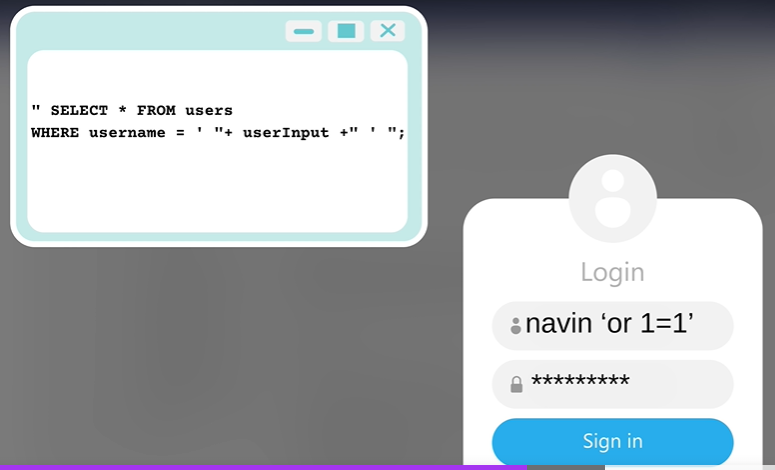
**Spring Security**

**Jun 5,2024**

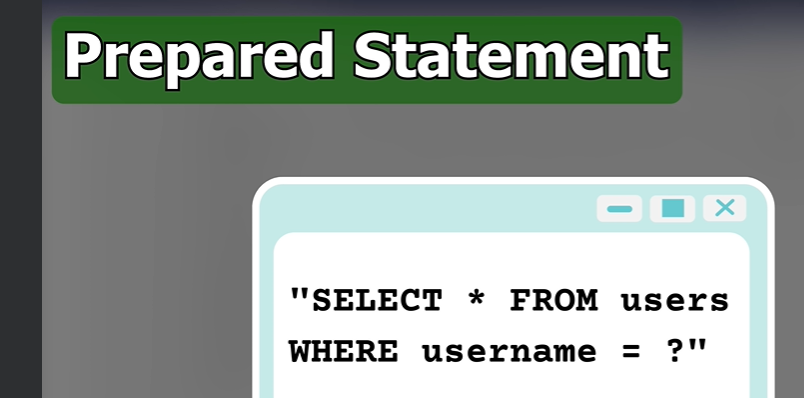
[**https://owasp.org/www-project-top-ten/**](https://owasp.org/www-project-top-ten/)

**SQL injection :**



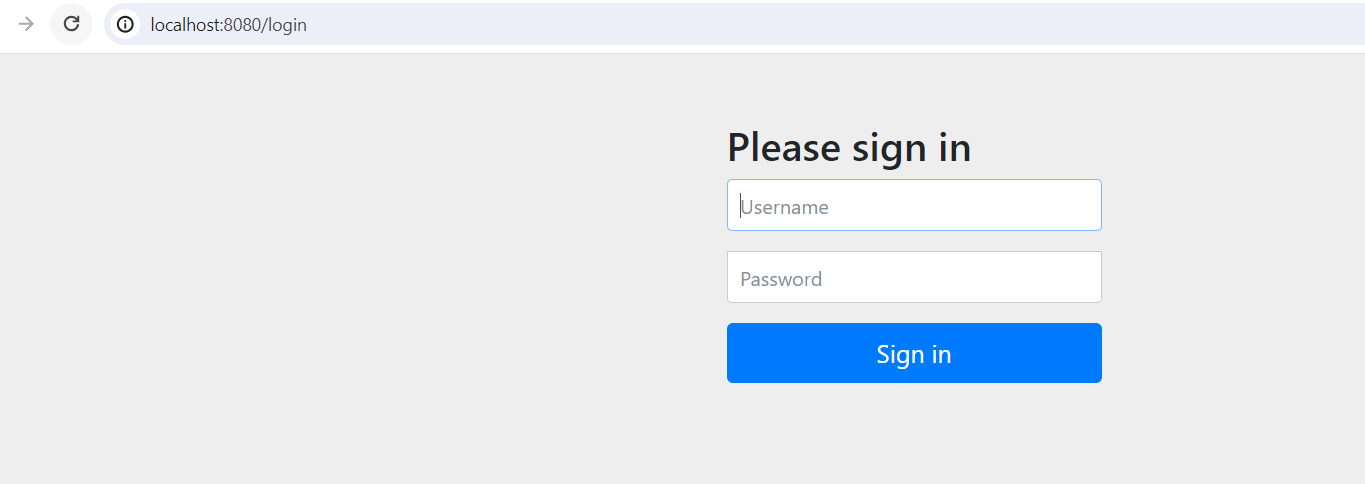
Instead of navin if I pass navin or 1=1 then in where clause of query it will become true.

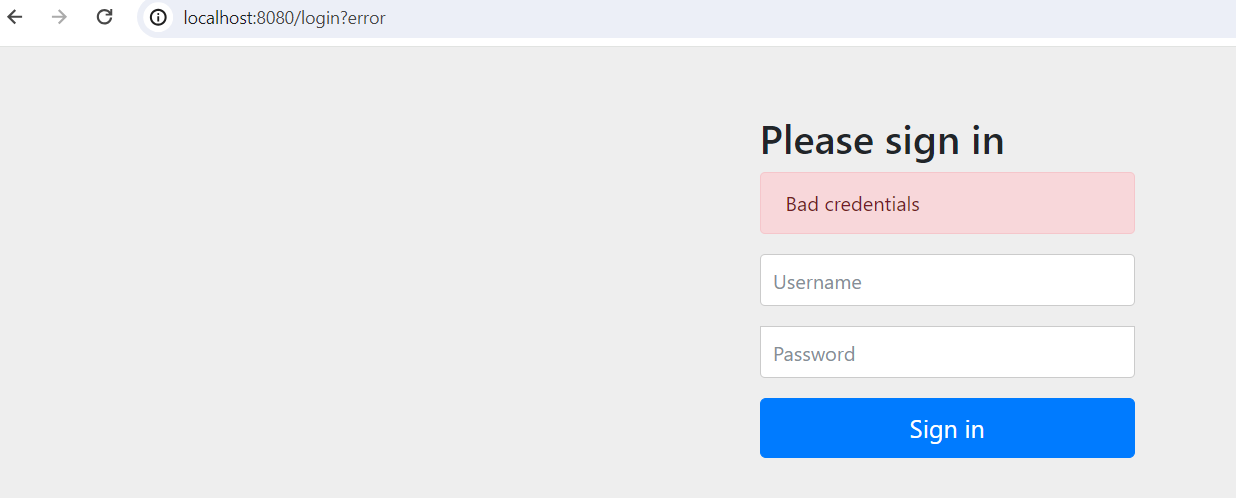
Way to avoid SQL injection is by using prepared statement.



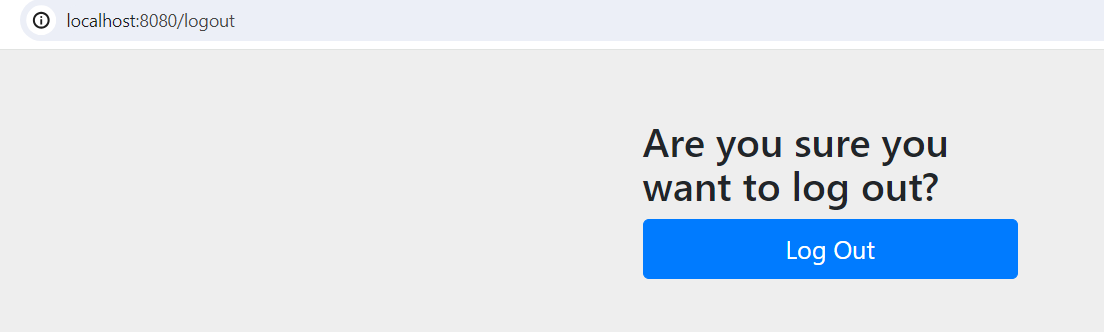
In **prepared statement it takes the value with type.**

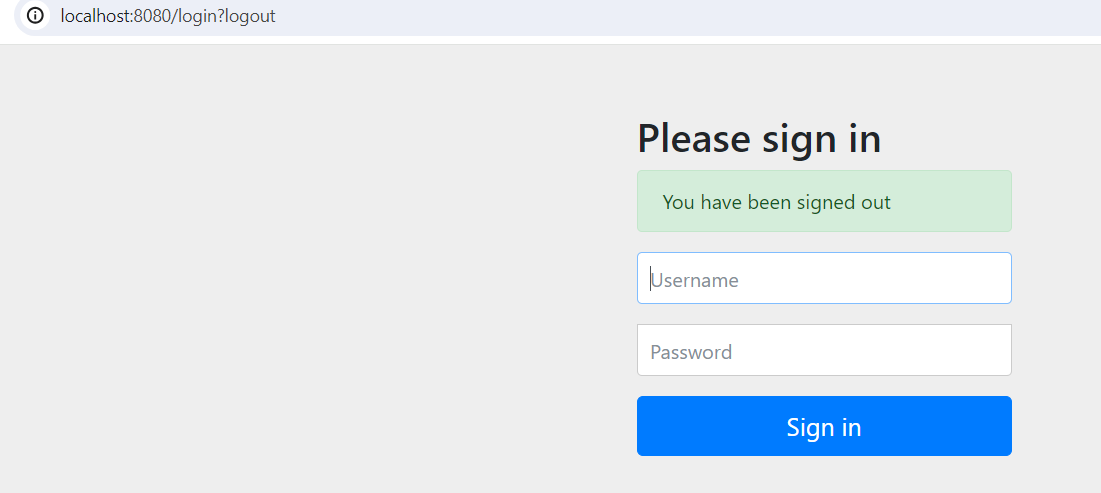
**With and without spring security dependency ?**

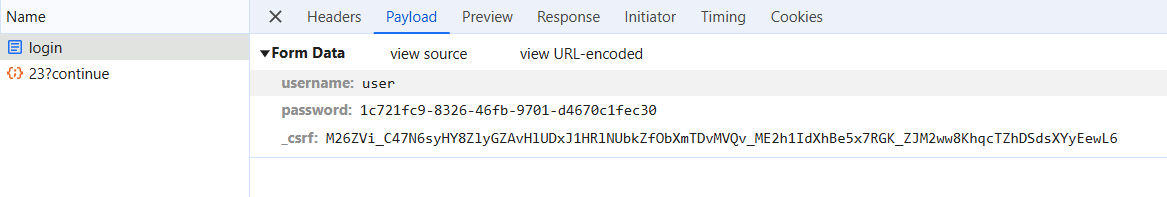
Without spring security dependency if you create a controller that will generate string output it will easily get displayed on the console.  
After adding spring security dependency and hit the same url without any code changes it will display you the below screen.  


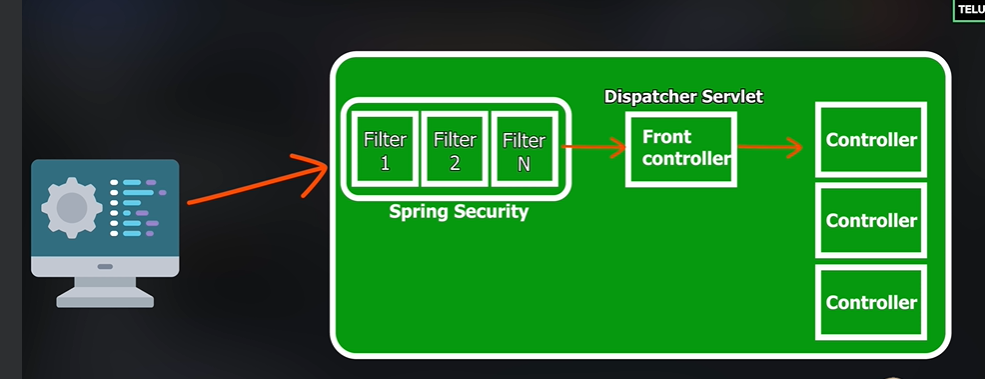
By default the user name is ‘user’ and the password is provided to you in the console.  


For wrong credentials it will show above screen.  
For right credentials it will show the output string on the screen.

If you want to log off then hit, 8080/logout

After clicking on logout , it will redirect you to login page again to signin. 



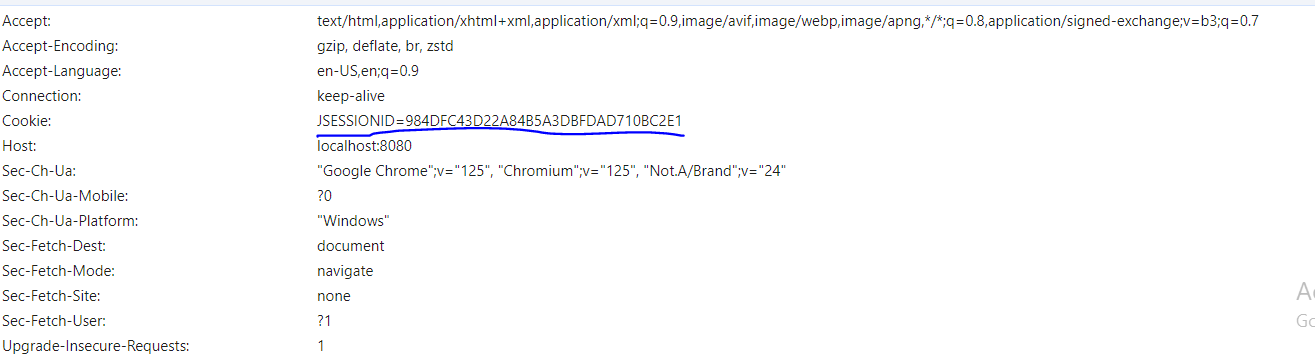


**Filters provided by spring security are :**

[org.springframework.security.web.session.DisableEncodeUrlFilter@5fe2bf66, org.springframework.security.web.context.request.async.WebAsyncManagerIntegrationFilter@410e066f, org.springframework.security.web.context.SecurityContextHolderFilter@4e5a5b5, org.springframework.security.web.header.HeaderWriterFilter@7f2d0611, org.springframework.web.filter.CorsFilter@6437ae55, org.springframework.security.web.csrf.CsrfFilter@4bf3f162, org.springframework.security.web.authentication.logout.LogoutFilter@4607b53a, org.springframework.security.web.authentication.UsernamePasswordAuthenticationFilter@493d9437, org.springframework.security.web.authentication.ui.DefaultLoginPageGeneratingFilter@76d3b578, org.springframework.security.web.authentication.ui.DefaultLogoutPageGeneratingFilter@141dd6b2, org.springframework.security.web.authentication.www.BasicAuthenticationFilter@32e8564f, org.springframework.security.web.savedrequest.RequestCacheAwareFilter@50493caa, org.springframework.security.web.servletapi.SecurityContextHolderAwareRequestFilter@61d93adc, org.springframework.security.web.authentication.AnonymousAuthenticationFilter@5e4e42d8, org.springframework.security.web.access.ExceptionTranslationFilter@4ccfadc2, [org.springframework.security.web.access.intercept.AuthorizationFilter@5336835](mailto:org.springframework.security.web.access.intercept.AuthorizationFilter@5336835)]

**Session :**

Once you login with username and password we can be able to access all the urls. The session id will be saved inside cookies. Once the session expires you need to login again.Once you logout also the session id with expire.

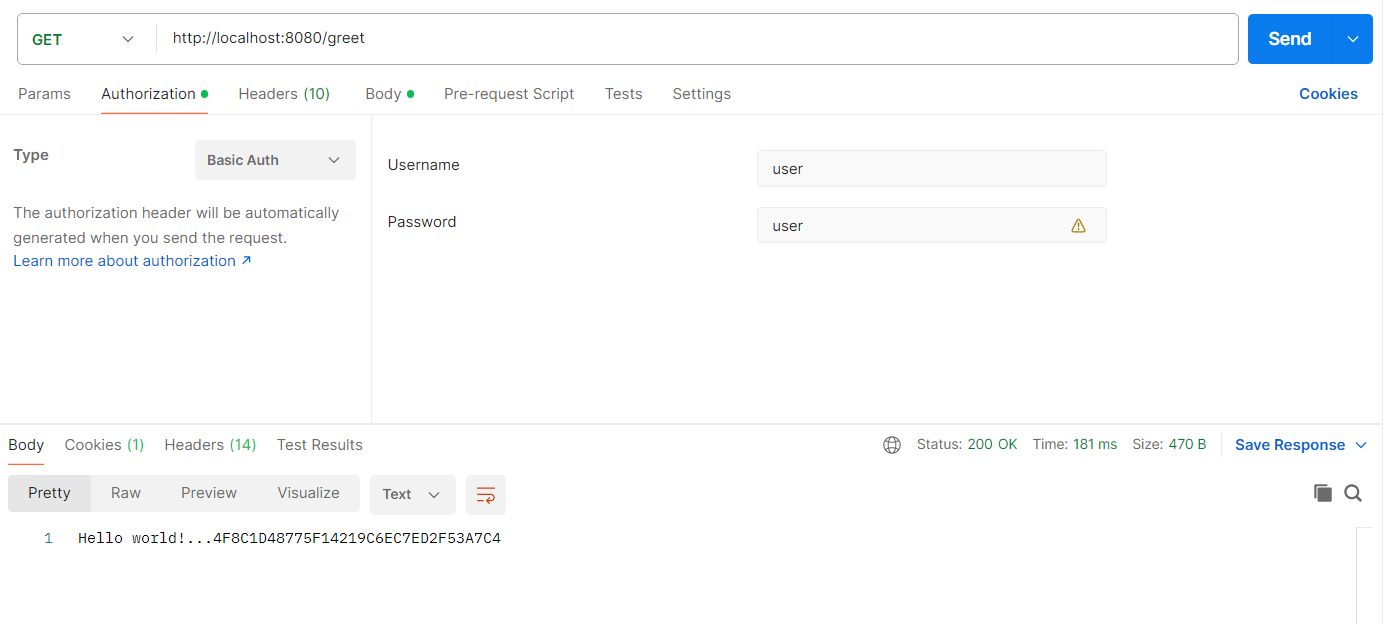


**Setting our own username and password :**

spring.security.user.name=user

spring.security.user.password=user

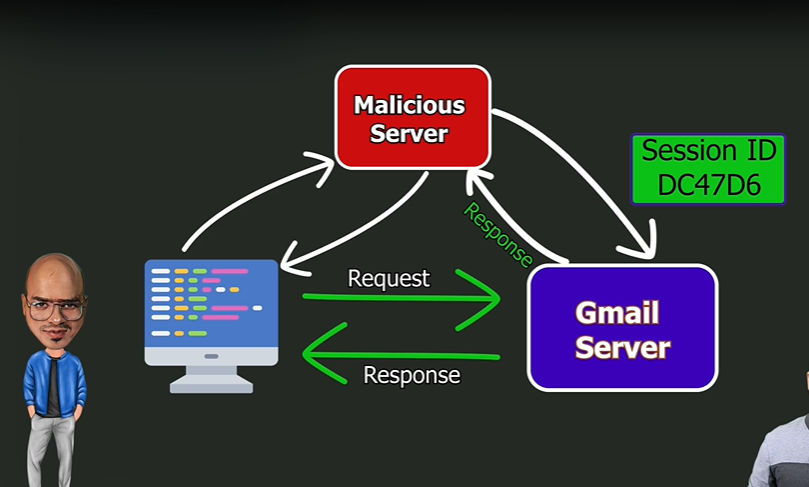
**How to access the url from postman ?**



**What is CSRF ?**

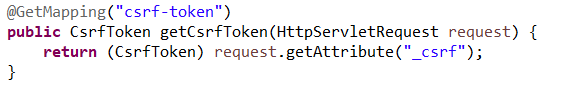
**Cross Site Request Forgery**

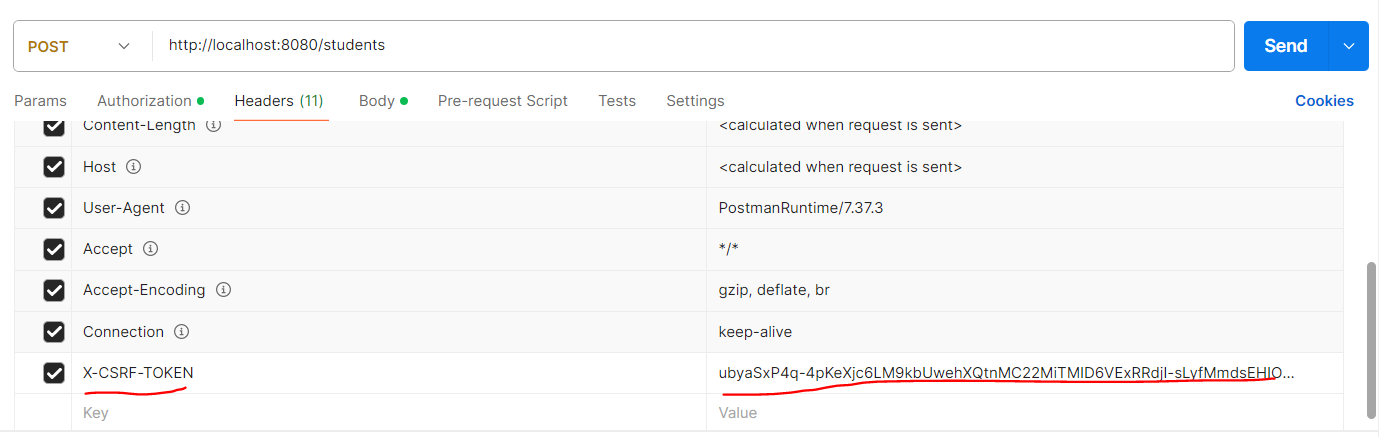
GET is the safest method because it doesn’t change the data on the server.  
By default spring security will apply CSRF on PUT,DELETE, POST not on GET.



**How to send CSRF token in POST request ?**

**How to get CSRF token ?**





**What if we tell that to use the session token from same website ? How can be do that ?**

server.servlet.session.cookie.same-site=strict

Types of API

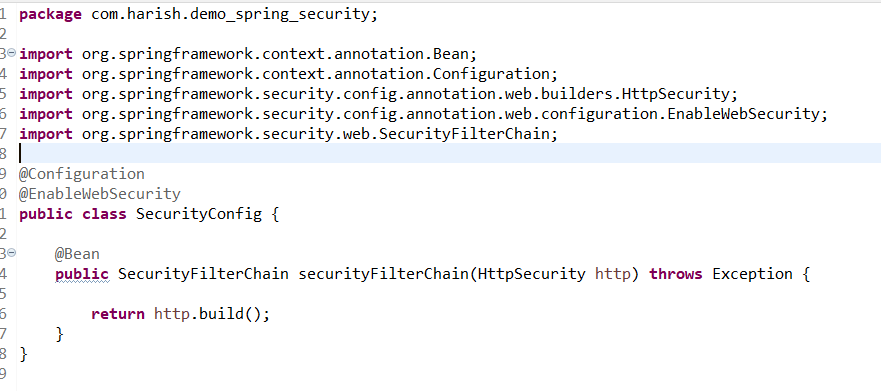
**Stateful**  - you maintain session

**Stateless** – does not require any session, need to send username and password (mostly used)

19th June, 2024

**How to disable the security features provided by spring ?**

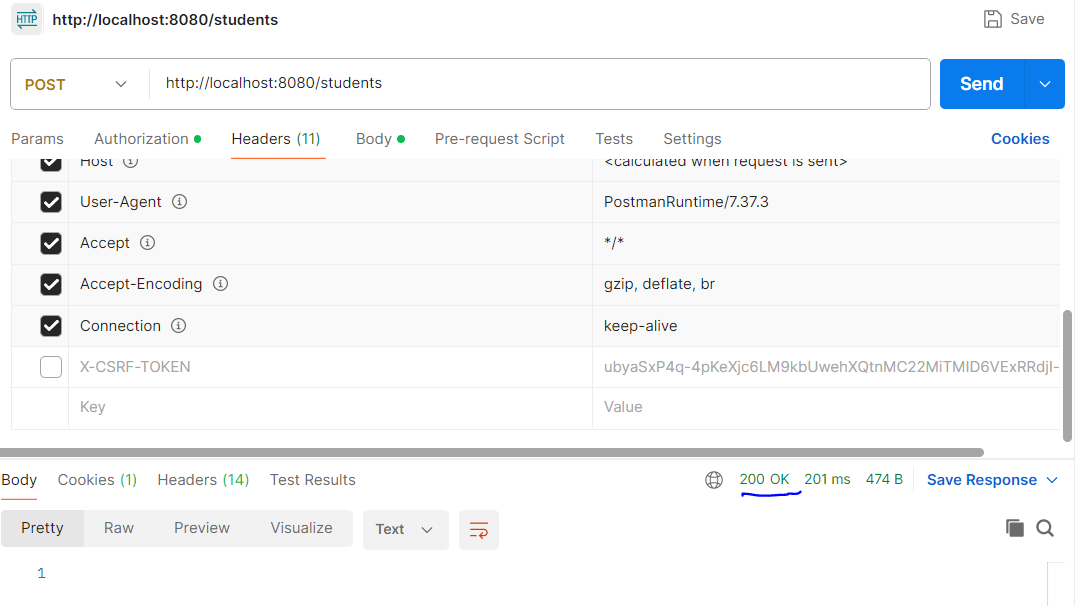
By using custom class



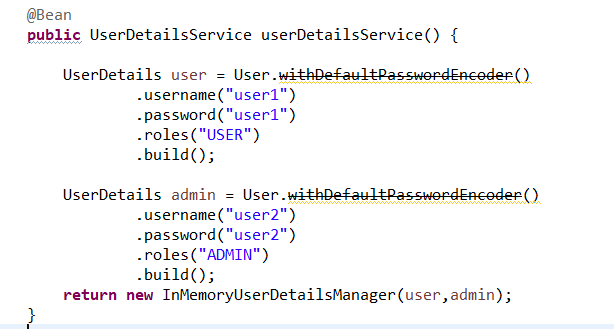
With this if you try to hit any api, it won’t ask for any user name or password and directly it will display data without spring security.

**To make it stateless we need to disable CSRF token, how to do it ?**

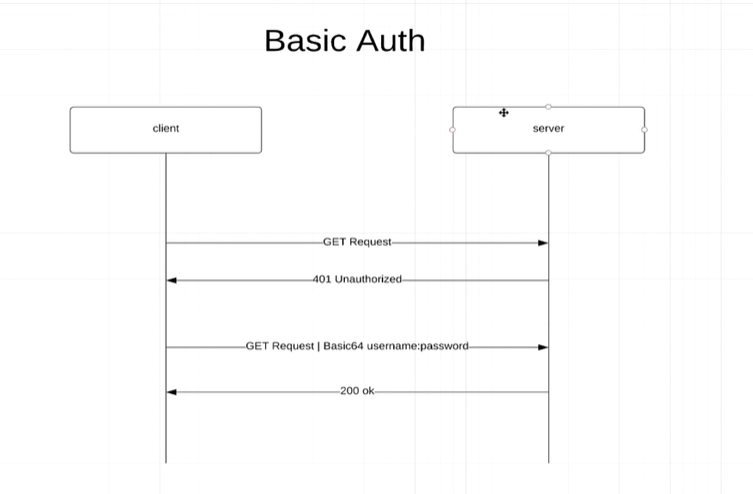


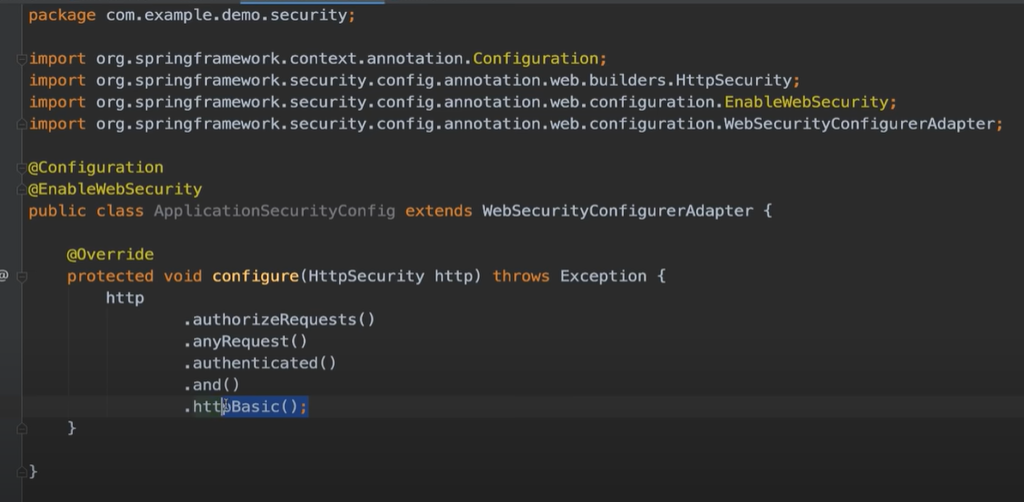
For every request new sessionId will be generated and we don’t need to pass csrf token in headers for POST, PUT , etc.. methods.  


**Currently we are hardcoding username and password, what if multiple users trying to access the API?**



By using like this we can allow these two credentials to access our api’s. UserDetialsService is by default provided by spring security.This is not good way since we are hardcoding the credentials, actually we need to get credentials from database. Lets check on how to get it from database next.

Basic Auth Authentication :  
  




Can’t replicate the code in local because of WebSecurityConfigurerAdapter class.

For each and every request you need to authenticate with user name and password.

DisAdv : you can’t logout as in form based authentication above.