<https://youtu.be/IAdnZWtb5QU?t=630>

Chức năng đăng nhập gồm

|  |
| --- |
| SecurityConfig |
| CustomUserDetailsService |
| UserRepository |
| User |

Class SecurityConfig có biến là CustomUserDetailsService

Class CustomUserDetailsService có biến là UserRepository

UserRepository có biến là User

CustomUserDetailsService lấy các hàm liên quan chức năng đăng nhập

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Trang web

Nội dung do AI tạo ra có thể không chính xác.

User implements UserDetails để lấy các hàm liên quan chức năng đăng nhập

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Biểu tượng máy tính

Nội dung do AI tạo ra có thể không chính xác.

UserRepo implement CrudRepository để lấy các hàm thao tác CSDL

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Trang web

Nội dung do AI tạo ra có thể không chính xác.

# SecurityConfig

|  |
| --- |
| @Configuration  @EnableWebSecurity  public class SecurityConfig {      private final UserDetailsService userDetailsService;      public SecurityConfig(UserDetailsService userDetailsService) {          this.userDetailsService = userDetailsService;      }      @Bean      public PasswordEncoder passwordEncoder(){          return new BCryptPasswordEncoder();      }      @Bean      public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {          http              .authorizeRequests(                  request1 -> request1.anyRequest().authenticated()              )              .formLogin(Customizer.withDefaults())              .logout(Customizer.withDefaults())              .userDetailsService(this.userDetailsService);            return http.build();      }  } |

CustomUserDetailsService

|  |
| --- |
| @Service  public class CustomUserDetailsService implements UserDetailsService {      private final UserRepository userRepository;      public CustomUserDetailsService(UserRepository userRepository) {          this.userRepository = userRepository;      }      @Override      public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {            Optional<User> optUser = this.userRepository.findByUsername(username);            if(optUser.isPresent()==true){              return optUser.get();          }          else{              return null;          }      }  } |

# User

|  |
| --- |
| @Entity  @Table  @Data  public class User implements UserDetails{        @Id      @GeneratedValue(strategy = GenerationType.IDENTITY)      @Column(name = "id")      private Long id;      @Column( name = "address", nullable = true)      private String address;      @Column(name = "sodienthoai", nullable = true)      private String sodienthoai;      @Column(name= "email", nullable= false)      private String email;      @Column(name= "username", nullable= false, unique = true)      private String username;      @Column(name= "password", nullable= false)      private String password;      @Column(name= "mota", nullable = true)      private String mota;      @Column(name = "enabled")      private Boolean enabled;      @Override      public Collection<? extends GrantedAuthority> getAuthorities() {          SimpleGrantedAuthority authority = new SimpleGrantedAuthority("ROLE\_USER");          return List.of(authority);      }      @Override      public String getPassword() {          return this.password;      }      @Override      public String getUsername() {          return this.username;      }      @Override      public boolean isAccountNonExpired() {          return true;      }      @Override      public boolean isAccountNonLocked() {          return true;      }      @Override      public boolean isCredentialsNonExpired() {          // TODO Auto-generated method stub          return true;      }      @Override      public boolean isEnabled() {          return this.enabled;      }      public User() {      }      public User          (              String address,              String sodienthoai,              String email,              String username,              String password,              String mota,              Boolean enabled          )          {              this.address = address;              this.sodienthoai = sodienthoai;              this.email = email;              this.username = username;              this.password = password;              this.mota = mota;              this.enabled = enabled;          }      } |

# UserRepository

|  |
| --- |
| @Repository  public interface UserRepository      extends CrudRepository<User, Long>  {      Optional<User> findByUsername(String username);  } |

# // chưa có chức năng đăng ký để tạo tài khoản