Par Hajar TAJAYOUTI
GLSID2

Activité Pratique Spring MVC, Spring Data JPA, Spring Security

Cette application Web basée sur Spring MVC, Spring Data JPA et Spring Security permet de gérer des étudiants.

Les entités utilisées dans l'application sont :

Student

```
@Entity
@Data @AllArgsConstructor @NoArgsConstructor
public class Student {
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @NotEmpty
    @Size(min = 3, max = 50)
    private String nom;
    @Size(min = 3, max = 50)
    private String prenom;
    private String email;
    @Temporal(TemporalType.DATE)
    @DateTimeFormat(pattern = "yyyy-MM-dd")
    private Date dateNaissance;
    @Enumerated(EnumType.STRING)
    private Gender genre;
    private Boolean enRegle;
}
```

Les fonctionnalités de l'application

L'application offre les fonctionnalités suivantes :

1. Chercher des étudiants par nom.
2. La suppression des étudiants en utilisant la méthode (DELETE au lieu de GET).
3. la pagination.
4. Ajouter des étudiants avec validation des formulaires.
5. la mis à jour des étudiants.

La sécurité avec Spring Security

L'accès à l'application est sécurisée avec un système d'authentification basé sur **Spring security** .

L'implémentation de UserDetailsService 'UserDetailsServiceImpl'

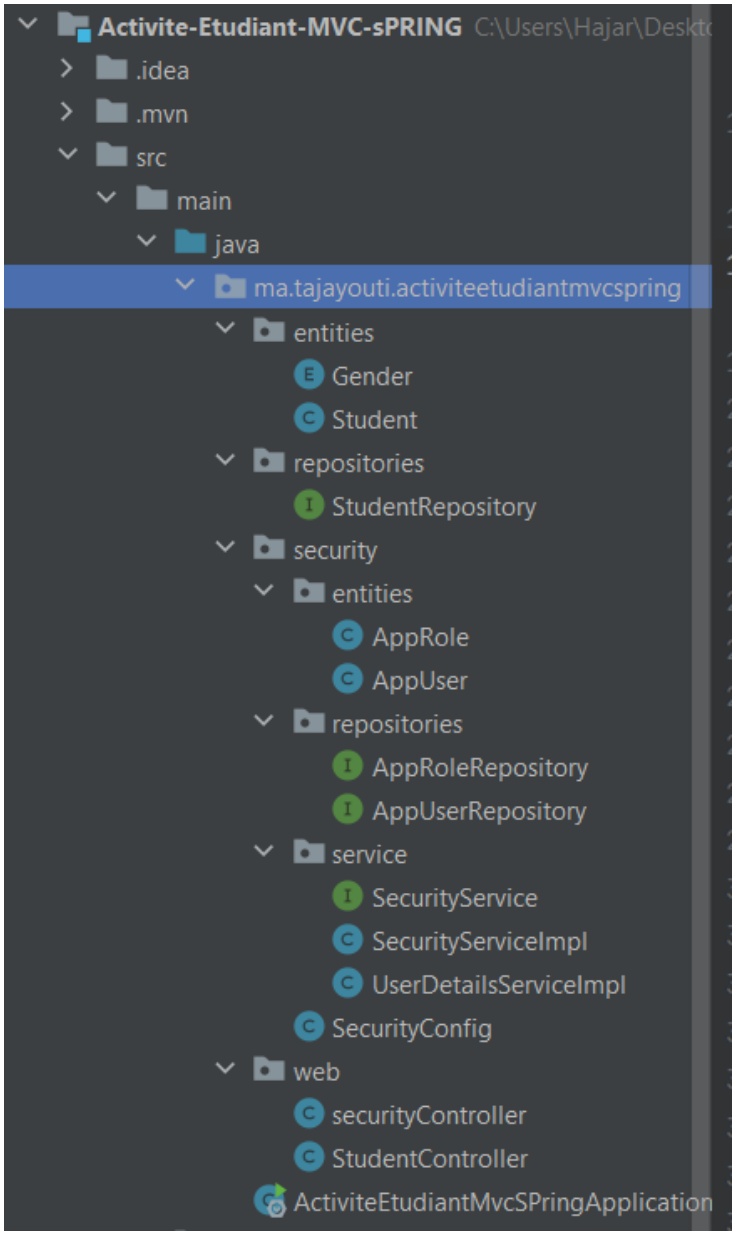
```
@Service
public class UserDetailsServiceImpl implements UserDetailsService {
    @Autowired
    private SecurityService securityService;


    @Override
    public UserDetails loadUserByUsername(String username) throws
UsernameNotFoundException {
        AppUser appUser = securityService.loadUserByUsername(username);

        Collection<GrantedAuthority> authorities1 = appUser
                .getAppRoles()
                .stream()
                .map(role-> new SimpleGrantedAuthority(role.getRoleName()))
                .collect(Collectors.toList());

        User user = new
User(appUser.getUsername(),appUser.getPassword(),authorities1);
        return user;
    }
}
```

La classe de configuration de la sécurité

```
//tte classe avec cette annotation va etre instancier au 1er lieu
@Configuration
@EnableWebSecurity
public class SecurityConfig extends WebSecurityConfigurerAdapter {
    @Autowired
    private DataSource dataSource;
    @Autowired
    private PasswordEncoder passwordEncoder;
    @Autowired
    private UserDetailsServiceImpl userDetailsService;
    @Override
    protected void configure(AuthenticationManagerBuilder auth) throws
Exception {

        auth.userDetailsService(userDetailsService); //chercher les users
    }

    @Override
    protected void configure(HttpSecurity http) throws Exception {

        http.formLogin(); //formul de login par defaut ... si je veux
```

```
personnalisé ce form http.formLogin().loginPage("/login")
  // droits d accees
        http.authorizeRequests().antMatchers("/").permitAll();

http.authorizeRequests().antMatchers("/admin/**").hasAnyAuthority("ADMIN");

http.authorizeRequests().antMatchers("/user/**").hasAnyAuthority("USER");
        http.authorizeRequests().antMatchers("/webjars/**").permitAll();

        http.authorizeRequests().anyRequest().authenticated();

        http.exceptionHandling().accessDeniedPage("/403");
    }


}
```

Structure du projet



Comment ça marche ?

Link : https://www.youtube.com/watch?v=7H4zpaVyGwI