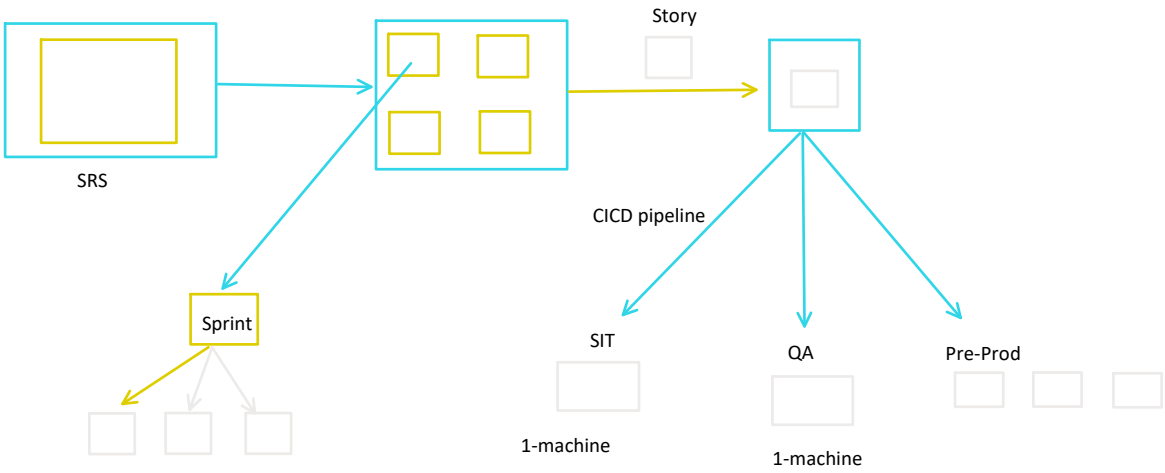
Spring Security

23 August 2025 09:32

Will make API and give end point to Ui developer, UI dev will make UI not the basis of requirement



Till Now is you API is secure ?

How to secure REST APIs using Spring Security

Fundamentals of spring security

1. Spring security is very important module for every web application
2. To protect our application & application data we need to implement security logic
3. Spring Security concept we can use to secure our web application / Rest APIs
4. To secure our app we need to add below starter in pom.xml file

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-security</artifactId>
  <version>3.5.1</version>
</dependency>
```

Note: when we add this dependency in pom.xml file then by default our application will be secured with basic authentication. It will generate random password to access our application.

Security = Authentication + Authorization

Authentication ==> Who are you ? (check username /pass)
Authorization ==> what can you do ? (Admin can approve loans, user can only check balance)

Without security all out APIs are public anyone can

1. Transfer money

2. Delete records
3. Access confidential data

How to override spring security default credentials ?

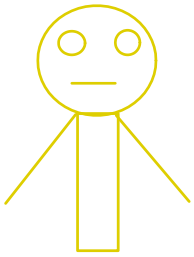
Mention below config in application.properties file

```
spring.security.user.name=root
spring.security.user.password=root
```

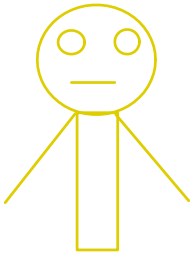
Project requirement:

We are developing banking application by default all apis are secure if we add security starter
But in real life not every APIs is need to secure

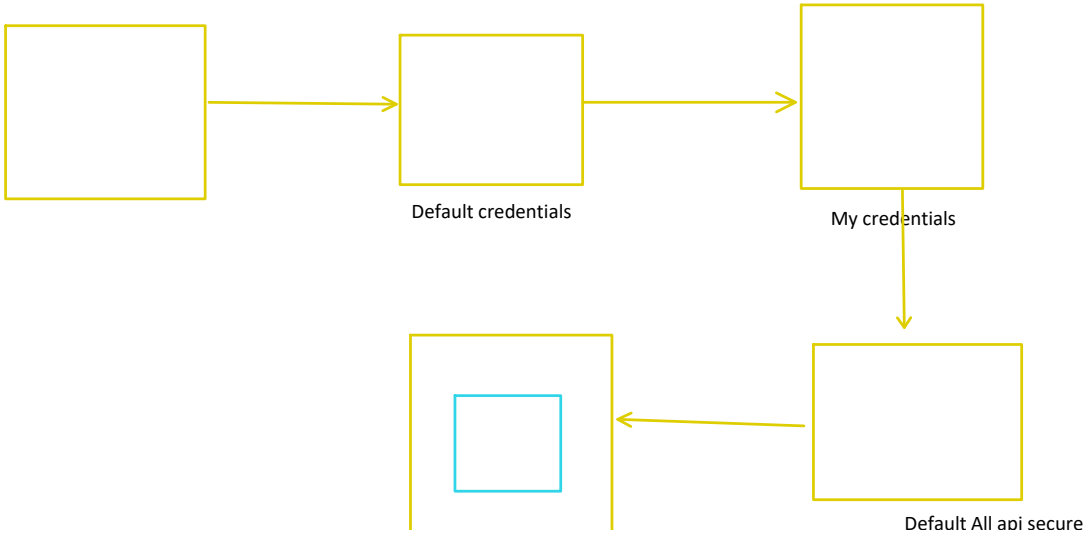
1. contactUs ---> public
2. /about ----> public
3. /transfer ---> private
4. /admin ---->private



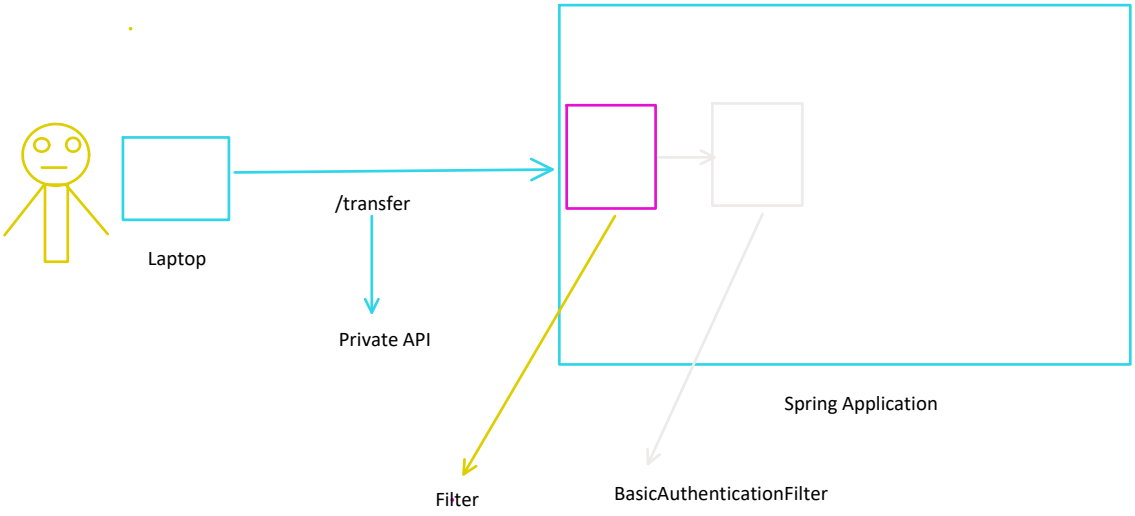
No need eat eggs



Fish hi khani



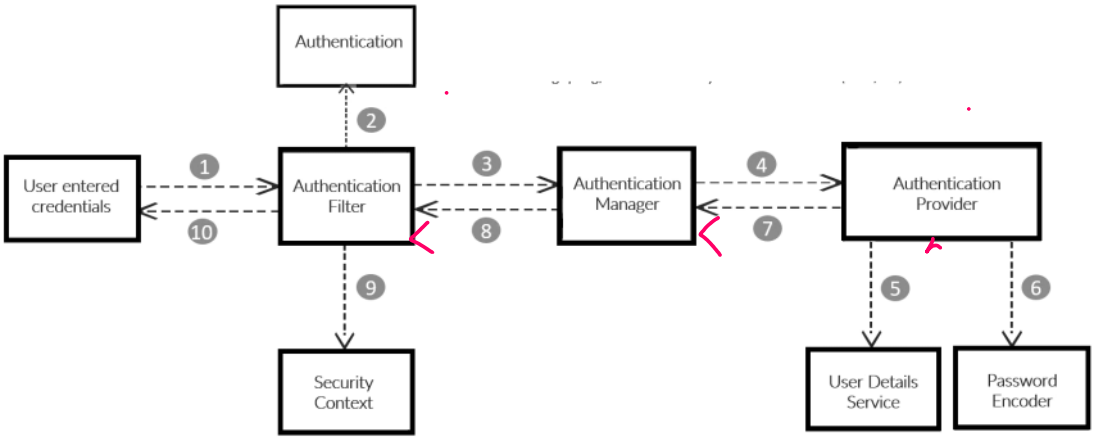
1. Spring boot starter = add all necessary classes for security
2. SecurityFilterChain = new way in spring boot 3.x instead of old WebSecurityConfigureAdapter
3. permetAll
4. Authenticated

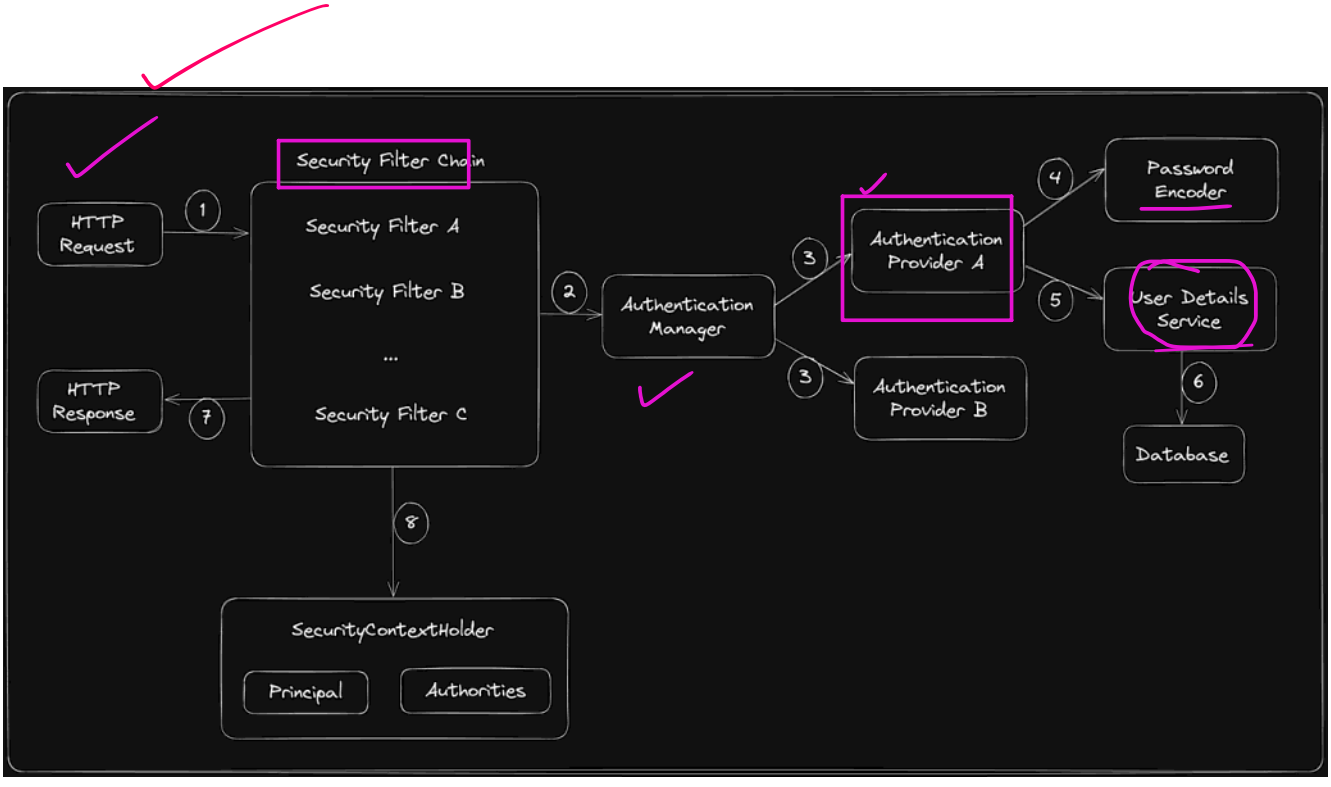


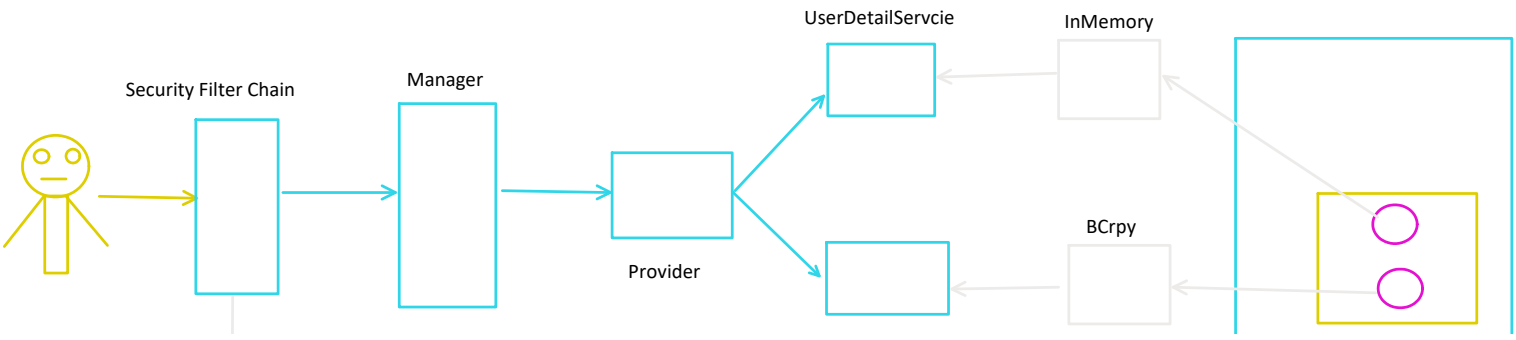
1. Request hits Security Filter Chain
2. BasicAuthenticationFilter Looks for Authorization : Basic header
   a. It will check username and pass
3. BasicAuthenticationFilter tell AuthenticationManager
   a. Call the UserdetailsService to fetch stored user info
   b. Use the passwordencoder to check pass
   c. These are the credentials please verify
4. UserDetailsService
   a. Loading user by username
   b. It useses InMemoryUserDetailsService
5. PassWordEncoder
   a. When we give password root spring runs it through the configured PasswordEncoder
6. Auntentication
   a. Username
   b. Password
   c. Will macth details
   d. If success then details will store in SecurityContext

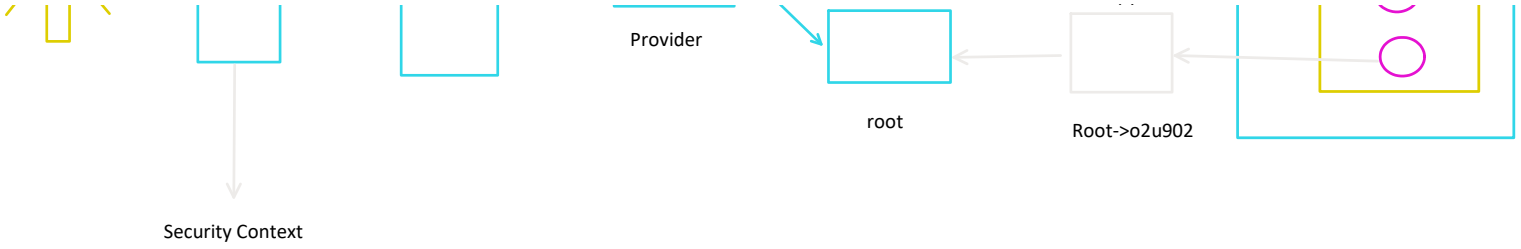
Note: Spring never store pass in plan text -> PasswordEncoder -> BCypt

Spring Security Flow



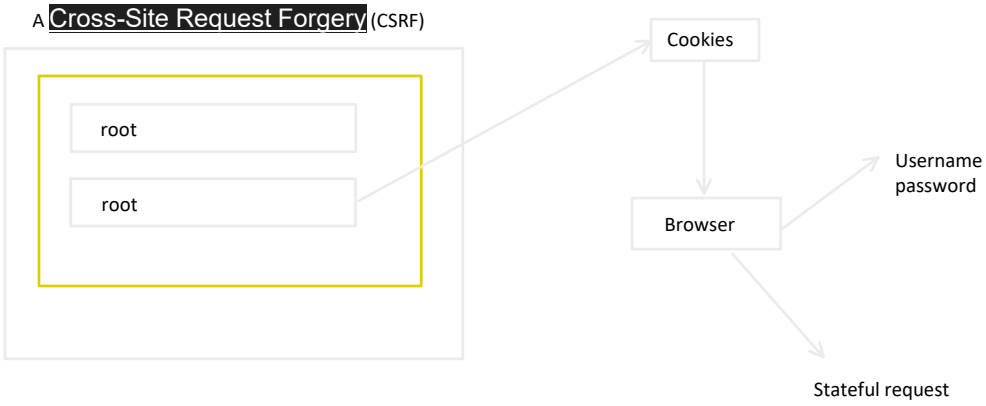


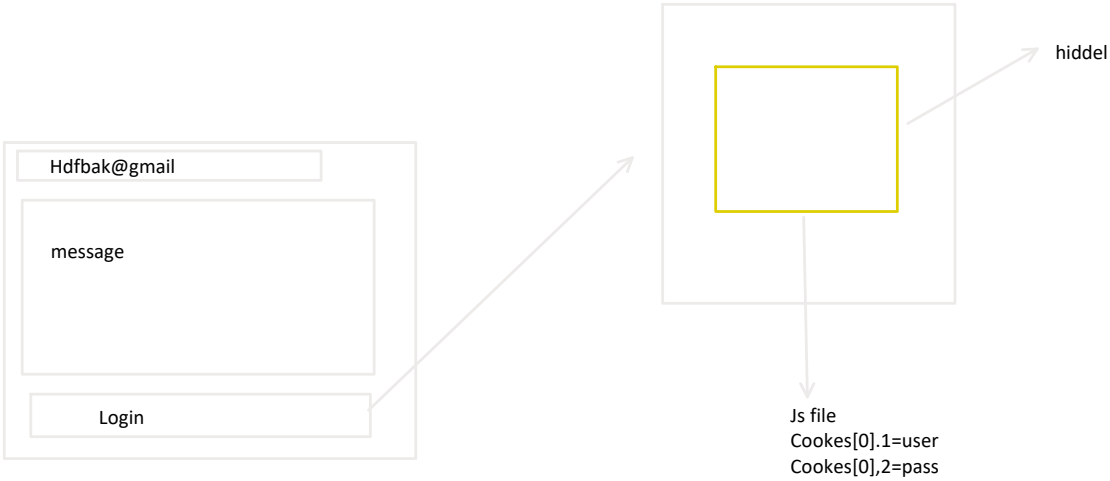




Security Context

A Cross-Site Request Forgery (CSRF)





Http request

1. GET
2. PUT
3. POST
4. PATCH
5. DELETE
6. HEAD
7. OPTIONS
8. TRACE

expectedHeaderName=X-Requested-With, expectedHeaderValue=XMLHttpRequest

React, Agular----> AJAX----->X-Requested-With

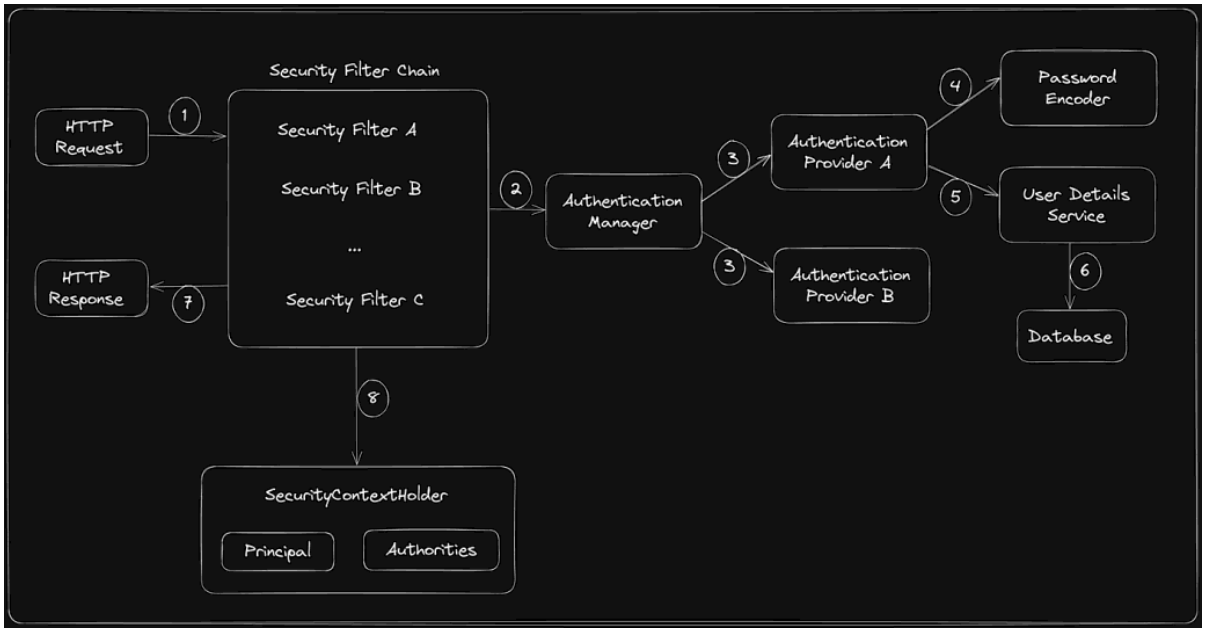
Csrf ----> statefull----> token

Knowledge Check
---

1. Spring security basic
2. Secure all api with default credentials
3. Change credentials
4. Change default flow and secure selected api
5. Understanding csrf
6. Disable csrf

Requirement = ROLE based Security
---

1. Balance api -----> user api
2. Close -----> admin

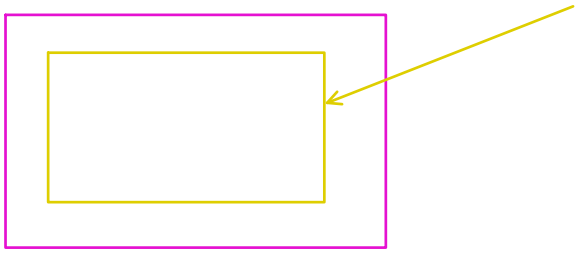



```
spring.security.user.name=root
spring.security.user.password=root
```

Note: By default spring security considered only ROLE_USER so use this annotations
@PreAuthorize("hasRole(ADMIN)") is useless

To use role based security we need to change default behaviour

Role bases security + Mysql + JPA
JWT



$2a$10$zRPNe.WmK6d7vv/WBRI7SOwwBxcusPWdUXl03J6yKMUAM5fF2852G
$2a$10$zRPNe.WmK6d7vv/WBRI7SOwwBxcusPWdUXl03J6yKMUAM5fF2852G