Spring Security

Spring Security

- Mechanism for providing security to Java application
- Less Configuration - More Portability
- Features beyond the JEE Standard

Authentication and Authorization

- Authentication ensuring that the user is who it claims to be
- Authorization - ensuring that the user has enough privileges to perform an action
- Customizable Authentication and Authorization

Extended Security Capabilities

- Protecting Web Resources (specific roles for URLs)
- Authorizing Method Invocations
- Restricting Entity Access

Additional Features

- CSRF Protection
- XSS Protection
- Password encoding

Spring Security Benefits

- Spring Model to Security
- Annotation Based Security
- Integration with Spring MVC
- Testing Support
- Global and Layered Security Approach

Security Principles

- Authentication
- Authorization
- Principal

Configuration - Step 1

- Register Spring Security Filter chain before any other filters

```
public class SecurityWebApplicationInitializer extends AbstractSecurityWebApplicationInitializer {

}
```

Configuration

- Java Config

```
@Configuration
@EnableWebSecurity
public class SecurityConfig extends WebSecurityConfigurerAdapter {
}
```

- Authentication

```
@Autowired
public void configureGlobal(AuthenticationManagerBuilder auth) {
       auth.inMemoryAuthentication().withUser("guest").password("guest").roles("USER")
               .and()
               .withUser("admin").password("admin").roles("ADMIN");
}
```

Authorization Configuration

```
@Override
protected void configure(HttpSecurity http) throws Exception {
   http
           .authorizeRequests()
           .antMatchers("/players/delete/**").hasRole("ADMIN")
           .antMatchers("/**").hasAnyRole("ADMIN","USER")
           .anyRequest()
           .authenticated()

}
```

Securing Front End

```
.antMatchers("/**").hasAnyRole("ADMIN","USER")

<sec:authorize url='/players/show/*'>
   <td>
       <spring:url var="showUrl" value="show/{id}">
           <spring:param name="id" value="${user.id}"/>
       </spring:url>
       <a href="${showUrl}">${user.id}</a>
   </td>
</sec:authorize>
```

```
 .antMatchers("/players/delete/**").hasRole("ADMIN")

<sec:authorize url='/players/delete/*'>
   <td>
       <spring:url var="deleteUrl" value="delete/{id}">
           <spring:param name="id" value="${user.id}"/>
       </spring:url>
       <a href="${deleteUrl}"><spring:message code="label.delete"/></a>
   </td>
</sec:authorize>
```