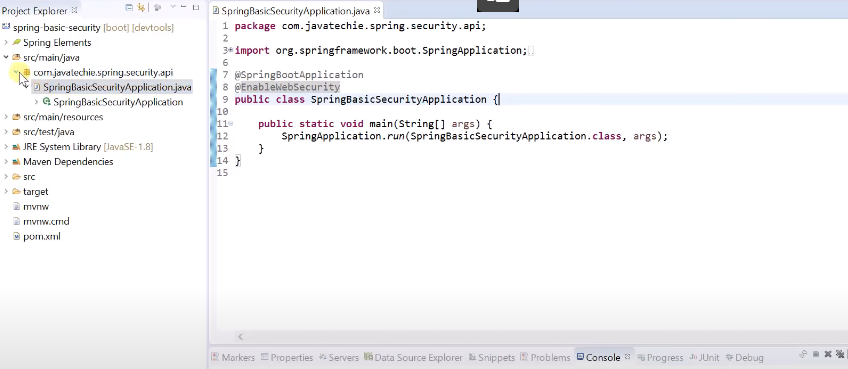
**Lecture 1 authentication**

1.First make a springboot application

And add annotation to it



2.Make a controller

A computer screen shot of a code

Description automatically generated

3.Mention the username an dpasword

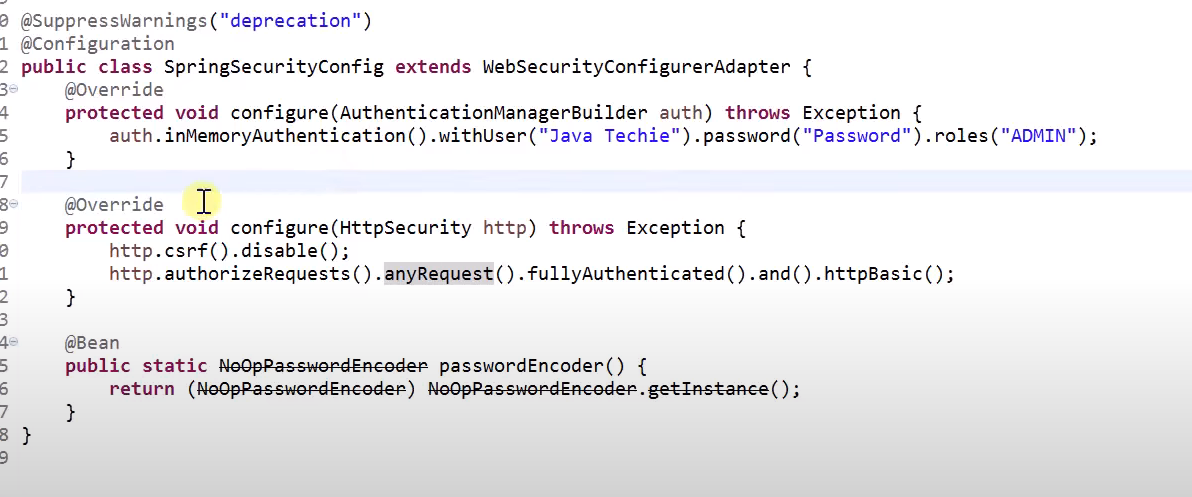
A screenshot of a computer

Description automatically generated

4.Now we will configure the userame an dpassword differently and comment the one which we have declared in application.properties

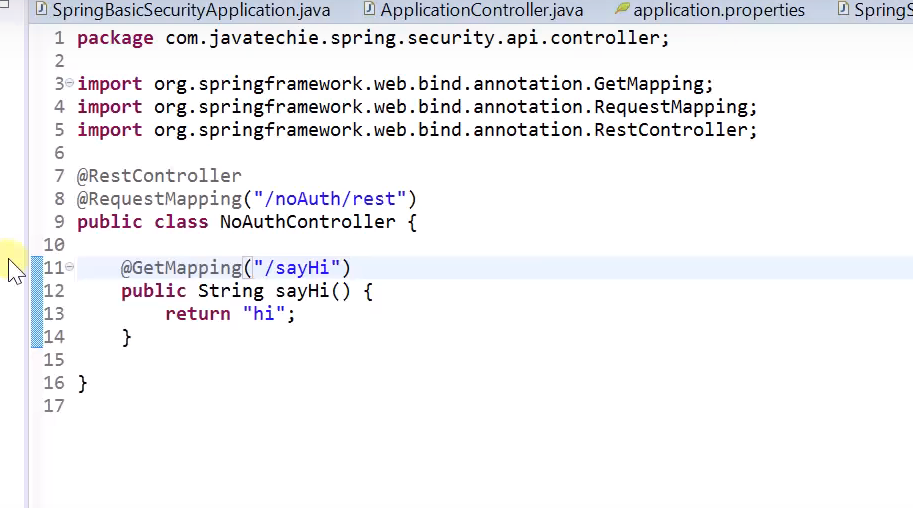
So we ill make a configuration class with the above mentioned two methods.

We will be using the inmemory authentication

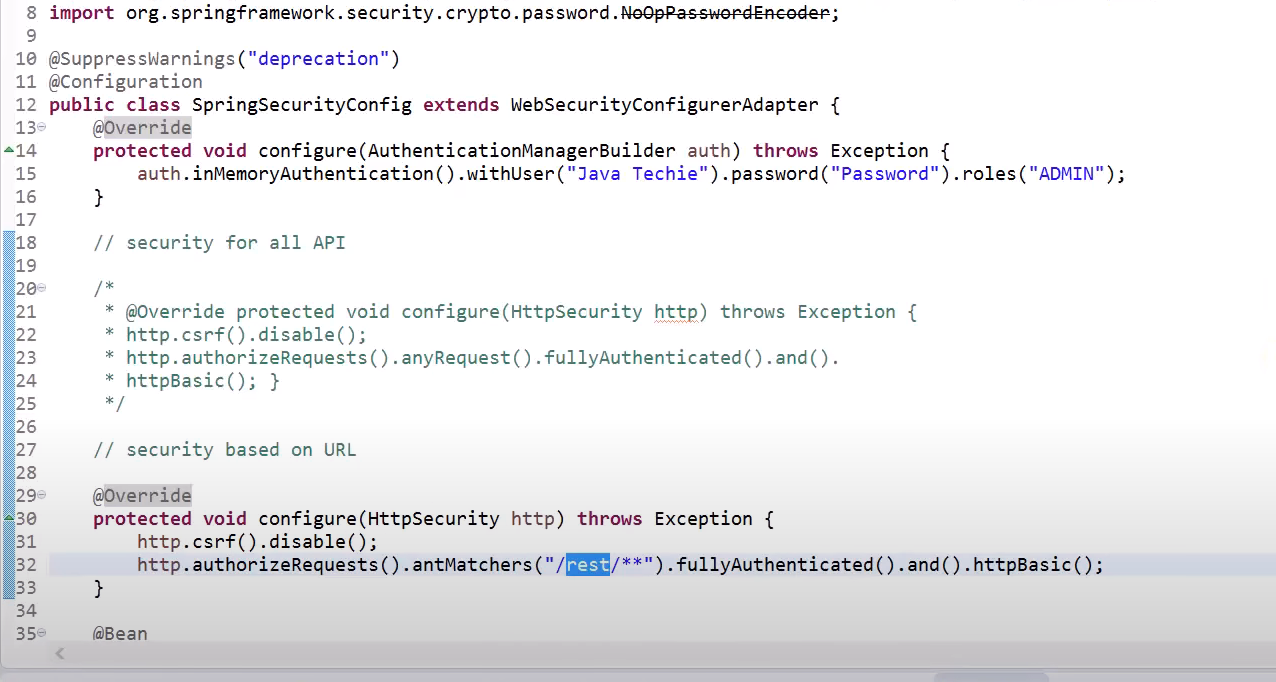


5.Now as we see above all the endpoints of the above mentioned controller will be authenticated

Now we ant a controller whose endpoints should not be authenticated

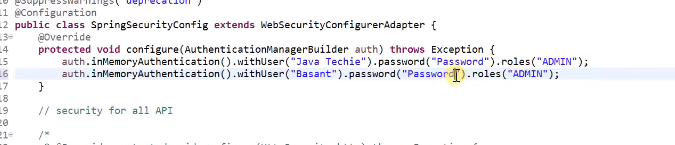


6.We want that only request coming to any endpoint starting with rest/ should be authenticated



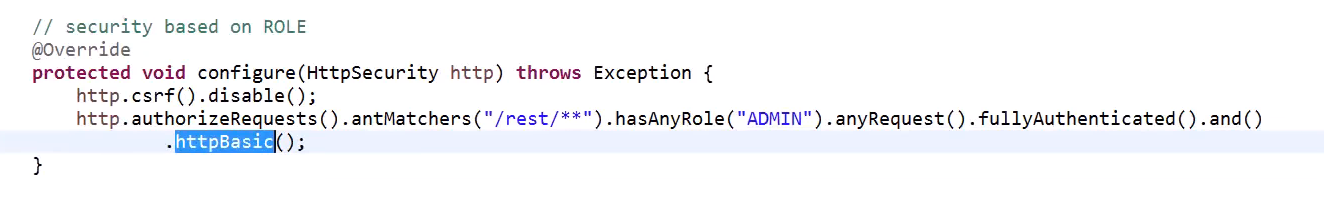
Thus we mention the above code for that and the ones which come to our other controller will not be authenticated.

7.Now lets make a  **role based authentication**

****

**H**ere in the above code we have mentioned the roles of various users.

8.So we will now implement the role based authentication.

  
in the above when any request comes to the /rest/\*\* endpoints then they will be authenticated.they will be popped up with a username and password.now when the username and password is put then it will be checkec if the user is having the role as admin or not.if not then he will not be authorized to enter.

**LECTURE 2**

1.Make a new project and add a spring security to it.

Make certain packages as controller,service,config,repository,model

Now make a new class in model

A screenshot of a computer

Description automatically generated

Now make a role class

A screenshot of a computer

Description automatically generated

3.now application.properties

A screenshot of a computer

Description automatically generated

4.makde a userrepository

A screenshot of a computer

Description automatically generated

Make a role repository

A screen shot of a computer

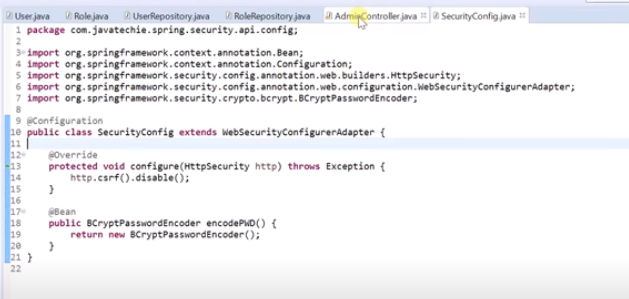
Description automatically generated

5.now add a controller

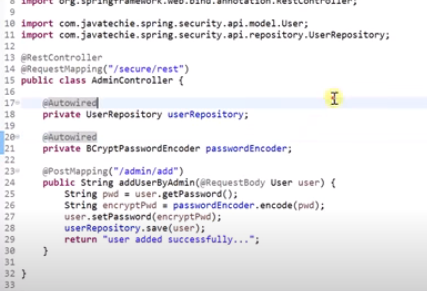
A screenshot of a computer

Description automatically generated

6.Now make a configuration class

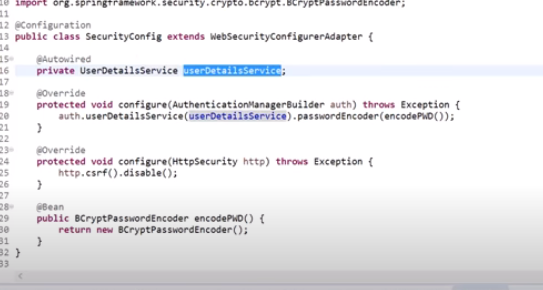


7.nw we will add code to save the user



8.after we run the request then users will be saved in the database.

9.now we wil do the authentication using the configure



Userdetail service is an interface we will make its implementation

A screenshot of a computer

Description automatically generated

10.name make a userdetails

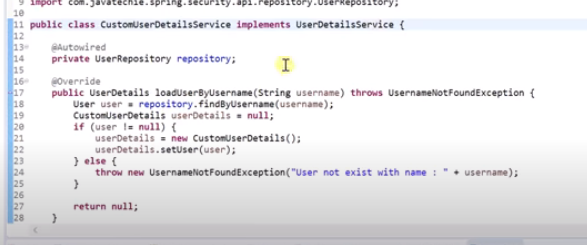
A screen shot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

11.now make changes in the userdetialservice



**HOW SPRING SECURITY WORK INTERNALLY**

First we will have a client who wants to send request to a server.So there will be a server in the middle who will authenticate our request

A diagram of a flowchart

Description automatically generated

The request from the client will come which will have the username and password it will be going to the delegatngfilterproxy who will make an authrntication object

A computer screen shot of a login

Description automatically generated

Now how filter know how to authenticate

Filter give this to authenticationmanager who will call the authenticate method.

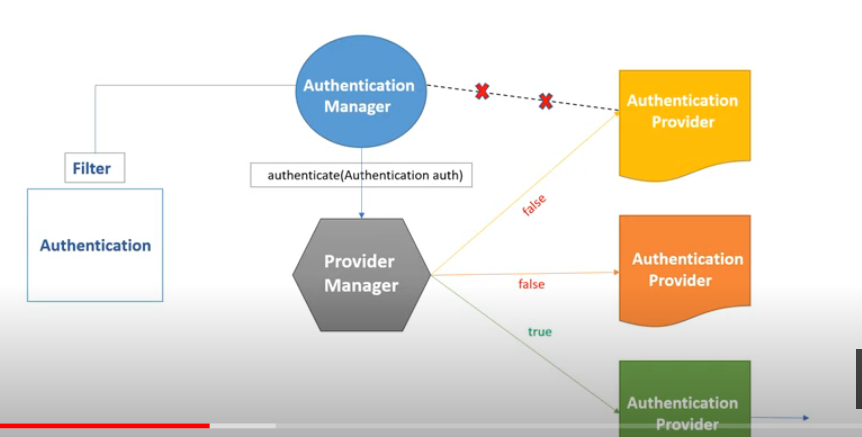
Now authrntication manager also doesn’t know how to authenticate so it will give to authentication provider.

There can be multiple authentication providers based on different mechanisms as one authentication provider knows token based authentication,other knows based on ldap etc.

It is difficult for authentication manager to decide to which provider should he redirect the request to.

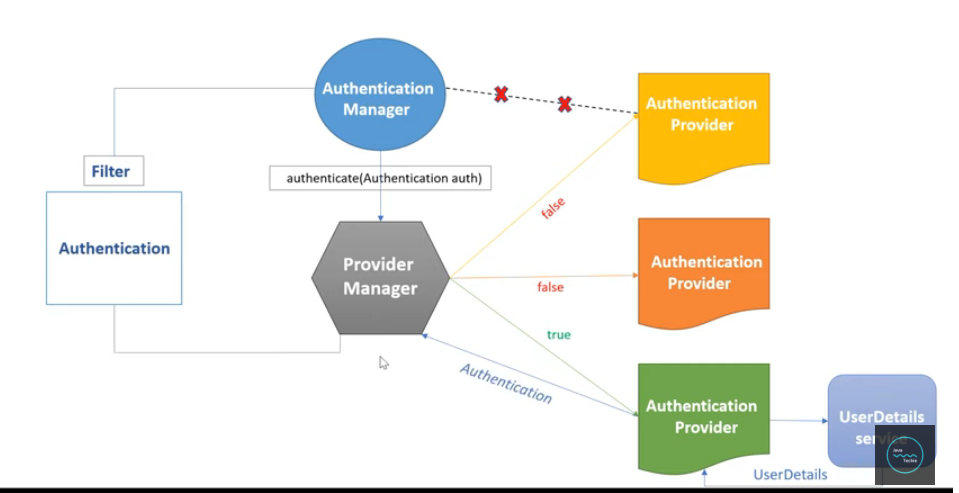
Authentication manager will give to Provider manager

Provider manager will check with each of the authentication provider by using the supports() method and decide the best authentication provider for that request.

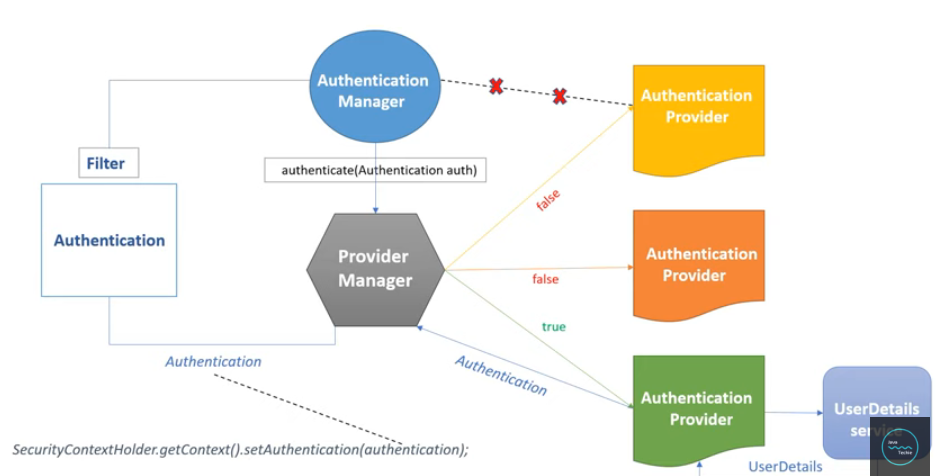


Now the authentication provider need to have information about the user details object.

Provider will take help from userdetails service who will check with the database for that user.



After the authentication is done provider manager will send back to the autherntication filter and set the authentication in the securitycontext



**NEW SPRING SECURITY CHANGES**

The websecurity configure class is romoved

We can make a bean of userdetials service and add the details in that

A computer screen with text

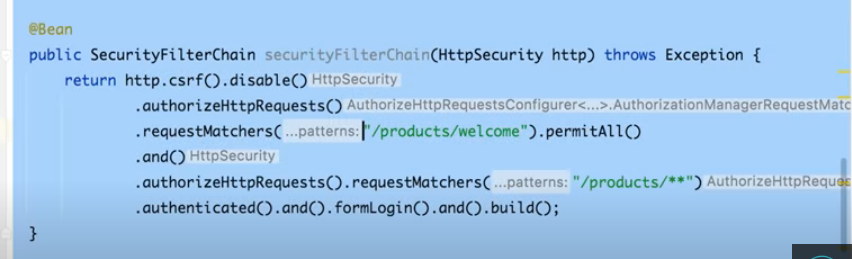
Description automatically generated

A screenshot of a computer

Description automatically generated

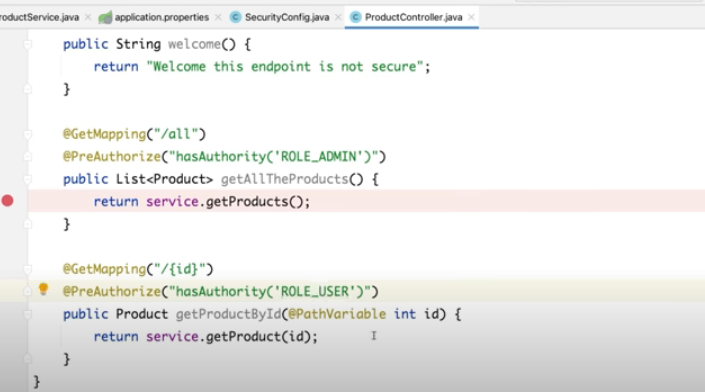
This above is what we do for authentication

2.for authorization we have anew class which we can make a bean

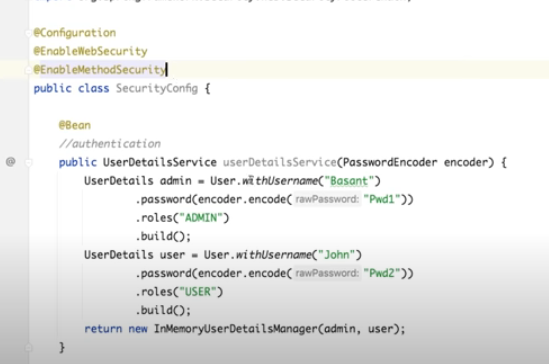


3.now if we want a method to be accessed only by some authorized person then we can do In controller

:



4.to enable method level security we need to add the following annotation



Until now we have hard coded the username and password we ned to load it from the db so we will do as below:

Make an entity



Now make a repository

A screenshot of a computer

Description automatically generated

User repo extend from jpa repo

Now we need to to authrntication by fetching from the db

A screenshot of a computer program

Description automatically generated

So we will make anew class UserInfoUserDetaislServcie which will help us to fetch values from he db.

A screenshot of a computer

Description automatically generated

Now what we are getting from the db we need to convert it to the userdetails object so we will make accustom class to implement the userdetials.

A screenshot of a computer

Description automatically generated

Public class UserInfoUserDetails implements UserDetaisl

A screenshot of a computer

Description automatically generated make the other required changes as well

Now in your service do the below changes

A computer screen shot of a computer code

Description automatically generated

Mention in application.properties file

A screenshot of a computer

Description automatically generated

Now we make an authentication provider

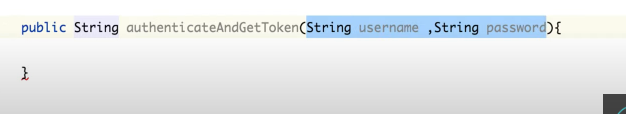
A screenshot of a computer

Description automatically generated

**JWT**

In order to work with jwt

First time the user who comes will give his username an dpassword and then he will get the token so we will make a controller endpoint where a user can send his username and password and get a token



We can also make use of an entity to do so.

A screenshot of a computer

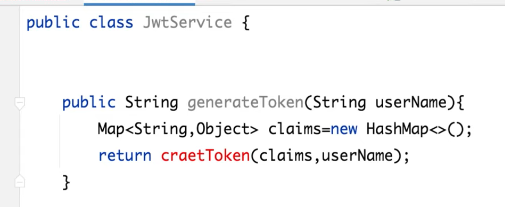
Description automatically generated

Now will pass the auth request object from the postman

A screenshot of a computer

Description automatically generated

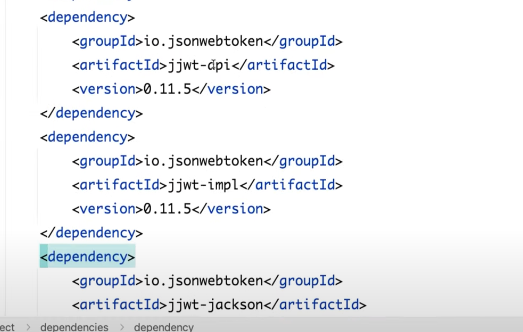
Now to create a token we will make a new java class as JWT SERVICE



Claims is all the parts of a token which include the header,payload and the signature.we make an empty claims map above

Now to create a token we need help of jwt

So we add the below dependencies

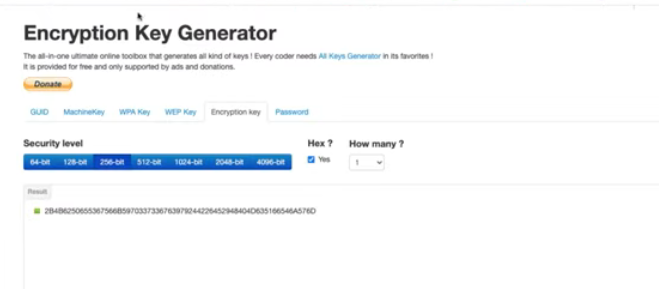


Now come to JwtServcie and do as below:



We will write a method to sign with get the signkey

To generate this key we need to make some secrte



We can go to encryption key generator and get the key from there in hexa format.



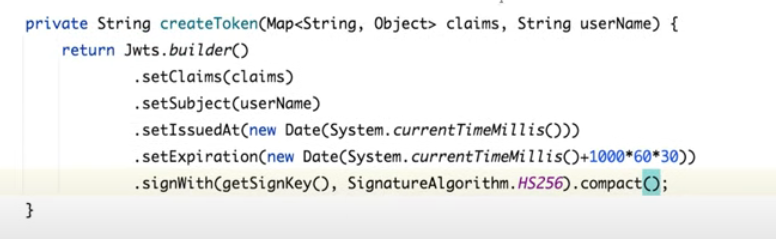
**A screen shot of a computer

Description automatically generated**

****

**We are using the secrte the one which we copied from the above site.**

Now we will specify the algorithm that we want to use



Now we will call this method to generate the token

A screenshot of a computer

Description automatically generated

Now as no one will be able to use this endpoint so in config class we need to bypass this endpoint so we will ake change sin he config class

A screenshot of a computer

Description automatically generated

We will do it a permit all.

No we will run this.

And we can generate a token. A screenshot of a computer

Description automatically generated

We will get token as return.

Now with the above approach we are giving some random username and password and we arw getting the token

But this is not right

We ant only an authenticated person to get the token

So we will make the below changes

**So before giving the token we need to authenticate the user**

**So we will inject the authenticationmanager**

****

**A screenshot of a computer

Description automatically generated**

**New UsernamePasswordAuthenticationToken(authrequest.getname,authrequest.getpassword)**

**We** need to explicitly define an authentication manager bean

Go to security config and define authentication manager.

A screenshot of a computer

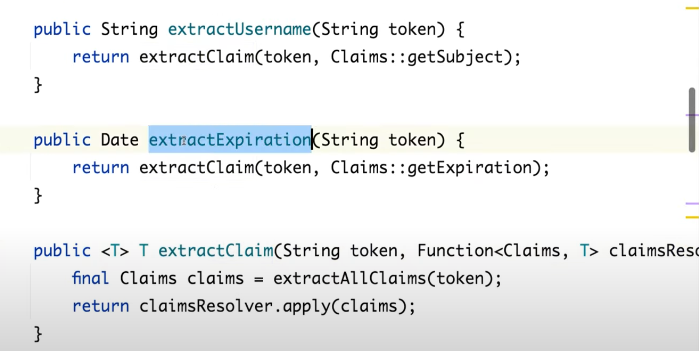
Description automatically generated

**NOW WITH TOKEN LET US CHECK IF WE ARE ABLE TO REACH OUT TO THE ENDPOINTS OR NOT.**

Not we will need to validate a token



Now go to jwtservice

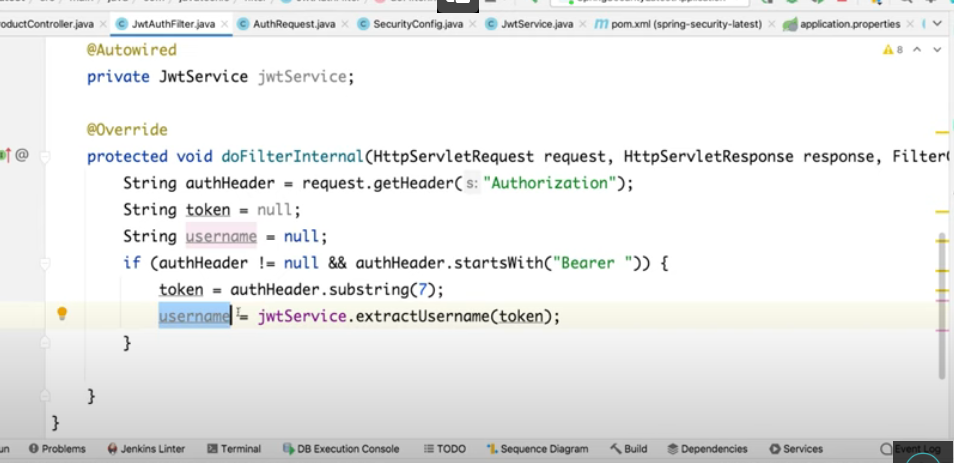


A screenshot of a computer program

Description automatically generated

A close-up of a computer code

Description automatically generated



By taking

