8.2C: Authorization and spring security

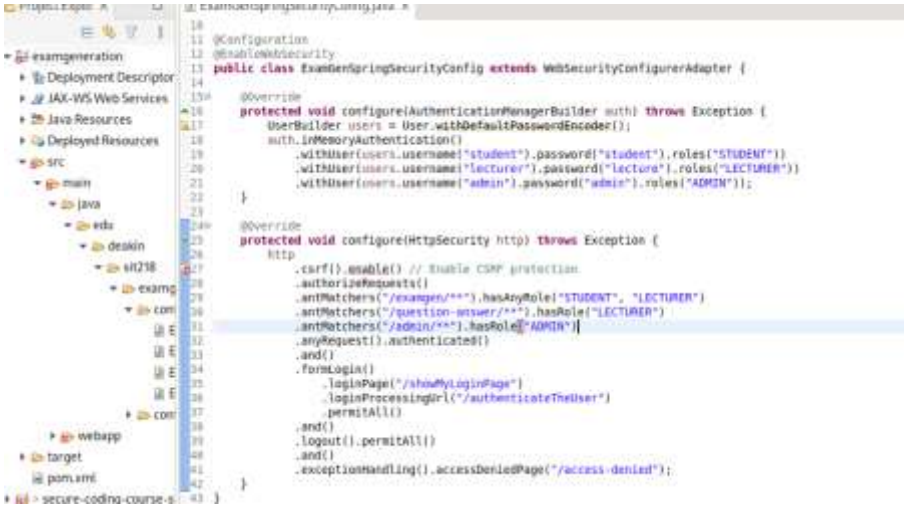
- As with any website creation, web applications have to be safeguarded against Cross-Site Request Forgery (CSRF). Spring security, one of the popular security framework of Java has a provision to prevent cross site request forgery. In other words, due to the necessity to protect the exam generation application against the threats resulting from the implementation of the CSRF attack, as well as to describe the given attack and its protection, Spring Security's CSRF protection mechanism is the aim of the task.

The CSRF Attack Defense Mechanism of Spring Security

- What concerns the configurations of Spring Security and Http Security, the CSRF protection is enabled by default. The CSRF is created by Spring Security in form of another cookie when users allow it. Next, this token is appended to the form in the form of a hidden field or appended to the headers of the AJAX request. Spring: After form submission, the token is resubmitted, if it has been changed, not present, or contains wrong token, then request is denied. This technique prevents unauthorized web sites from employing the user's session ID in a manner that can hurt him/her as the number of activities commanded by tokens can only be so many – what is allowed for authorized users.

Integration within the exam generation software

- To ensure that indeed CSRF is enabled, we checked that the CSRF property was active in the ExamGen Spring Security Config class of the exam generation application. The relevant section of the setup is shown below:



The csrf() at http. The CSRF protection is enabled in this code by using the enable() method. To ensure that all the web application endpoints that that require user login such as /examgen/** and /admin/**, this will ensure that every form in the application has CSRF Tokens.

Mechanism Works

- Finally, the CSRF token is included along with the form data after the form has been filled and submitted (for instance for data transfer or login). This token plays very important role in the cross-site request forgery check since it ensures that the request is from the person who is logged in and using the website and not from a third party website that wants to fool the browser into submitting the form.
- After the completion of the form, the server checks the token that was also included in the form submission. It maintains this token and the token stored in the server and used throughout the session of a user. The action is processed by the server so it is provided that both tokens are similar and the tokens are comprehensible in this way. If, and only if the tokens are out-aligned, or not present at all, it halts the request in an attempt to halt further work. The user is a lot more protected from attempts that try to force the browser to perform tasks the user is not comfortable with by doing this.
- Spring Security most times creates a special token which is called a CSRF token at the time a user logs into the exam generation app. Each person thus or has a token linked to the given session in particular. It then comes back in whatever format the user enters on the website in a hidden field thereafter. This ensures that the token is sent together with any information the user sends through the from.

Thus, it would not be wrong to conclude that through the use of CSRF protection it becomes possible to safeguard the new exam-generation app against CSRF attacks. This application now checks all the requests that regard state, thus ensuring Spring Security's intrinsic CSRF ability, only allows certain users with tokens to perform critical functions. This layer additionally enhances security of the web application in a sense that undesired parties cannot use user sessions for their advantage.