**Spring Security**

Spring Security is powerful and highly customizable authentication and access control framework for Java application.

Spring Security is a framework that focuses on providing both authentication and authorization to Java application.

**Authentication**

Authentication is a process to verify the identity of user.

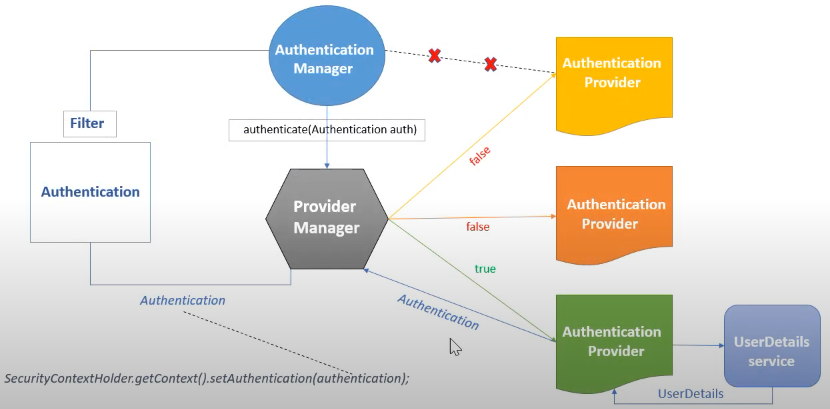
Spring security supports various authentication mechanisms such as form-base, basic, digest, OAuth, JWT.

**Authorization**

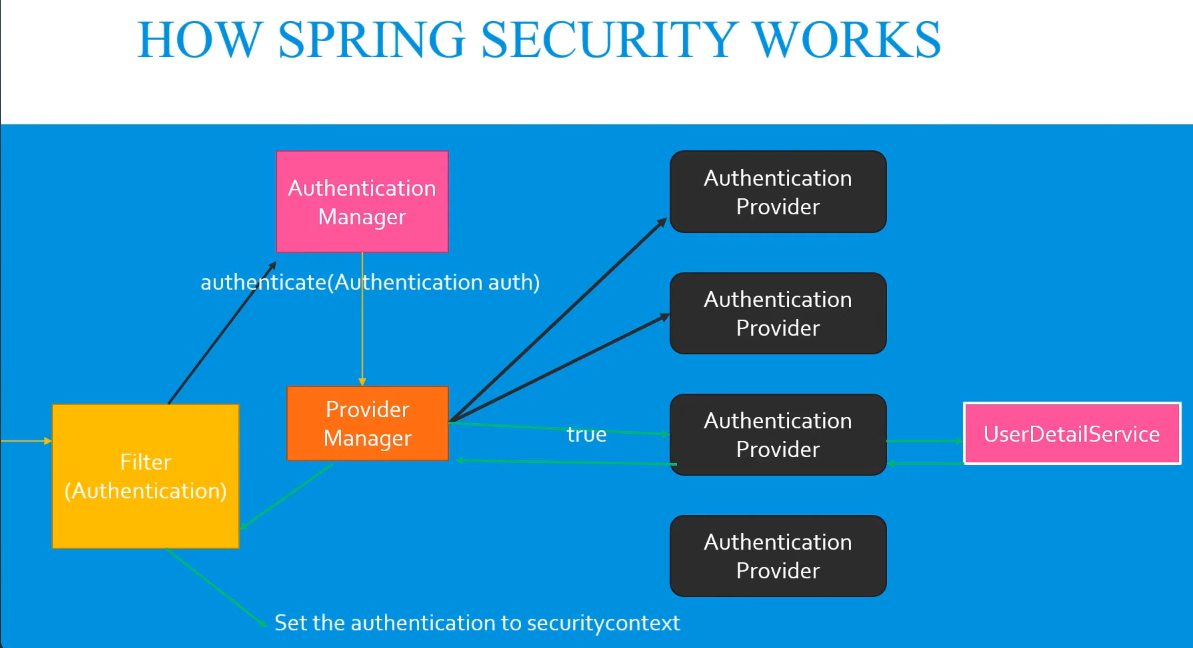
Authorization means it gives the permission to user to access a specific resource or functions after successful Authentication.

It also provide user based access control authorization where user can access a resource as role bases assigning.

**How Spring Security Works**



1. **Request will redirect to filter**
2. **Name of this filter is DeligatingFilterProxy which is given by Spring Security**
3. **This will convert servlet request in Authentication Object**
4. **This Authentication object is contained username and password as principal object and also credential object.**
5. **Filter dose not contain any logic to authenticate this request**
6. **Now Filter will delegate this request to Authentication Manager**
7. **Authentication manager take this authentication object as an argument and it will call authenticate(Authentication auth) method**
8. **Now authentication manager also doesn’t know how to authenticate it**
9. **Now Spring framework provide multiple Authentication Provider based on authentication mechanism**
10. **So, Authentication manager will delegate this request to Provider manager**
11. **Multiple authentication provider having different logic to authentication**
12. **Provider manager will take this responsibility to identify the appropriate authentication provider to perform this authentication mechanism**
13. **Now this provider manager will go each and every provider and provider will call support method to check this kind of authentication mechanism or not**
14. **Now if some of the provider will support this mechanism then provider manager will delegate this request to this authentication provider**
15. **Now authentication provider will take the help of UserDetailsServce to authentication**
16. **Now UserDetailsService will go to externl source like database or inmemory chache**
17. **If it found the correct details the it will return the this details or if not then it will throw exception**
18. **Once it found the details it returns back the authentication provider**
19. **Now authentication provider will authenticate it and it will return valid authentication object to Provicer manager**
20. **Now Provider manager will return this object to Filter**
21. **Filter will set that Authentication object to security context**
22. **Once Authentication object is set to security context the request will redirect to your server side application**



First Request is came to Filter (Authentication)

Filter will create object of Authentication with basic information’s and it will forward this request to Authentication manager.

Now authentication manager responsibility is to authenticate to this request is valid or not.

But this manager is abstraction mean it has not implemented feature so for that there is number of authentication provides are available like Token base, Password base and so on.

Now this authentication manager will call Provide manager.

Provider manager will check which best Authentication provider for application.

Authentication provider will take an help oh UserDetailService.

UserDeatilService have data and now Authentication provider will call authenticate function.

After this function it will return valid authentication object and in this object having validity to Provider manager.

Now provider manager will provide this valid authentication object to Filter.

Lastly Filter will set this authentication into SecurityContext.

Now application works with authenticate APIs with valid user.

**Why Spring Security is needed?**

We need a secure application which will access or used by only and only genuine users for that we need Spring Security. Spring Security is provide a features like authenticate the user and give the authorization to user for accessibility of application.

**How Spring Security is used to secure your application at a high level?**

For Example

We have Hotel application

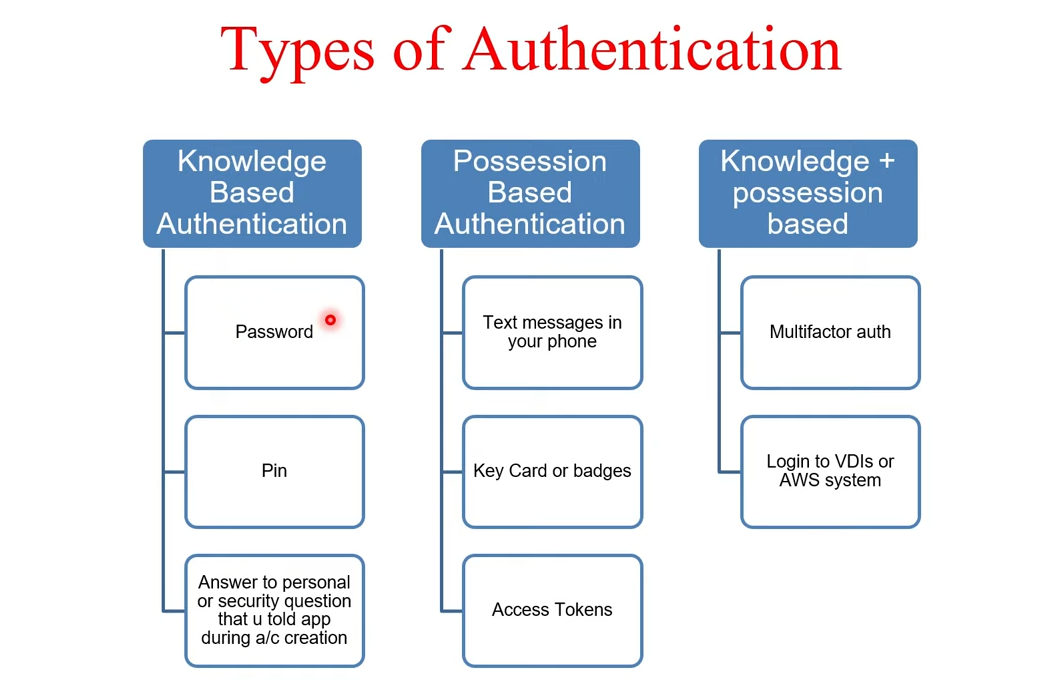
First we need to go in reception desk and ask there to access the room booked for XYZ name.

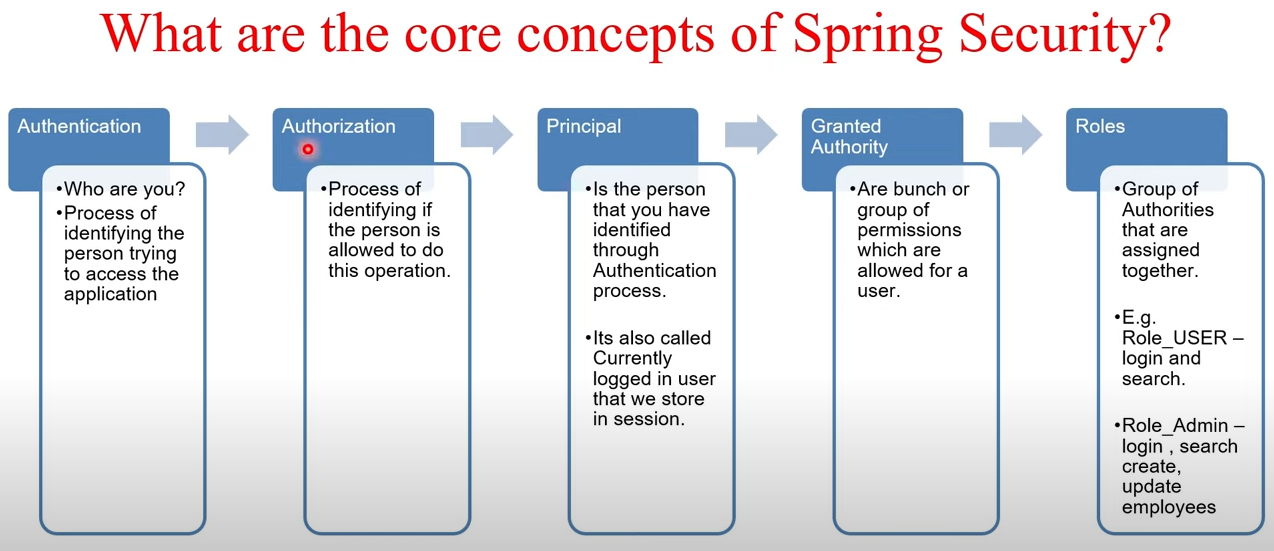
Now this reception desk will ask user id and password.

Now the user will tell the user id password to reception desk

Then he will give us access hot hotel room with some specific key which is authentication key

And reception desk will authorized this user to access hotel room.





**How we can implement Spring Security in our Application?**

First we need to add dependency in pom.xml file.

<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-security</artifactId>  
</dependency>

This will do our spring application secure with default user ID and random generated password to the application.

Now if we need to modify or add user ID and password for application so we need to add configurations to the application.

There is method to create the User ID password for the application.

But there is very basic way to add this we need to add property in application.property file

Spring.security.user.name=XYZ

Spring.security.user.password=XYZ

And other way is custom configuration class

We need to extend WebSecurityConfigurerAdapter class and Annoted with @EnableWebSecurity.

Now we need override the configure method with AuthenticationManagerBuilder parameter.

Now in latest spring boot WebSecurityConfigurerAdapter is depreciated

Other way is

We can create security configuration class.

Add annotation on this class as @Configuration and @EnableWebSecurity.

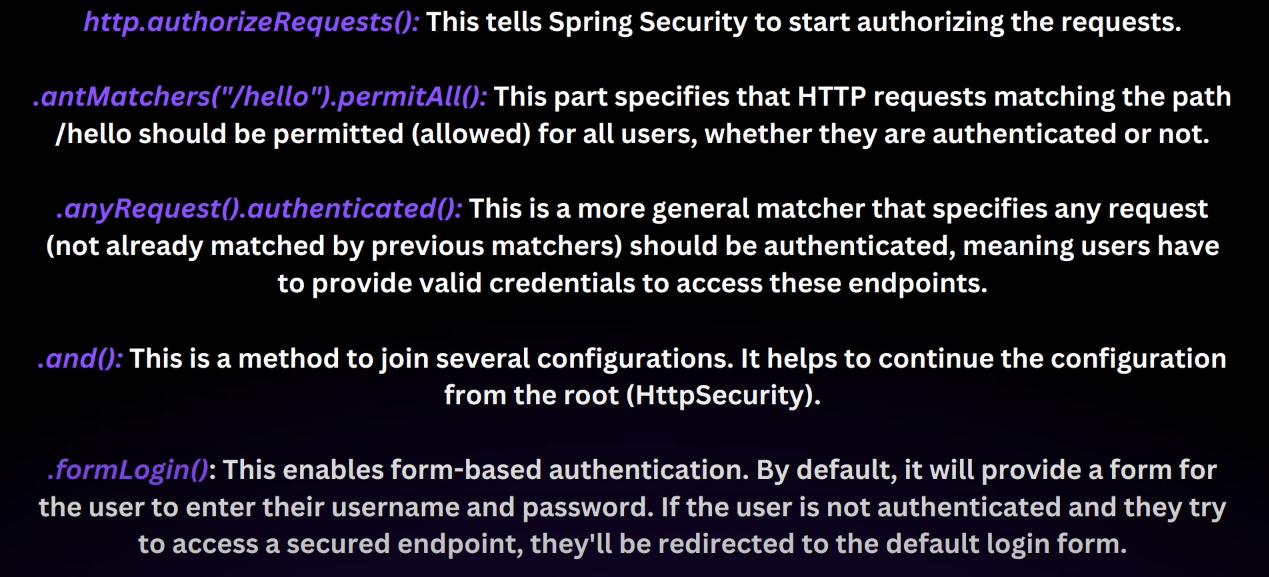
@EnableWebSecurity – this annotation tells to spring to enable its web security support.

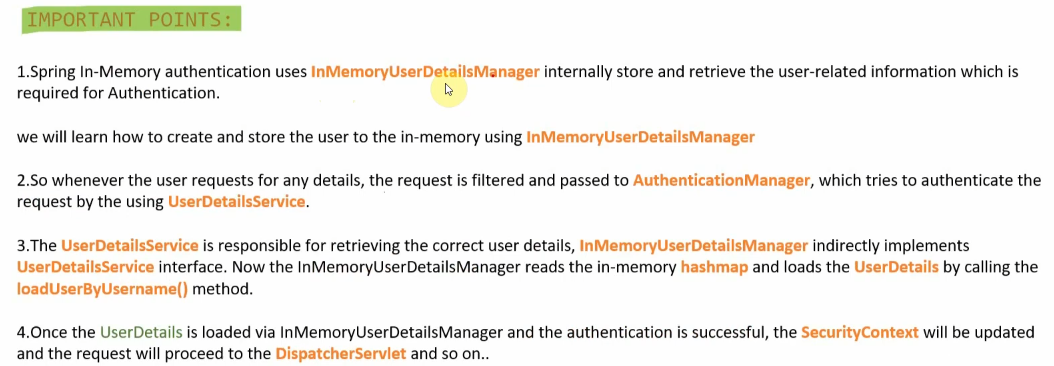
This annotation allow to default and customization of feature in security of application.

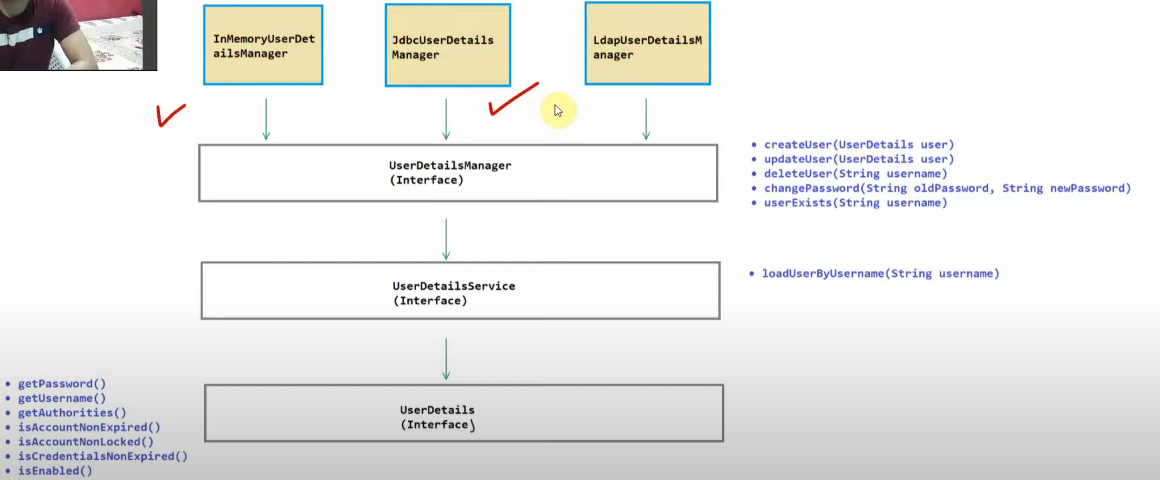
@Configuration  
@EnableWebSecurity  
public class SecurityConfig{  
 @Bean  
 public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {  
 http  
 .authorizeHttpRequests((authz) -> authz  
 .anyRequest().authenticated()  
 )  
 .httpBasic(Customizer.*withDefaults*());  
 return http.build();  
 }  
  
 @Bean  
 public InMemoryUserDetailsManager userDetailsService() {  
 UserDetails user1 = User.*withDefaultPasswordEncoder*()  
 .username("adminUser")  
 .password(this.passwordEncoder().encode("admin"))  
 .roles("ADMIN")  
 .build();  
 UserDetails user2 = User.*withDefaultPasswordEncoder*()  
 .username("normalUser")  
 .password(this.passwordEncoder().encode("normal"))  
 .roles("NORMAL")  
 .build();  
 return new InMemoryUserDetailsManager(user1,user2);  
 }  
  
 public PasswordEncoder passwordEncoder()  
 {  
 return NoOpPasswordEncoder.*getInstance*();  
 }  
}

**Now how work in background for multiple request for basic spring security.**

For example we create spring security application and we hit the request with basic auth in authentication and with username and password then it will share one **JSESSIONID in Cookies** with request and it will use this same cookies of jsessionid with multiple upcoming Request.







**How Spring Security Interception your Request?**

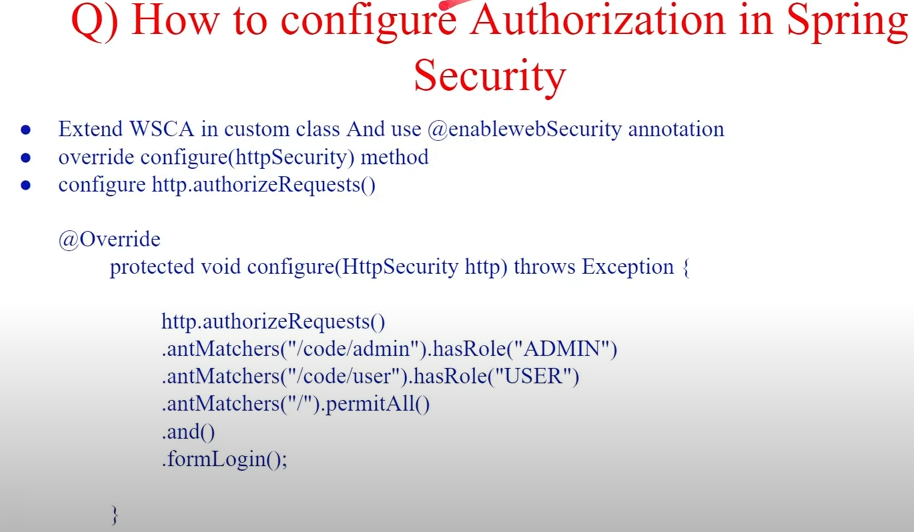
Because of filters of servlet

So filter is your receptionist validating all guests if they are genuine and legit user.

**How to configure the authorization in spring security?**

We need override this configure method with HttpSecurity parameter

And Using antMatchers and hasRole we will authorised a specific url.



**How dose spring security integration with OAuth2 for authorization?**

Spring security integration with OAuth2 for authorization by acting as a client that can request the application for access token from an OAuth2 provider.

It uses this token to authenticate and authorized the user to access the protected application resources.

When user can tries to access this resource Spring security will redirect them to OAuth2 provider for login.

After successful authentication the user get an access token to spring security which it will use to verify the users permissions and grand access to the resource.

This integration enables seamless access control in applications.

**Explain Cross-Origin Resource sharing (CORS) and how would you configure in a spring boot application?**

Cross Origin resource sharing allows a website to safely access resource from another website.

In spring boot we can set up CROS by adding @CrossOrigin to controller class or configuration it globally.

This annotation tells spring application which other web sites can use its resources and what type of request they can make and what header they can use.

This way we can control who can interact with our application keeping it secure while letting it communicate across different web domains.

**Explain SecurityContext and SecurityContext holder in spring security.**

In spring SecurityContext where the details about the currently authenticated user are stored, user details and granted authority.

The SecurityContextHolder class is helper class that holds the SecurityContext. It like container or storage space that keeps track of the authentication information of the current user throughout the application.

This makes it easy to access the user’s detail anywhere in the application ensuring that security decisions can be made based on the user’s authentication status and roles.

What do you mean by OAuth2 Authorization code grant type?

Spring Boot Security with JWT

Spring Boot Security with JWT with Database

Create entity class and impliment UserDetails.

Immpliment all method means override all methods.

Create class for UserDetailService and implement UserDetailServise and annoted this as @Service.

Implement all method in it.

Create repository interface and extend JpaRepository.

Create method in it.

Create bean in security config class doAuthenticationProvider.