MỤC TIÊU

- Cài đặt ứng dụng web với Spring Boot Security
- Cấu hình bean UserDetailsService để quản lý nguồn dữ liệu người dùng trực tiếp với InMemoryUserDetailsManager
- Cấu hình bean PasswordEncoder mặc định với DelegatingPasswordEncoder
- Cấu hình bean SecurityFilterChain với 2 chế độ cho phép và yêu cầu đăng nhập
- Sử dụng authentication với form đăng nhập cấu hình mặc định và tùy biến.

BÀI 1: DỰ ÁN SPRING BOOT SECURITY (2 ĐIỂM)

Tạo dự án Spring Boot có tích hợp Security.

Hướng dẫn:

- Khai báo Dependency

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-security</artifactId>
</dependency>
<dependency>
    <groupId>org.thymeleaf.extras</groupId>
    <artifactId>thymeleaf-extras-springsecurity6</artifactId>
</dependency>
```

- Tạo file cấu hình Security và cấu hình cho 3 bean như sau:
  - @Bean PasswordEncoder: Sử dụng DefaultPasswordEncoder

```
@Bean
public PasswordEncoder passwordEncoder() {
    return PasswordEncoderFactories.createDelegatingPasswordEncoder();
}
```

  - @Bean UserDetailsService: Sử dụng InMemoryUserDetailsManager để quản lý nguồn dữ liệu User với 3 user như sau:
    - Username: user@gmail.com, Password: 123
    - Username: admin@gmail.com, Password: 123
    - Username: both@gmail.com, Password: 123

```
@Bean
public UserDetailsService userDetailsService(PasswordEncoder pe) {
    String password = pe.encode("123");
    UserDetails user1
        = User.withUsername("user@gmail.com").password(password).roles().build();
    UserDetails user2
```

```
                =
                User.withUsername("admin@gmail.com").password(password).roles().build();
        UserDetails user3
                = User.withUsername("both@gmail.com").password(password).roles().build();
        return new InMemoryUserDetailsManager(user1, user2, user3);
}
```

- @Bean SecurityFilterChain
  - Bỏ CSRF và CORS
  - Phân quyền sử dụng cho phép truy xuất tất cả các tài nguyên (permitAll())
  - Sử dụng form đăng nhập mặc định của hệ thống

```
@Bean
public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
        // Bỏ cấu hình mặc định CSRF và CORS
        http.csrf(config -> config.disable()).cors(config -> config.disable());
        // Phân quyền sử dụng
        http.authorizeHttpRequests(config -> {
                config.anyRequest().permitAll();
        });
        // Form đăng nhập mặc định
        http.formLogin(config -> config.permitAll());
        // Form đăng nhập
        http.formLogin(config -> {
                config.permitAll();
        });
        // Ghi nhớ tài khoản
        http.rememberMe(config -> {
                config.tokenValiditySeconds(3*24*60*60);
        });
        // Đăng xuất
        http.logout(Customizer.withDefaults());
        return http.build();
}
```

- Tạo Controller chứa các phương thức điều khiển được ánh xạ với các địa chỉ URL như sau
  - {"/", "/poly/url0"} => method0()
  - "/poly/url1" => method1()
  - "/poly/url2" => method2()
  - "/poly/url3" => method3()
  - "/poly/url4" => method4()

  Tất cả các phương thức điều khiển trên đều return "page" với biến ${message} có nội dung phù hợp

```
@Controller
```

```
public class MyController {
	@RequestMapping("/")
	public String home(Model model) {
		model.addAttribute("message", "@/ => home()");
		return "page";
	}
	@RequestMapping("/poly/url0")
	public String method0(Model model) {
		model.addAttribute("message", "@/poly/url0 => method1()");
		return "page";
	}
	@RequestMapping("/poly/url1")
	public String method1(Model model) {
		model.addAttribute("message", "@/poly/url1 => method1()");
		return "page";
	}
	@RequestMapping("/poly/url2")
	public String method2(Model model) {
		model.addAttribute("message", "@/poly/url2 => method2()");
		return "page";
	}
	@RequestMapping("/poly/url3")
	public String method3(Model model) {
		model.addAttribute("message", "@/poly/url3 => method3()");
		return "page";
	}
	@RequestMapping("/poly/url4")
	public String method4(Model model) {
		model.addAttribute("message", "@/poly/url4 => method4()");
		return "page";
	}
}
```

- Tạo page.html để hiển thị biến ${message} và các liên kết dẫn đến các url tương ứng của controller như sau:

```
<a href="/">Home</a> |
<a href="/poly/url0">URL0</a> |
<a href="/poly/url1">URL1</a> |
<a href="/poly/url2">URL2</a> |
<a href="/poly/url3">URL3</a> |
<a href="/poly/url4">URL4</a>
<hr>
<h1>Page: <b th:text="${message}"></b></h1>
```

- Chạy website và test thử các liên kết

BÀI 2: AUTHENTICATION (3 ĐIỂM)

- Sao chép lại Lab1.1 và hãy hiệu chỉnh SecurityFilterChain để phân quyền truy xuất theo yêu cầu sau đây
  - "/poly/**": Authenticated (yêu cầu phải đăng nhập)
  - Tất cả các địa chỉ URL còn lại được phép thuy xuất nặc danh
- Xây dựng Spring Bean AuthService hỗ trợ xử lý thông tin Authentication như sau:

```
@Service("auth")
public class AuthService {
	public Authentication getAuthentication() {
		return SecurityContextHolder.getContext().getAuthentication();
	}
	public String getUsername() {
		return this.getAuthentication().getName();
	}
	public List<String> getRoles() {
		return this.getAuthentication().getAuthorities().stream()
				.map(au -> au.getAuthority().substring(5)).toList();
	}
	public boolean isAuthenticated() {
		String username = this.getUsername();
		return (username != null && !username.equals("anonymousUser"));
	}
	public boolean hasAnyRole(String... rolesToCheck) {
		var grantedRoles = this.getRoles();
		return Stream.of(rolesToCheck).anyMatch(role -> grantedRoles.contains(role));
	}
}
```

- Sử dụng bean AuthService để lập trình bảo vệ và tùy biến giao diện như sau
  - Tạo view user-info.html và viết mã như sau

```
<div th:switch="${@auth.authenticated}">
    <ul th:case="true">
        <li>Username: <b>[[${@auth.username}]]</b></li>
        <li>Roles: <b th:text="${@auth.roles}"></b></li>
        <li><a href="/logout">Sign-out</a></li>
        <li th:if="${@auth.hasAnyRole('ADMIN')}">
            <a href="#">Goto Administration</a>
        </li>
    </ul>
    <ul th:case="false">
        <li>Bạn chưa đăng nhập</li>
    </ul>
</div>
```

- Chèn user-info.html vào cuối page.html

- Chạy và kiểm thử các liên kết đến các tài nguyên được bảo vệ

BÀI 3: TÙY BIẾN FORM ĐĂNG NHẬP (3 ĐIỂM)

Sao chép lại Lab1.2 và hiệu chỉnh SecurityFilterChain để cho phép sử dụng form đăng nhập do chính bạn thiết kế theo yêu cầu như sau:

- Tạo giao diện form đăng nhập

```
<i th:text="${message}"></i>
<form action="/login/check" method="post">
    <p>Username: <input name="username">
    <p>Password: <input name="password">
    <p><input name="remember-me" type="checkbox"> Remember me?
    <p><button>Login</button>
</form>
```

- Tạo LoginController để điều khiển các địa chỉ url như sau
  - "/login/form" => hiển form đăng nhập với ${message} "Vui lòng đăng nhập"
  - "/login/success" => hiển form đăng nhập với ${message} "Đăng nhập thành công"
  - "/login/failure" => hiển form đăng nhập với ${message} "Đăng nhập thất bại"
  - "/login/exit" => hiển form đăng nhập với ${message} "Đăng xuất thành công"

```
@Controller
public class LoginController {
	@RequestMapping("/login/{action}")
	public String login(Model model, @PathVariable("action") String action) {
		switch(action) {
			case "form" -> model.addAttribute("message", "Đăng nhập");
			case "success" -> model.addAttribute("message", "Đăng nhập thành công");
			case "failure" -> model.addAttribute("message", "Sai thông tin đăng nhập");
			case "exit" -> model.addAttribute("message", "Đăng xuất thành công");
		}
		return "login";
	}
}
```

- Cấu hình xử lý đăng nhập và đăng xuất tùy biến

```
// Form đăng nhập
http.formLogin(config -> {
	config.loginPage("/login/form");
	config.loginProcessingUrl("/login/check");
	config.defaultSuccessUrl("/login/success");
	config.failureUrl("/login/failure");
	config.permitAll();
	config.usernameParameter("username");
	config.passwordParameter("password");
});
```

```
// Ghi nhớ tài khoản
http.rememberMe(config -> {
	config.tokenValiditySeconds(3*24*60*60);
	config.rememberMeCookieName("remember-me");
	config.rememberMeParameter("remember-me");
});
```

```
// Đăng xuất
http.logout(config -> {
	config.logoutUrl("/logout");
	config.logoutSuccessUrl("/login/exit");
	config.clearAuthentication(true);
	config.invalidateHttpSession(true);
	config.deleteCookies("remember-me");
});
```

BÀI 4: GIẢNG VIÊN CHO THÊM (2 ĐIỂM)