs-theme="dark">
    <div style="padding-left: 40px">
      <a class="navbar-brand" th:href="@{/products}">Home</a>
      <a class="navbar-brand" th:href="@{/products}">Products</a>
    </div>
  </nav>
  <div layout:fragment="content" class="container body-content">
  </div>
  <div class="card-footer text-center">
    <hr/>
    <p>&copy; Nguyễn Huy Cường - Lập trình ứng dụng với Java</p>
  </div>
</body>
</html>
```

Demo

B6: Product Controller

- **ProductController:** Quản lý sản phẩm

Home Products					
#	Name	Image	Price	Category Name	
1	SP1		111	Laptop	<button>Edit</button> <button>Add To Cart</button>
2	San pham 2		2024	Laptop	<button>Edit</button> <button>Add To Cart</button>

© Nguyễn Huy Cường - Lập trình ứng dụng với Java

```

@Controller
@RequestMapping("/products")
public class ProductController {
    @Autowired
    private ProductService productService;
    @Autowired
    private CategoryService categoryService;
    @GetMapping()
    public String index(Model model) {
        model.addAttribute("listproduct", productService.getAll());
        return "product/products";
    }
    @GetMapping("/edit/{id}")
    public String edit(@PathVariable int id, Model model) {
        Product product = productService.get(id);
        if (product == null) {
            throw new IllegalStateException("Product not found with ID: " + id);
        }
        model.addAttribute("product", product);
        model.addAttribute("categories", categoryService.getAll());
        return "product/edit";
    }
    @PostMapping("/edit")
    public String edit(@Valid Product editProduct,
        BindingResult result,
        @RequestParam("imageProduct") MultipartFile imageProduct,
        Model model) {
        if (result.hasErrors()) {
            model.addAttribute("product", editProduct);
            model.addAttribute("categories", categoryService.getAll());
            return "product/edit";
        }
        productService.updateImage(editProduct, imageProduct);
        productService.update(editProduct);
        return "redirect:/products";
    }
}

```


5 - Security Dialect

- Tích hợp giữa **Spring Security** và **Thymeleaf**, giúp việc triển khai tính năng xác thực và phân quyền

```
<dependency>
  <groupId>org.thymeleaf.extras</groupId>
  <artifactId>thymeleaf-extras-springsecurity6</artifactId>
  <version>3.1.2.RELEASE </version>
</dependency>
```

```
<html lang="en"
  xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"
  xmlns:th="http://www.thymeleaf.org"
  xmlns:sec="http://www.thymeleaf.org/extras/spring-security">
```

- Sử dụng: 1 số thuộc tính

❑ sec:authorize

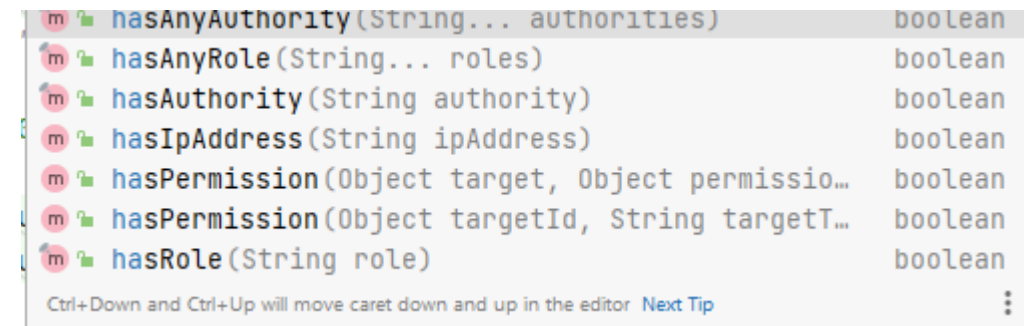
sec:authorize="isAuthenticated()"

sec:authorize="isAnonymous()"

sec:authorize="hasAnyAuthority('ADMIN')"

❑ sec:authentication

sec:authentication="name">







```
<div style="padding-right: 20px">
  <span sec:authorize="isAuthenticated()" class="navbar-brand">
    <span sec:authentication="name" ></span> |
    <a sec:logout th:href="@{/logout}">Logout</a>
  </span>
  <span sec:authorize="!isAuthenticated()" class="navbar-brand">
    <a sec:authorize="isAnonymous()"
th:href="@{/login}">Login</a>
  </span>
</div>
```

Admin Products Categories

admin | [Logout](#)

Create New Product

#	Name	Image	Price	Category Name	
1	SP1		111	Laptop	Edit Delete
2	San pham 2		2024	Laptop	Edit Delete

^^





Sales: Edit, User: AddToCart

```
<td>
  <span sec:authorize="hasAnyAuthority('SALES')">
    <a th:href="@{/products/edit/{id}(id=${product.id})}" custom:linkMethod="post" class="btn btn-secondary">Edit</a>
  </span>
  <span sec:authorize="hasAnyAuthority('USER')">
    <a th:href="@{/products/AddToCart/{id}(id=${product.id})}" custom:linkMethod="post" class="btn btn-danger">Add To Cart</a>
  </span>
</td>
```

→ localhost:8080/products

Home Products sales | [Logout](#)

#	Name	Image	Price	Category Name	
1	SP1		111	Laptop	Edit
2	San pham 2		2024	Laptop	Edit



Sales: Edit, User: AddToCart



#	Name	Image	Price	Category Name	
1	SP1		111	Laptop	Edit
2	San pham 2		2024	Laptop	Edit

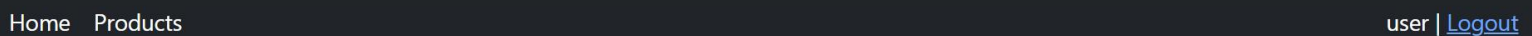
no usages



@Configuration

@EnableWebSecurity

@EnableGlobalMethodSecurity(prePostEnabled = true)

public class SecurityConfig {



#	Name	Image	Price	Category Name	
1	SP1		111	Laptop	Add To Cart
2	San pham 2		2024	Laptop	Add To Cart

```
<td>
  <span sec:authorize="hasAnyAuthority('SALES')">
    <a th:href="@{/products/edit/{id}(id=${product.id})}"
      custom:linkMethod="post" class="btn btn-
      secondary">Edit</a>
  </span>
  <span sec:authorize="hasAnyAuthority('USER')">
    <a
      th:href="@{/products/AddToCart/{id}(id=${product.id})}"
      custom:linkMethod="post" class="btn btn-danger">Add
      To Cart</a>
  </span>
</td>
```

@GetMapping("/edit/{id}")

@PreAuthorize("hasAuthority('SALES')")

public String edit(@PathVariable int id, Model model) {...}

no usages

@PostMapping("/edit")

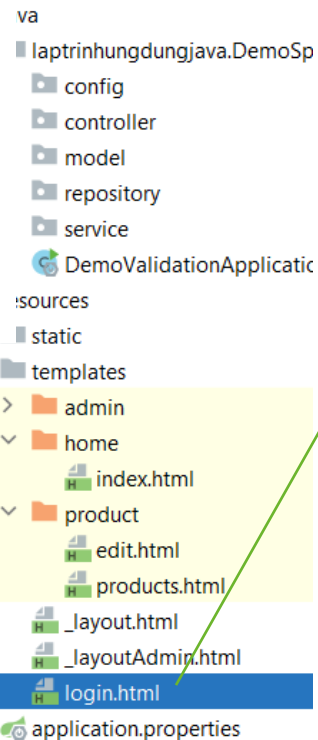
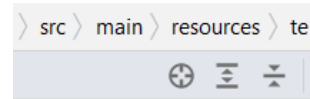
@PreAuthorize("hasAuthority('SALES')")

public String edit(@Valid Product editProduct,
 BindingResult result,
 @RequestParam("imageProduct") MultipartFile imageProduct,
 Model model) {...}

Custom login, logout

@Bean

```
public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {
    http.authorizeRequests(authorize -> authorize
        .requestMatchers("/", "/home/**", "/login", "/logout").permitAll()
        .requestMatchers("/products/**").hasAnyAuthority("USER", "SALES")
        .requestMatchers("/admin/**").hasAnyAuthority("ADMIN")
        .anyRequest().authenticated()
    )
    .formLogin(formLogin -> formLogin
        .loginPage("/login")
        .loginProcessingUrl("/login")
        .defaultSuccessUrl("/home", true)
        .failureUrl("/login?error")
        .permitAll()
    )
    .logout((logout) -> logout.permitAll())
    .httpBasic(withDefaults())
    .authenticationProvider(authenticationProvider());
    return http.build();
}
```

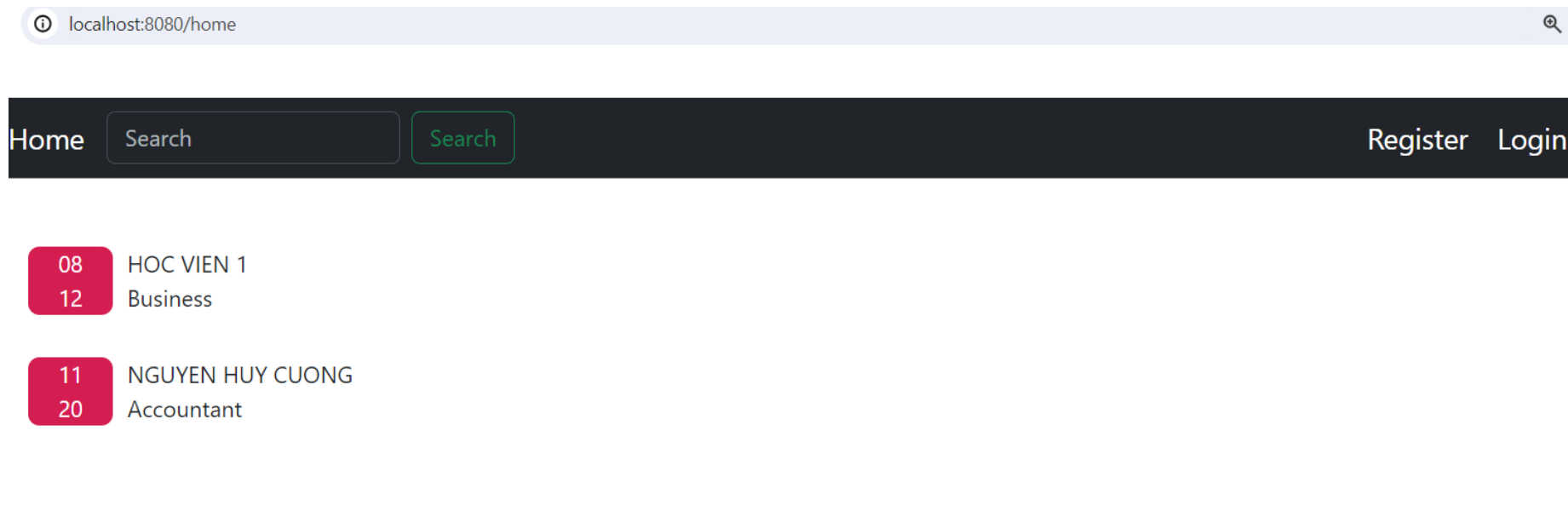


```
<!DOCTYPE html>
<html lang="en"
    xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"
    xmlns:th="http://www.thymeleaf.org"
    layout:decorate="_layout">
<head>
    <meta charset="UTF-8">
    <title>Trang chủ</title>
</head>
<body>
    <div layout:fragment="content" class="container body-content">
        <form th:action="@{/login}" method="post" style="max-width: 350px; margin: 0 auto;">
            <div th:if="${param.error}">
                <p class="text-danger">[[${session.SPRING_SECURITY_LAST_EXCEPTION.message}]]</p>
            </div>
            <div th:if="${param.logout}">
                <p class="text-warning">You have been logged out.</p>
            </div>
            <div class="border border-secondary p-3 rounded">
                <p>Access to Product system</p>
                <p>
                    <input type="text" name="username" class="form-control" required autofocus/>
                </p>
                <p>
                    <input type="password" name="password" class="form-control"
                    placeholder="Password" required />
                </p>
                <p>
                    <input type="checkbox" name="remember-me" /> &nbsp; Remember Me
                </p>
                <p>
                    <input type="submit" value="Login" class="btn btn-primary" />
                </p>
            </div>
        </form>
    </div>
</body>
</html>
```

ASM7: BIGSCHOOL – Đăng ký, đăng nhập

Yêu cầu 1: Khi chưa đăng nhập

- ❑ Khi chưa cần đăng nhập có thể truy xuất: **/login /logout /home /register**
- ❑ Trang chủ hiện thị **register** / Login
- ❑ Home: các khóa học sắp tới (chỉ xem - không edit, delete)



© Nguyễn Huy Cường - Lập trình ứng dụng với Java

- ❑ Thực hiện Register để đăng kí thông tin cho User (id, email, name, password, user_name)
- ❑ Thực hiện Login với **user name/ password** (sẽ sang yêu cầu 2)

ASM7: BIGSCHOOL – Đăng nhập

Yêu cầu 2: Đăng nhập

- ❑ Lấy thông tin đăng nhập được lấy từ bảng **users** (**user_name** / **password**)

Search Register **Login**

Access to BigSchool system

nguyenhuycuong

.....

☐ Remember Me

Login

gschool2.users: 3 rows total (exact)

	id	email	name	password	user_name
1	ŽÆz•GÊ\$ªR%...	user1@gmail.com	HOC VIEN 1	\$2a\$12\$TqnQLXqgv/Hxk/ARKGCoB02tk3VUFEBfEDYw3zPiWbHJqXu4PULcS	user1
2	z]Ò`□àCô£q□...	nd.anh@gmail.com	NGUYEN DINH ANH	\$2a\$12\$TqnQLXqgv/Hxk/ARKGCoB02tk3VUFEBfEDYw3zPiWbHJqXu4PULcS	nguyendinhanh
3	Đ□Q'H□O□¶...	nh.cuong@hutech.edu.vn	NGUYEN HUY CUONG	\$2a\$12\$TqnQLXqgv/Hxk/ARKGCoB02tk3VUFEBfEDYw3zPiWbHJqXu4PULcS	nguyenhuycuong

Lấy thông tin **name** đã đăng nhập của giảng viên ở góc phải menu

localhost:8080/home

Home **Add Course** Search Search

NGUYEN HUY CUONG ▾

ASM7: BIGSCHOOL – Đăng ký khóa học

Yêu cầu 3: Đăng ký khóa học

- ❑ **Add Course:** với giảng viên là user đã đăng nhập.
- ❑ Xóa **lecture_name** đã lưu ở course.
- ❑ Khi tạo khóa học: **lecture_id** = **users.id** (id của user đã đăng nhập)
- ❑ Sửa lại các chức năng liên quan (home, edit)

The screenshot displays the BigSchool application interface for creating a new course. The browser address bar shows `localhost:8080/courses/create`. The application header includes a 'Home' link, a search bar, and the logged-in user 'NGUYEN HUY CUONG'. The main form, titled 'Thông tin khóa học', contains fields for 'Place' (Thủ Đức), 'Start Date' (20/11/2025 08:30), and 'Category' (Accountant). A green arrow points from the 'Add Course' button to the database table view.

The database table view shows the `bigschool2.course` table with 2 rows. The second row is highlighted with a red box, showing the following data:

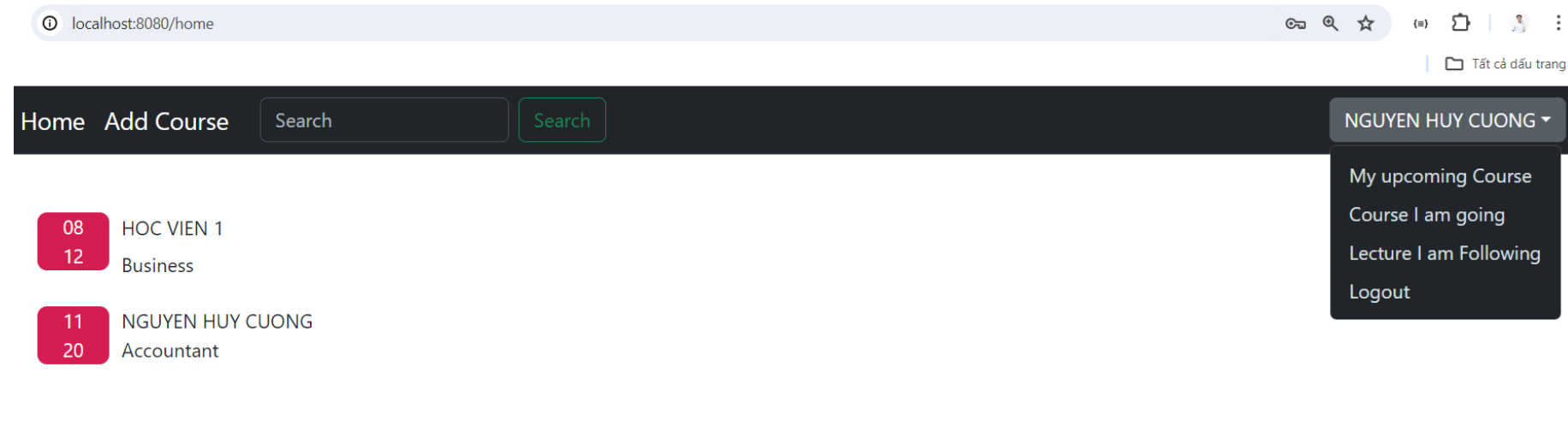
#	id	place	start_date	category_id	lecture_id
1	1	BÌNH DUONG	2024-08-12 08:30:31.000000	1	Ảz•GÊs3IR%Ả^q
2	3	Thủ Đức	2025-11-20 08:30:00.000000	3	Đ□Q'H□O□□qãô¼%ok

A green arrow also points from the 'Category' field (Accountant) to the user list on the right, which shows 'NGUYEN HUY CUONG' with ID 11 and role Accountant.

ASM7: BIGSCHOOL – dropdown item

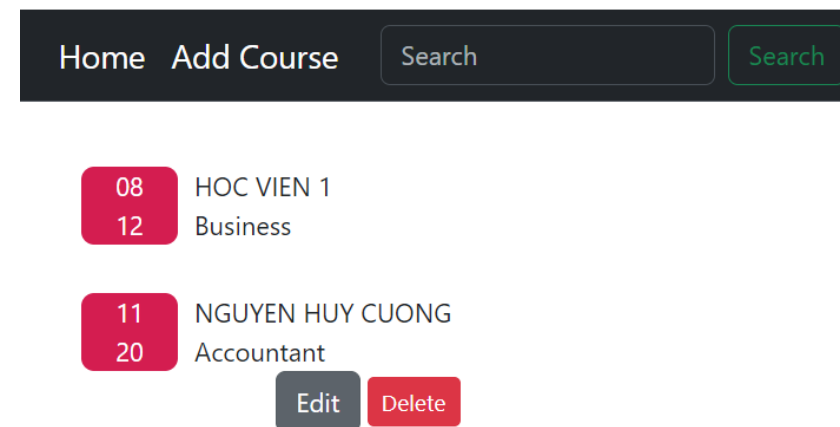
Yêu cầu 4: Thể hiện các dropdown item khi chọn từ tên đăng nhập

- ☐ My upcoming Course
- ☐ Course I am going
- ☐ Lecture I am Following
- ☐ Logout



© Nguyễn Huy Cường - Lập trình ứng dụng với Java

- ☐ Khi hover vào các khóa học (đã đăng nhập)
- Hiển thị Edit/ Delete tương ứng



ASM7: BIGSCHOOL – Khóa học của tôi

- Yêu cầu 5: My Upcoming courses : khóa học của tôi
Lấy thông tin của các khóa học do giảng viên đăng nhập tạo ra

HuyCuong Database: bigschool2 Table: users Data Query

bigschool2.users: 3 rows total (exact)

#	id	email	name	password	user_name
1	ŽÆz•GÊš@R%Ã~q	user1@gmail.com	HOC VIEN 1	\$2a\$12\$TqnQLXqgy/Hbk/ARKGCoB02tk3VUFEbf...	user1
2	z]0'□àC0£q□□†Ý3^	nd.anh@gmail.com	NGUYEN DINH ANH	\$2a\$12\$TqnQLXqgy/Hbk/ARKGCoB02tk3VUFEbf...	nguyendinhanh
3	Đ□Q'H□O□¶¶ăô¼%ok	nh.cuong@hutech.edu.vn	NGUYEN HUY CUONG	\$2a\$12\$TqnQLXqgy/Hbk/ARKGCoB02tk3VUFEbf...	nguyenhuycuong

HuyCuong Database: bigschool2 Table: course Data Query

bigschool2.course: 2 rows total (exact)

#	id	place	start_...	lecture_id
1	1	BINH DUONG	2024-...	ŽÆz•GÊš@R%Ã~q
2	3	Thủ Đức	2025-...	Đ□Q'H□O□¶¶ăô¼%ok: nh.cuong@hutech.edu.vn

localhost:8080/courses/mine

Home Add Course Search Search

Mine

11 NGUYEN HUY CUONG
20 Accountant

NGUYEN HUY CUONG ▾

- My upcoming Course
- Course I am going
- Lecture I am going
- Logout

Home Add Course Search Search

Mine

08 HOC VIEN 1
12 Business

HOC VIEN 1 ▾

- My upcoming Course
- Course I am going
- Lecture I am going
- Logout

ASM7: BigSchool – Tham gia khóa học

Yêu cầu 6: Trạng thái tham gia khóa học: Going? / Going

Home: thể hiện trạng thái user này có tham gia khóa học

TH1: Going? Khi khóa học không có dữ liệu ở bảng **Attendance**

[Home](#)
[Add Course](#)

NGUYEN DINH ANH ▾

08
12

HOC VIEN 1
Business

Going?

11
20

NGUYEN HUY CUONG
Accountant

Going?

HuyCuong Database: bigschool2 Table: users

bigschool2.users: 3 rows total (exact)

#	id	email	name	password	user_name
1	z]0'□àC6Eq□□+Y3^	user1@gmail.com	HOC VIEN 1	\$2a\$12\$TqnQLXqgv/Hxk/ARKGCoBO2tk3VUFEbf...	user1
2	z]0'□àC6Eq□□+Y3^	nd.anh@gmail.com	NGUYEN DINH ANH	\$2a\$12\$TqnQLXqgv/Hxk/ARKGCoBO2tk3VUFEbf...	nguyendinhanh
3	Đ□Q'H□O□□□ä6¼%ok	nh.cuong@hutech.edu.vn	NGUYEN HUY CUONG	\$2a\$12\$TqnQLXqgv/Hxk/ARKGCoBO2tk3VUFEbf...	nguyenhuycuong

TH2: Going Khi khóa học có dữ liệu tham gia ở bảng **Attendance**

[Home](#)
[Add Course](#)

NGUYEN DINH ANH ▾

08
12

HOC VIEN 1
Business

Going?

11
20

NGUYEN HUY CUONG
Accountant

Going

HuyCuong Database: bigschool2 Table: attendance

bigschool2.attendance: 1 rows total (exact)

#	course_id	attendee
1	3	z]0'□àC6Eq□□+Y3^: nd.anh@gmail.com

ASM7: BigSchool – Tham gia khóa học

Yêu cầu 7: Tham gia khóa học

TH1: Khi click **Going?**

- Thêm 1 dòng vào bảng Attendance với

Attendee là mã thông tin đăng nhập

Course_id: mã khóa học

- Khi trở lại Going? Sẽ trở thành Going

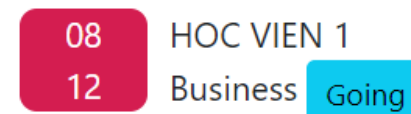
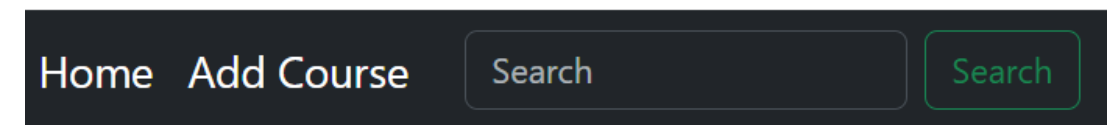
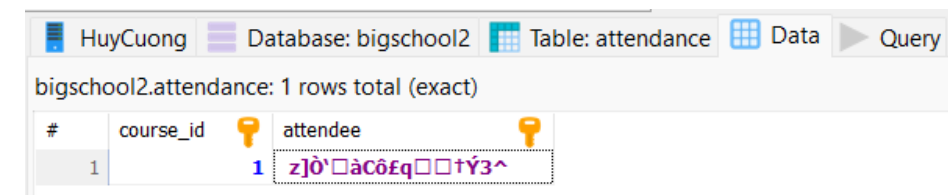
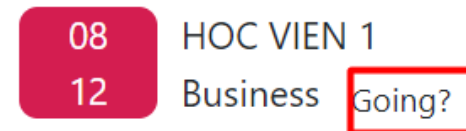
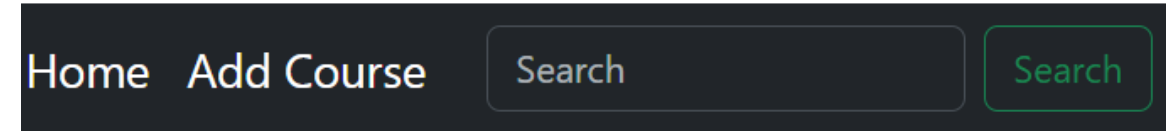
(Đã tham gia khóa học)

TH2: Khi click Going (ngược lại)

- Xóa dữ liệu ở bảng Attendance
- Trở về Going?

Lưu ý: Không hiện thị Going hoặc Going?

Đối với những khóa học của giảng viên này tạo ra (y/c 9)



ASM7: BigSchool – Khóa học đã tham gia

Yêu cầu 7.2: Course I am going: lấy tất cả khóa học mà **user** đăng nhập đang tham gia (có dữ liệu ở bảng Attendance)

HuyCuong Database: bigschool2 Table: attendance Data Query

bigschool2.attendance: 2 rows total (exact)

#	course_id	attendee
1	3	user1@gmail.com
2	1	z]ô' àCôEq □ □ †Ý3^

localhost:8080/courses/attending

Home Add Course Search Search

HOC VIEN 1 ▾

- My upcoming Course
- Course I am going**
- Lecture I am Following
- Logout

Attendance

11 NGUYEN HUY CUONG
20 Accountant **Going**

ASM7: BigSchool – Theo dõi giảng viên

Yêu cầu 8

- ❑ Tạo bảng **Following**(**follower_id**, **followee_id**): thể hiện follower_id theo dõi các khóa học của user có id là followee_id

The screenshot shows a database management interface for a database named 'bigschool2'. The 'following' table is selected, and its structure is displayed in the 'Columns' section. The table has two columns: 'follower_id' and 'followee_id', both of type 'BINARY' with a length of 16. Both columns are unsigned, allow NULL, and have no default values or comments.

The 'Foreign keys' section shows two foreign key constraints:

Key name	Columns	Reference table	Foreign col...	On UPDATE	On DELETE
FK7o8rmeuf83dqi0b2bvdhuwo9g	follower_id	bigschool2.users	id	NO ACTION	NO ACTION
FKkbpwn1k4c22go17juoxkk4007	followee_id	bigschool2.users	id	NO ACTION	NO ACTION

The 'Columns' section also displays the following table structure:

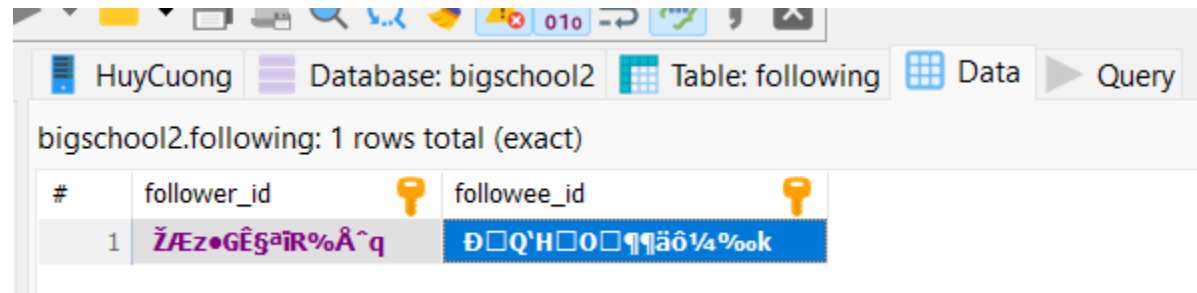
#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default	Comment	Coll
1	follower_id	BINARY	16	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default		
2	followee_id	BINARY	16	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default		

ASM7: BigSchool – Theo dõi giảng viên

Yêu cầu 8: Lecture I am Following

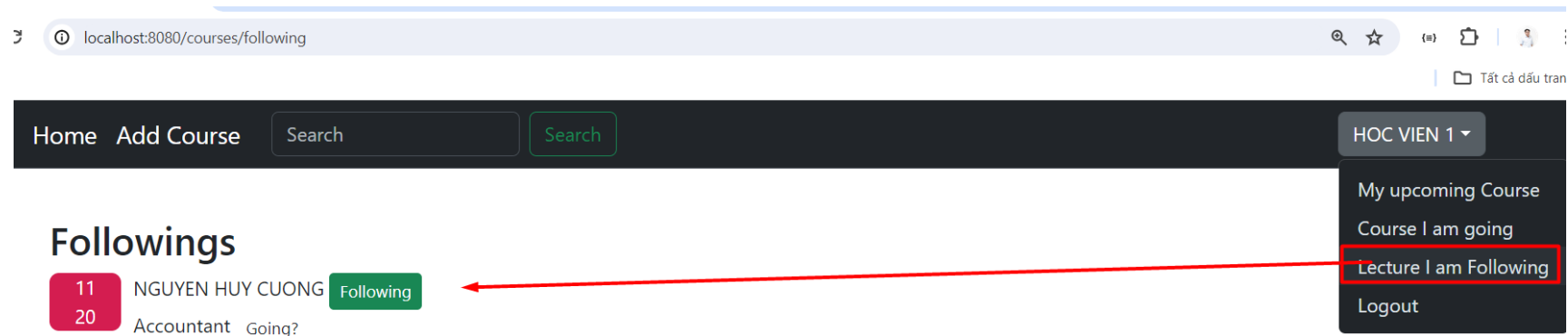
❑ **Course I am following:** lấy tất cả khóa học (của các giảng viên) mà **user** đăng nhập đang theo dõi (có dữ liệu ở bảng Following)

ví dụ 1: user1 (ZÆz•GÊ§ªiR%Ả^q) theo dõi giảng viên nguyenuyctuong (ĐQ'H O□□¶¶äô¼%ok)



#	follower_id	followee_id
1	ZÆz•GÊ§ªiR%Ả^q	ĐQ'H O□□¶¶äô¼%ok

❑ Khi đăng nhập bằng user1



localhost:8080/courses/following

Home Add Course Search Search

HOC VIEN 1 ▾

- My upcoming Course
- Course I am going
- Lecture I am Following**
- Logout

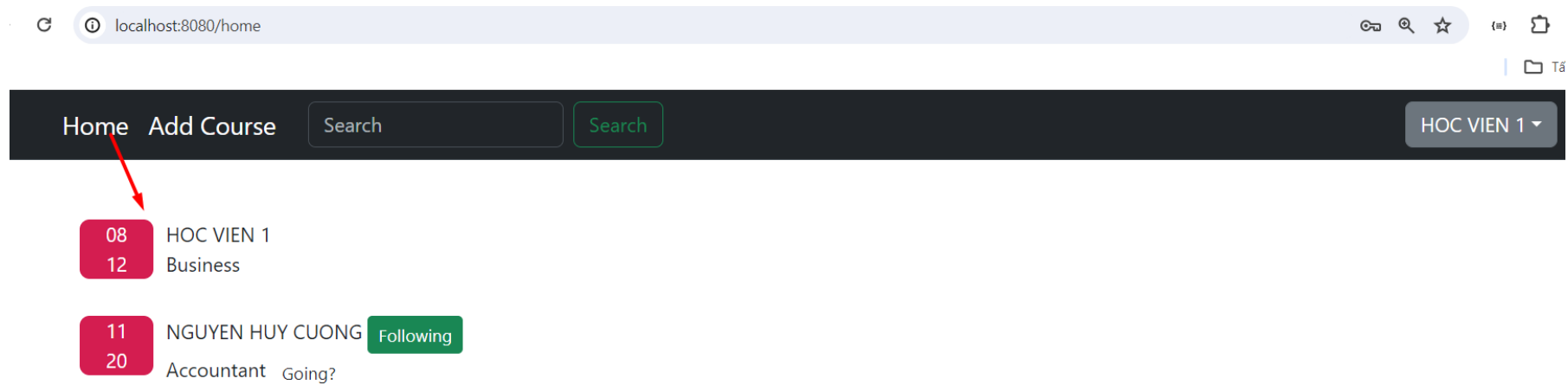
Followings

11 20	NGUYEN HUY CUONG Accountant Going?	Following
----------	---------------------------------------	-----------

ASM7: BigSchool – Theo dõi giảng viên

Yêu cầu 9: Theo dõi giảng viên

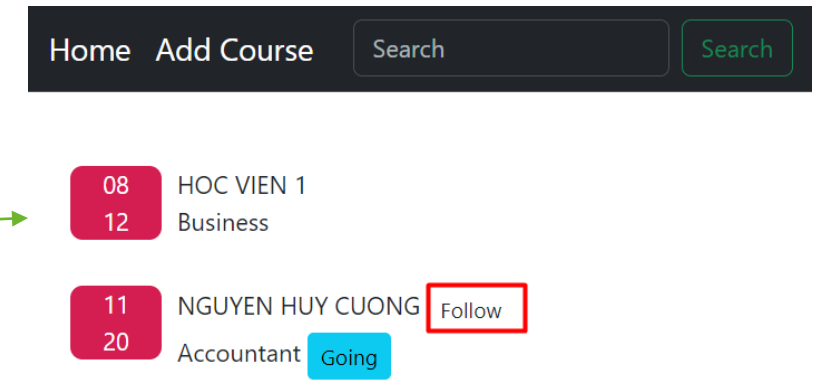
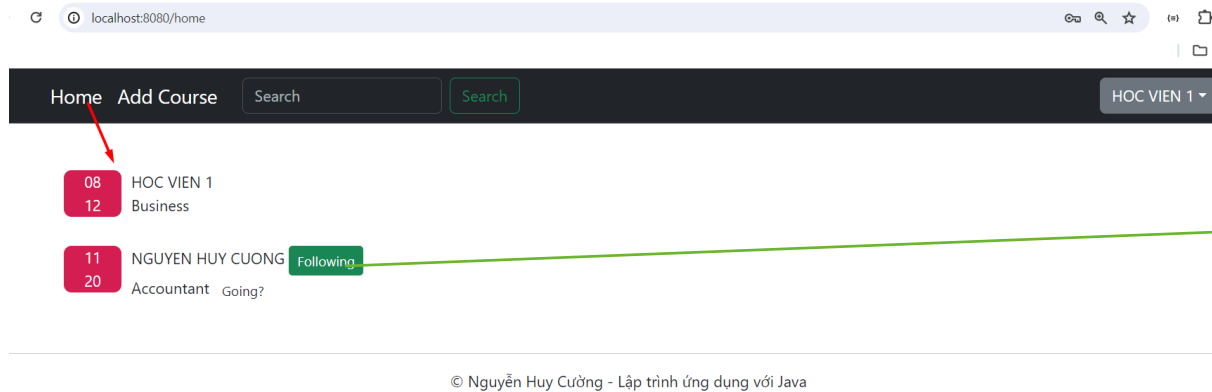
- ❑ **Home:** lấy tất cả khóa học sắp tới, với mỗi khóa học
 - Không hiện thị **going/ going? / follow / following** đối với những khóa học do người đăng nhập (giảng viên) tạo ra
 - Hiện thị **follow** với khóa học mà có giảng viên chưa theo dõi (so sánh ở bảng follow)
 - Hiện thị **following** của khóa học mà đã theo dõi giảng viên này



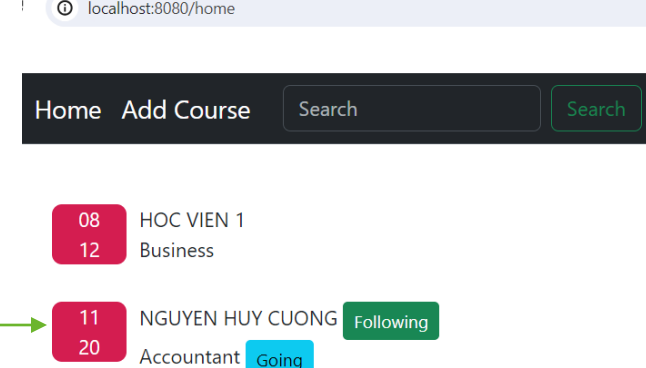
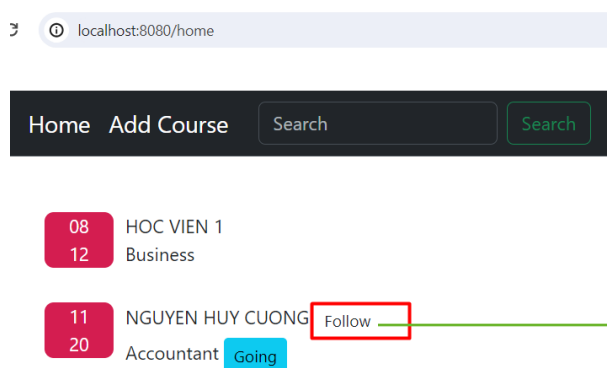
ASM7: BigSchool – Theo dõi giảng viên

Yêu cầu 10: Theo dõi giảng viên (update/ remove) với CSDL

- ❑ Khi click vào Following: sẽ hủy theo dõi giảng viên => xóa dữ liệu ở bảng **Following** và trở lại trạng thái Follow



- ❑ Khi click vào **Follow**: sẽ đăng kí theo dõi giảng viên => thêm dữ liệu ở bảng **Following**



Q&A

Thank you!