Introduction:

[[TutsNode.com] - Spring Security Core Beginner to Guru\02 - Introduction to Spring Security\26127142-IntrotoSpringSecurity.pdf](%5bTutsNode.com%5d%20-%20Spring%20Security%20Core%20Beginner%20to%20Guru/02%20-%20Introduction%20to%20Spring%20Security/26127142-IntrotoSpringSecurity.pdf)

OWASP : **Open Web Application Security Project** (OWASP) is a nonprofit foundation dedicated to improving software security.

PDF Link: [[TutsNode.com] - Spring Security Core Beginner to Guru\02 - Introduction to Spring Security\25604716-CommonWebVulner.pdf](%5bTutsNode.com%5d%20-%20Spring%20Security%20Core%20Beginner%20to%20Guru/02%20-%20Introduction%20to%20Spring%20Security/25604716-CommonWebVulner.pdf) \*\*

Will be used in FE.

Cross Site Scripting (XSS) \*\*:

[[TutsNode.com] - Spring Security Core Beginner to Guru\02 - Introduction to Spring Security\25605506-CrossSiteScripting.pdf](%5bTutsNode.com%5d%20-%20Spring%20Security%20Core%20Beginner%20to%20Guru/02%20-%20Introduction%20to%20Spring%20Security/25605506-CrossSiteScripting.pdf)

Injectiong js scripts in input fields.

CSRF \*\*:

[[TutsNode.com] - Spring Security Core Beginner to Guru\02 - Introduction to Spring Security\25605760-CrossSiteForgery.pdf](%5bTutsNode.com%5d%20-%20Spring%20Security%20Core%20Beginner%20to%20Guru/02%20-%20Introduction%20to%20Spring%20Security/25605760-CrossSiteForgery.pdf)

Sending a redirect link which has cookies in it.

Csrf is a token that will be passed in the header or url to the server(from the Original Page). Bcz hacker is just redirecting, browser will not have the csrf token in url or header.

3.Http basic auth

User credentials are either be sending in **url** or **Header**

**url**: <https://username:password@www.xyz.com>

**Header**: Authorization: Basic <Base64 Encoded String> (encoded string = username:password (for testing, goto base64 encoder, then username:password and pass it to Authorization))

**Note:** Spring Security uses web mvc in the background.

3.6 testing spring security with Junit5

3.8 Spring security filter chain

[[TutsNode.com] - Spring Security Core Beginner to Guru\03 - HTTP Basic Auth\25683390-SpringSecurityFilterChain.pdf](%5bTutsNode.com%5d%20-%20Spring%20Security%20Core%20Beginner%20to%20Guru/03%20-%20HTTP%20Basic%20Auth/25683390-SpringSecurityFilterChain.pdf)

Spring Security filter chains will come before and after Dispatcher Servlet (front controller).

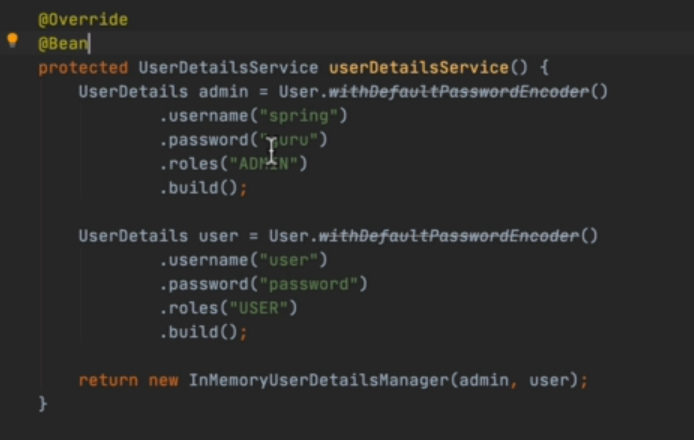
4.in memory authentication provider

4.1 Spring security Authentication flow:

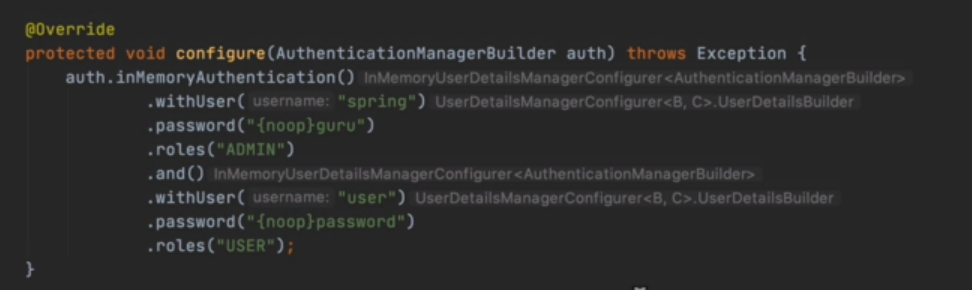
[[TutsNode.com] - Spring Security Core Beginner to Guru\05 - In Memory Authentication Provider\25732492-SpringSecAuthProcess.pdf](%5bTutsNode.com%5d%20-%20Spring%20Security%20Core%20Beginner%20to%20Guru/05%20-%20In%20Memory%20Authentication%20Provider/25732492-SpringSecAuthProcess.pdf)

We were doing configuration of basic inMemory http auth by, spring.security.user.name=”xx” spring.security.user.password=”xx”

We can do this manually



Using Fluent Api technique:



Delegating Password Encoder

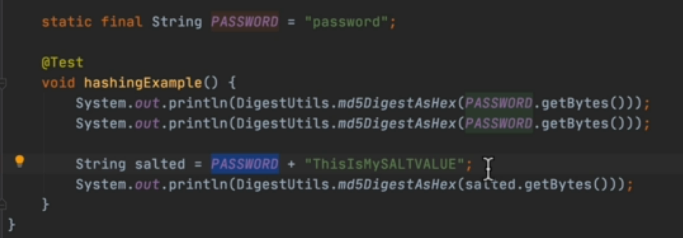
Allows storage of password hashes in multiple formats: {encodername}<somepasswordHashValue>.

Encodername= noop, bcrypt, …etc

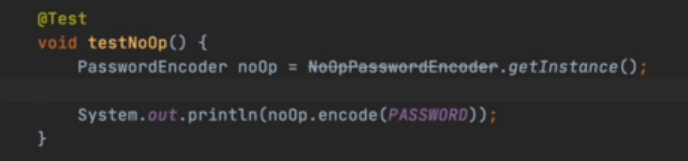
[[TutsNode.com] - Spring Security Core Beginner to Guru\06 - Password Security\25734930-PasswordEncoding.pdf](%5bTutsNode.com%5d%20-%20Spring%20Security%20Core%20Beginner%20to%20Guru/06%20-%20Password%20Security/25734930-PasswordEncoding.pdf) \*\*

4.3 Hashing Algorithms

MD5 hash & password salt: (not recommended)

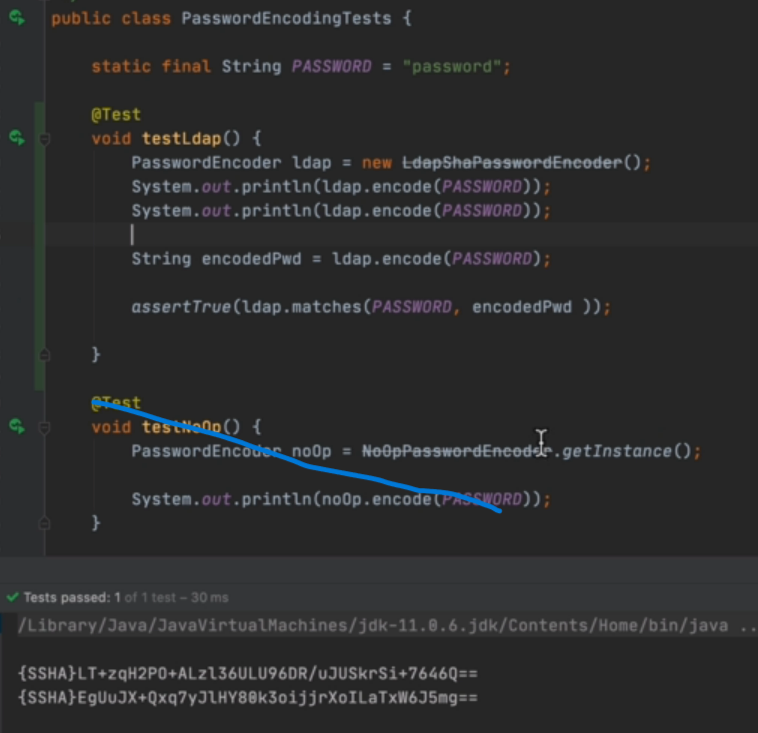


NoOp password Encoder: (not recommended)

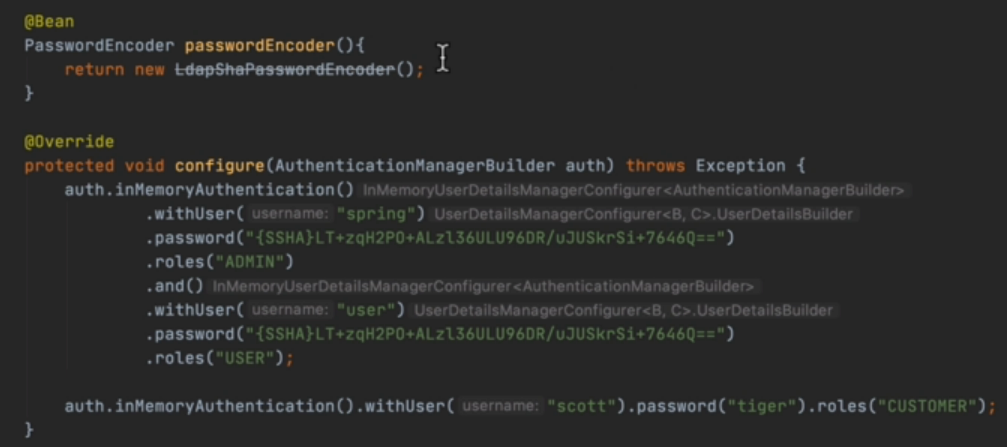


Ldap password Encoder: (not recommended)

-uses random salt



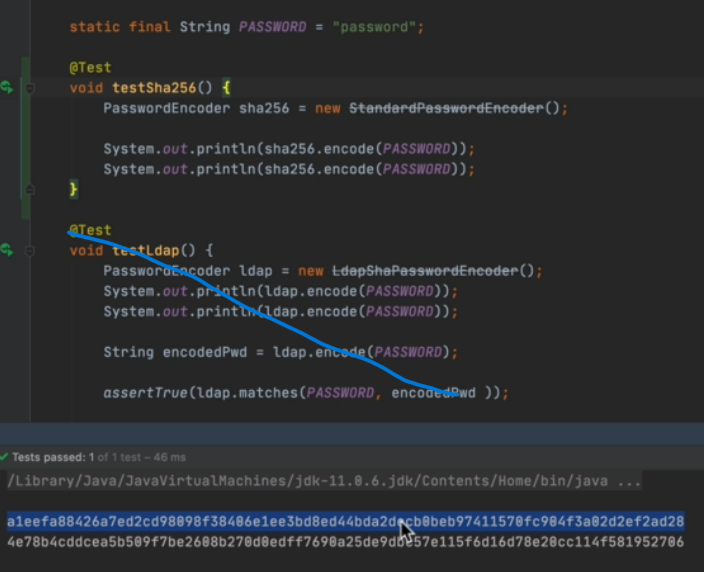
Declaring as Bean,



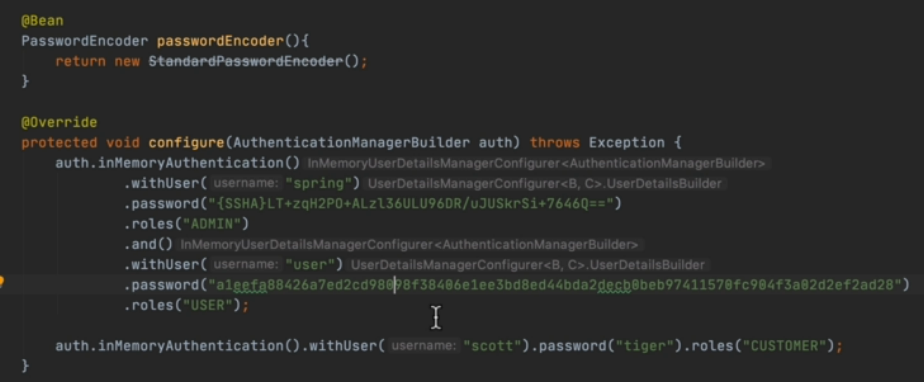
SHA-256 Password Encoder(not recommended)

-It was default in previous versions of spring security

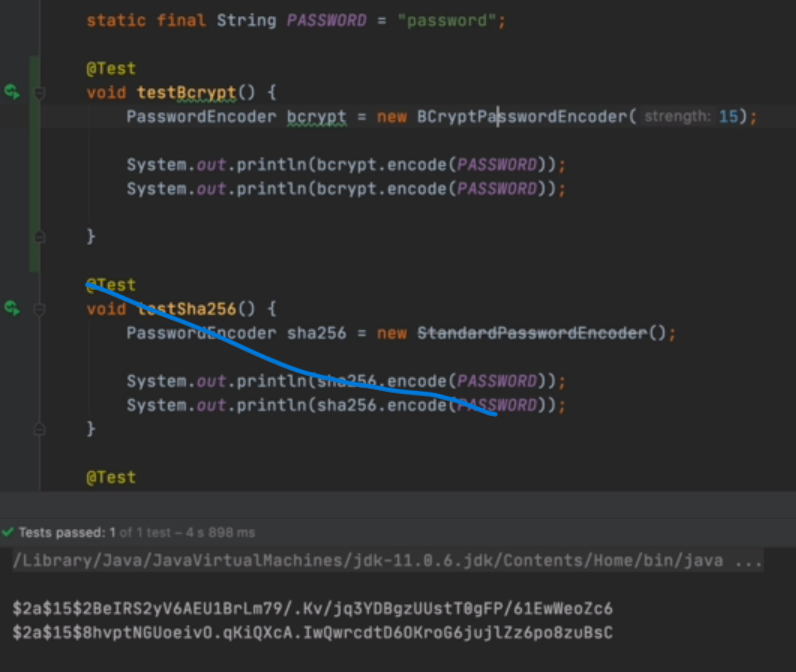
- Cons: tooFast in case of brut force attack. Brut Force attack requires more computational power. So other algorithms are slow compare to this.



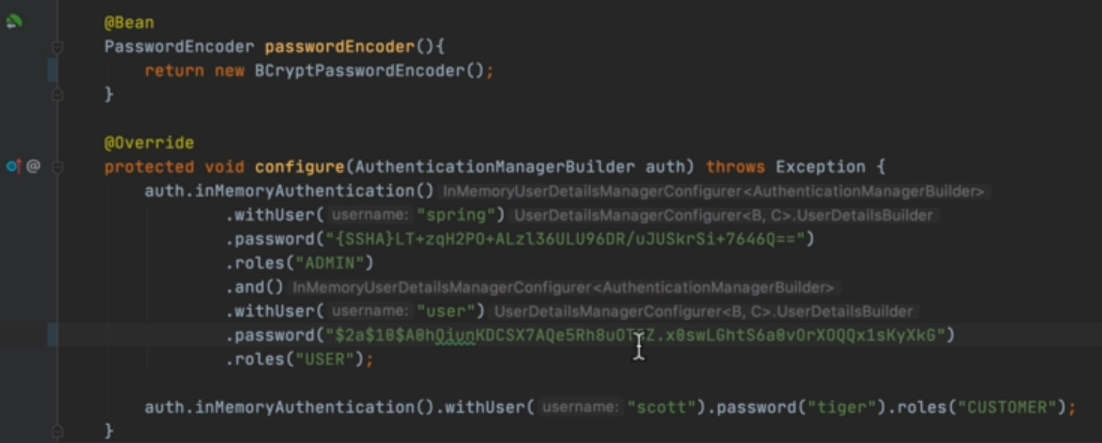
Declaring as Bean,



Bcrypt Password Encoder



Observation: It is decrypting the password that we passed in the config ( we are passing normal password in test cases, which is working)

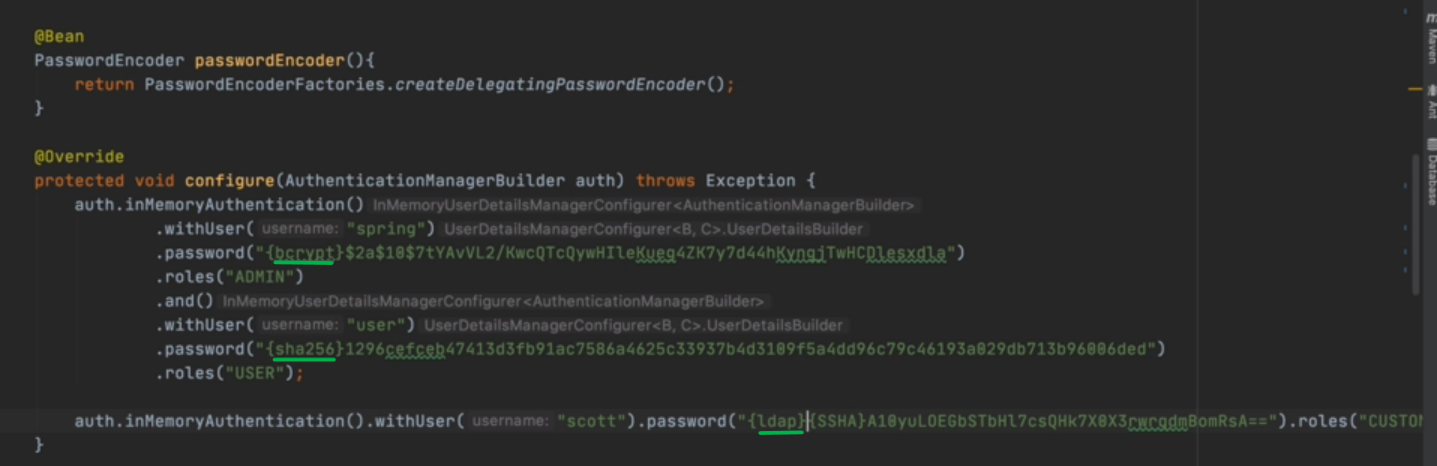


In testCase,



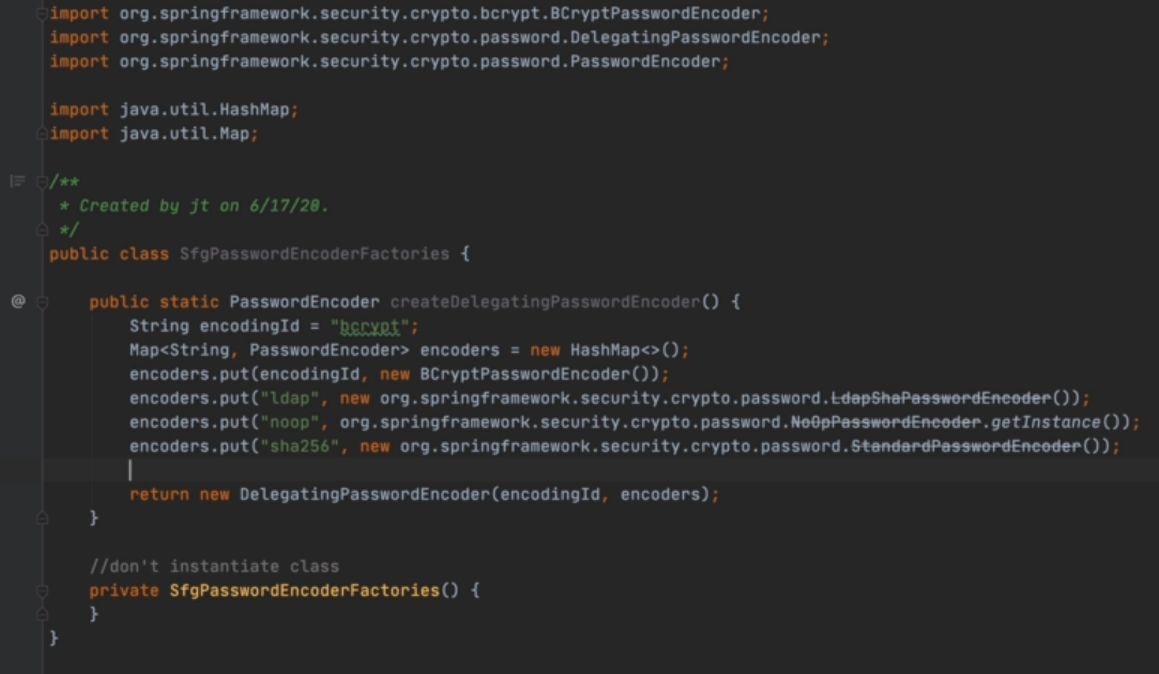
Delegating Password Encoder

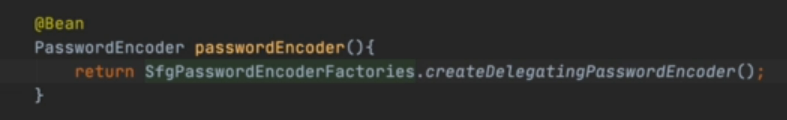
We can use multiple encoding algorithms at a time.



Custom Delegating Password Encoder

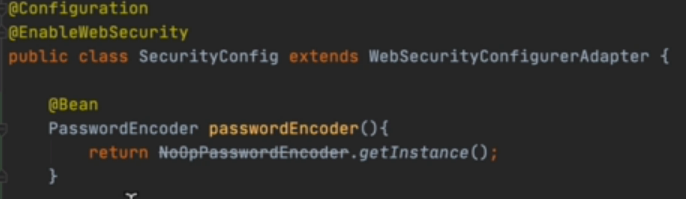
Create a class, here we are copying and paste the method “createDelegatingPasswordEncoder” from “PasswordEncoderFactoriesClass” and modify according to our need.



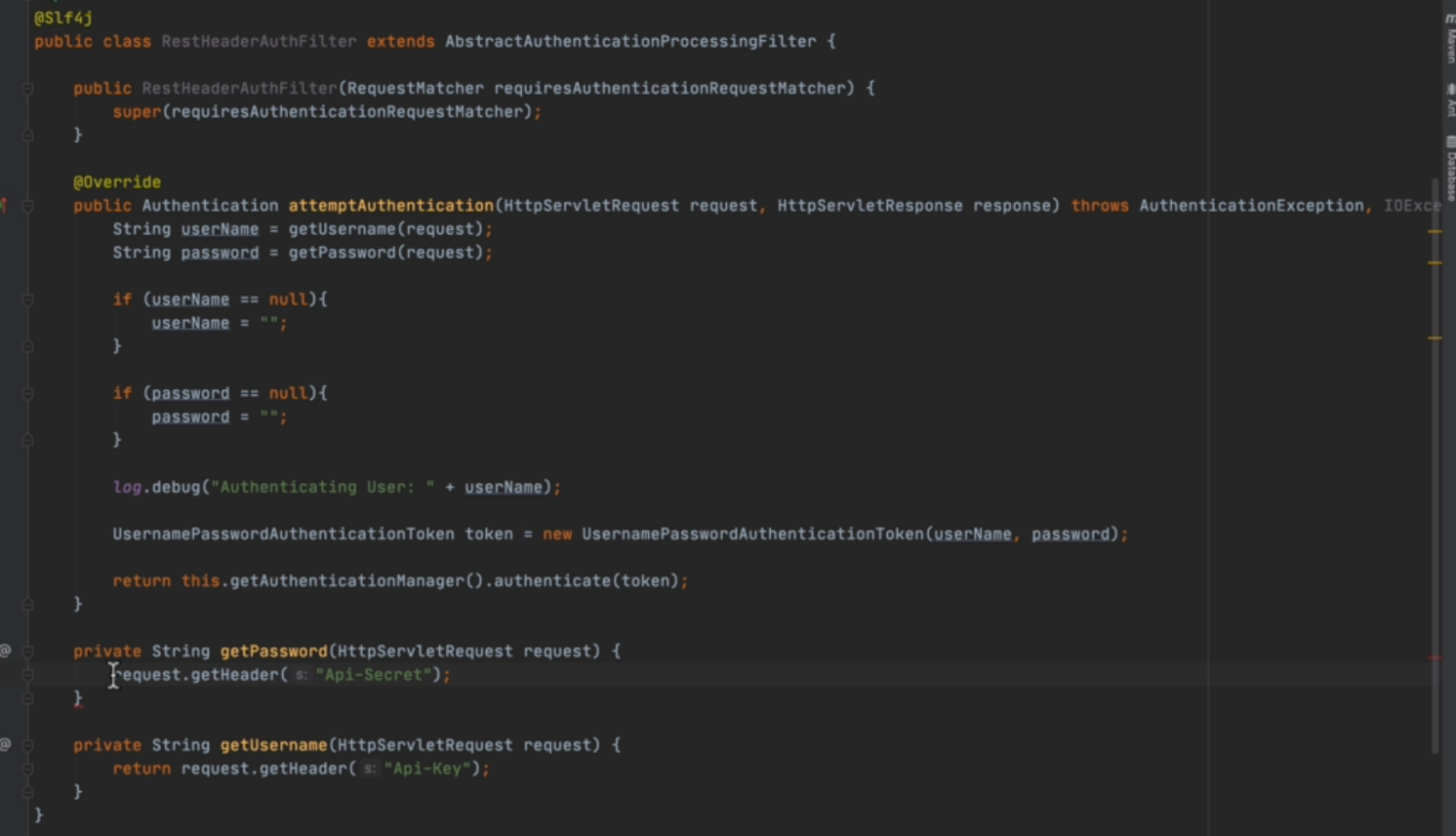


**Note:** By defining a specific password encoder bean, we can avoid doing {encodertype}”password” to “password” directly.

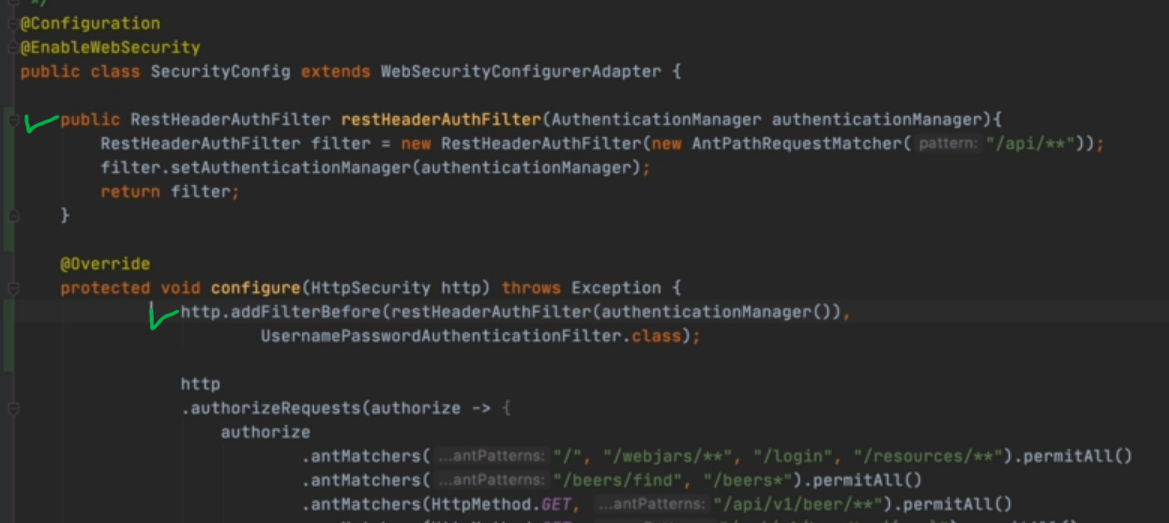
Example:



5. custom authentication filter

# Add a customAuthenticationfilter class extends AbstractAuthenticationProcessingFilter 

#Add this custom filter to security configuration



*Security debugging configuration:*

