HW#5 – İbrahim Çelik

1 – What are Authentication and Authorization ?

Authentication is the process of identifying users and validating who they claim to be. It is the first step in accessing the network. It is a phase that approves the login to the network by verifying the user's credentials and password by comparing them over the database.

Authorization happens after a user’s identity has been successfully authenticated. After accessing the system, it is called authorization, which determines what operations can be performed so that that user can access it.

2 - What is Hashing in Spring Security ?

By definition, hashing is in the process of successfully concluding with the use of a hash of obvious fleeting entries. Below you can see the images of the introduction in three different text sections.

- hashing("merhaba") = C39436EE452E641CDE2EB992AB397911
- hashing("marhaba") = ECF853B448D836DB76DEBF2C342E0369
- hashing("medium") = 075A3E36A0A52DCBC568C05788E8A713

Hash algorithms are deterministic. That is, as long as the input does not change, the same hash is always output. For example, even if you repeatedly hash the "hello" value above, the result will always be the value you see on the right. Among the two most used algorithms for hashing, the SHA-256 hashing algorithm can produce 256-bit outputs, while MD5 produces 128-bit outputs. Hash algorithms work one way. So even if we know the value of a hash, we cannot go back to its unhashed form. In addition, as seen in the outputs, the hashed output changes completely even with the slightest letter change.

3 - What is Salting and why do we use the process of Salting ?

According to OWASP Guidelines, a salt is a value generated by a cryptographically secure function that is added to the input of hash functions to create unique hashes for every input, regardless of the input not being unique. A salt makes a hash function look non-deterministic, which is good as we don't want to reveal duplicate passwords through our hashing.

Anyway, when salting, the additional value is referred to as a “salt.”

4 - What is “intercept-url” pattern ?

Most web applications using Spring Security only have a couple of intercept-urls because they only have very basic security requirements. We need to have unauthenticated access to the login and login-error screens and usually some aspect of the public site, so that can be a few URL patterns. Then there's often an admin section, and then everything else is ROLE_USER.

If need more roles, it's customary to associate them with top level URL path components. Although it's not required, it makes it easier to be sure that resources are appropriately protected.

5 - What do you mean by session management in Spring Security ?

Session management is the process of securely handling multiple requests to a web-based application or service from a single user or entity. HTTP is used to communicate between websites and browsers, and a session is a series of HTTP requests and transactions created by the same user. The session management implementation specifies the process for sharing and continually exchanging the session ID between the user and the web application.

As HTTP protocol is stateless, and to keep track of customer behavior, we need session management. Session Management is a web container framework used to store session data for a specific user.

6 – Why we need Exception Handling ?

The process of writing an error-free code is almost impossible, we may encounter errors inevitably, and by catching these errors with the use of java exception handling, we can learn from where and why our problem originates.

7 - Explain what is AuthenticationManager in Spring security ?

AuthenticationManager is a container for authentication providers and offers a consistent interface to all of them. In most cases, the default AuthenticationManager is more than enough.

Generally speaking, the AuthenticationManager passes some type of AuthenticationProviders to each of the AuthenticationTokens, and each of them checks and if they can use it for authentication, it returns answers like "Authenticated", "Not authenticated" or "Failed to authenticate".

This is the mechanism that allows you to bind other authentication schemes such as authenticating to an LDAP or Active Directory server or OpenID, and is one of the main extension points within the Spring Security framework.

8 - What is Spring Security Filter Chain ?

Spring Security uses the filter chain to implement most of the security functions. We drive Spring Security via the servlet filters in a web application. Servlet filters are used to block the request until it enters the physical resource (e.g. the Spring Controller).

- The client sends a request for a resource (MVC controller). Application container Create Filter Chain to handle incoming requests.
- Each HttpServletRequest passes through a filter chain based on the URI request route. (We can customize whether we run a filter chain for some request or URI related request).
- Filters use the following rationale for much of the web framework.
    - Update the HttpServletRequest or HttpServletResponse until you reach our Spring MVC controller.
    - Can stop processing the request and send a response to the client. (e.g. Servlet that does not make requests to unique URIs).

For a web application that uses Spring Security, all incoming HttpServletRequest passes through the spring security filter chain until it hits the Spring MVC controller.

9 – What are the differences between OAuth2 and JWT ?

The main differences between JWT & OAuth

- OAuth 2.0 defines a protocol & JWT defines a token format.
- OAuth can use either JWT as a token format or access token which is a bearer token.
- OpenID connect mostly use JWT as a token format.

10 - What is method security and why do we need it ?

In simple terms, Spring method security allows us to support / add authorization supports at the method level. On a high level, we can configure which roles are allowed to access what method within the same service class. Let’s take an example of CustomerService class.

- A customer service can only use the view method.
- We only allow the user with Admin permission to call the delete method in the same service class.

In this article, we will look at the steps and configuration to enable spring method level security using the different annotations. Spring security supports both JSR-250 based annotation and Spring security based annotation, which allows us to use the new and powerful Spring expression language.

11 – What Proxy means and how and where can be used ?

Proxy is one of the structural patterns in Gang of Four (GoF), that provides a substitute or placeholder for another object to control access. The proxy pattern allows you to perform some logic either before or after invoking the original object.

On adding @Cacheable annotation, spring creates a new caching proxy class. This class will be in charge of adding Caching behavior and will be used for dependency injection.

12 – Waht is Wrapper Class and where can be used ?

There are two types of variable types in the Java programming language,

- Primitive Types
- Reference Types

Primitive Types : We all know primitive types, numeric or boolean values are the most basic types we keep (such as byte, int, short, long, char, float and double). Primitive types are used in the stack of memory. When passed to a method as a variable, a copy is made and copied. The transaction is made on, so the current value remains unchanged.

Reference Types: Reference types are classes and arrays. Contrary to value types, reference types are not processed over their copies when we pass to methods, they are processed on existing objects.

Wrapper Classes : While writing programs in Java, we sometimes need Primitive types to be wrapped by another class as above. This is inevitable, especially if we are working with collections classes.

This is because collections objects can only contain objects. If we want to keep a Primitive type inside a collections object, we will have to wrap that Primitive type with another class. Therefore, in Java, there is a wrapper class for each primitive type under the java.lang package.

There are some functions that we use a lot in Wrappers.

- The valueOf() method turns a String into a Wrapper.
- For xxxValue() methods;
- The shortValue() method turns a Wrapper into a primitive.
- The byteValue() method returns a Wrapper byte.
- for parseXXX;
- The parseInt() method returns the Wrapper primitive.

13 – What is SSL ? What is TLS ? What is the difference ? How can we use them ?

SSL stands for Secure Sockets Layer and, in short, it's the standard technology for keeping an internet connection secure and safeguarding any sensitive data that is being sent between two systems, preventing criminals from reading and modifying any information transferred, including potential personal details. The two systems can be a server and a client (for example, a shopping website and browser) or server to server (for example, an application with personal identifiable information or with payroll information).

TLS (Transport Layer Security) is just an updated, more secure, version of SSL. We still refer to our security certificates as SSL because it is a more commonly used term

SSL	**TLS**
SSL stands for “Secure Socket Layer.”	TLS stands for “Transport Layer Security.”
Netscape developed the first version of SSL in 1995.	The first version of TLS was developed by the Internet Engineering Taskforce (IETF) in 1999.
SSL is a cryptographic protocol that uses explicit connections to establish secure communication between web server and client.	TLS is also a cryptographic protocol that provides secure communication between web server and client via implicit connections. It’s the successor of SSL protocol.

Three versions of SSL have been released: SSL 1.0, 2.0, and 3.0.	Four versions of TLS have been released: TLS 1.0, 1.1, 1.2, and 1.3.
All versions of SSL have been found vulnerable, and they all have been deprecated.	TLS 1.0 and 1.1 have been “broken” and are deprecated as of March 2020. TLS 1.2 is the most widely deployed protocol version.

These are the essential principles to grasp for understanding how SSL/TLS works:

- Secure communication begins with a TLS handshake, in which the two communicating parties open a secure connection and exchange the public key
- During the TLS handshake, the two parties generate session keys, and the session keys encrypt and decrypt all communications after the TLS handshake
- Different session keys are used to encrypt communications in each new session
- TLS ensures that the party on the server side, or the website the user is interacting with, is actually who they claim to be
- TLS also ensures that data has not been altered, since a message authentication code (MAC) is included with transmissions.

14 - Why do you need the intercept-url ?

(I found this information on stackoverflow.)

Roles can be defined by you arbitrarily and permission access set for each role as you like.

The intercept URLs need to be listed from most to least specific, because if you put the least specific one first, like this:

pattern="/foo/bar/**" pattern="/foo/bar/baz*"

When someone navigates to /foo/bar/baz, the permission settings from /foo/bar will get applied, because it is matched first in the list of intercept URLs. This requires more effort on the part of the developer, but it is faster than matching the exact string over every URL in the list.