Spring Security

Конфигурационный файл:

```
 1. @EnableWebSecurity
 2. public class SecurityConfig
 3.     extends WebSecurityConfigurerAdapter {
 4.
 5.   @Override
 6.   protected void configure(HttpSecurity http)
 7.       throws Exception {
 8.
 9.     http.authorizeRequests()
10.
11.           // авторизированные пользователи
12.           .antMatchers("/authenticated/**")
13.               .authenticated()
14.
15.           // доступ по ролям
16.           .antMatchers("/admin/**")
17.               .hasAnyRole("ADMIN", "SUPERADMIN")
18.
19.           // доступ по правам authority
20.           .antMatchers("/profile/**")
21.               .hasAuthority()
22.
23.         .and()
24.           // всплывающее окно
25.           .httpBasic()
26.           // своя форма логина
27.           .formLogin()
28.           // url страницы для входа
29.           .loginProcessingUrl("/hellologin")
30.           // ...
31.           .successForwardUrl("/authenticated")
32.           // страница успешного входа
33.           .defaultSuccessUrl("/authenticated")
34.           // обработчик успешной аутентификации
35.           .successHandler()
36.
37.         .and()
38.           // страница после выхода
39.           .logout().logoutSuccessUrl("/");
40.     }
```

```
41. }
```

Стандартная форма входа:

```
 1. <form class="form-signin" method="post"
 2.     action="/security/login">
 3.   <h2 class="form-signin-heading">
 4.       Please sign in</h2>
 5.   <p>
 6.     <label for="username" class="sr-only">
 7.         Username</label>
 8.     <input type="text" id="username"
 9.         name="username" class="form-control"
10.         placeholder="Username" required autofocus>
11.   </p>
12.   <p>
13.     <label for="password" class="sr-only">
14.         Password</label>
15.     <input type="password" id="password"
16.         name="password" class="form-control"
17.         placeholder="Password" required>
18.   </p>
19.   <input name="_csrf" type="hidden"
20.       value="b65c8057-6296-44b9-af77-23158cccb80d"
21.       />
22.   <button class="btn btn-lg btn-primary btn-block"
23.       type="submit">Sign in</button>
24. </form>
```

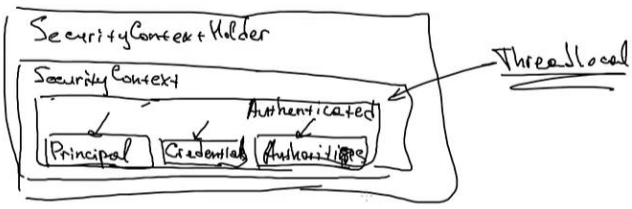
Выделенные атрибуты name не стоит изменять. Их считывает Spring Security.

Для входа без БД можно воспользоваться логином user и паролем, сгенерированным в консоли.

Объект Principal можно заключить в параметры метода контроллера и получить информацию о пользователе:

```
1. @GetMapping("/authenticated")
2. public String pageForAuthenticatedUser(
3.     Principal principal) {
```

```
4.    return principal.getName();
5.  }
```



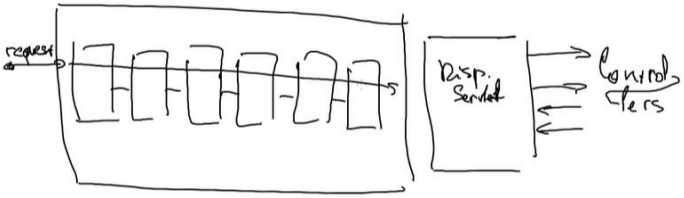
SecurityContextHolder — основное хранилище.

SecurityContext — хранилище данных, хранит данные в ThreadLocal переменной (для каждого потока свои данные)

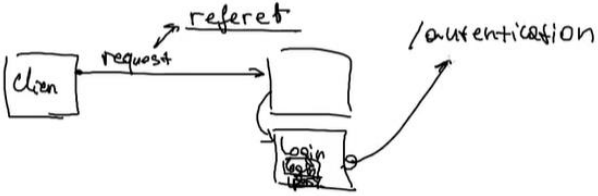
Authenticated: Principal (информация о пользователе), Credentials (пароль, который нужно проверить), Authorities (права доступа).

Credentials чистится сразу после проверки пароля. Principal не хранит в себе пароля. Сделано в целях безопасности.

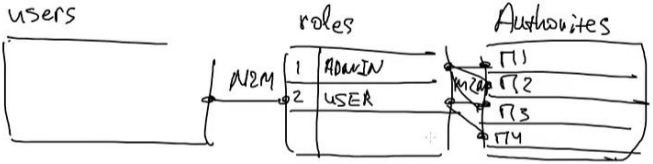
Данные храниться во ThreadLocal переменной тоже в целях безопасности. Пользователь в своем потоке работает, и только о себе информацию знает.



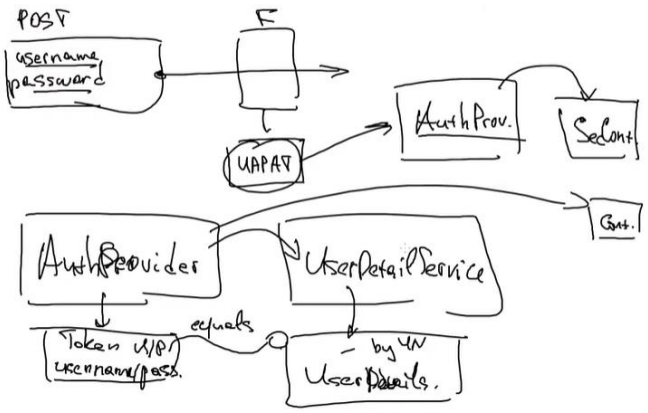
Процесс аутентификация происходит до диспетчера сервлета и обрабатывается множеством фильтров.



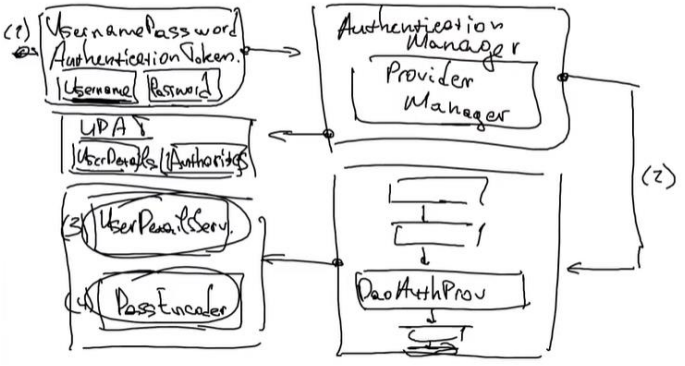
Asd



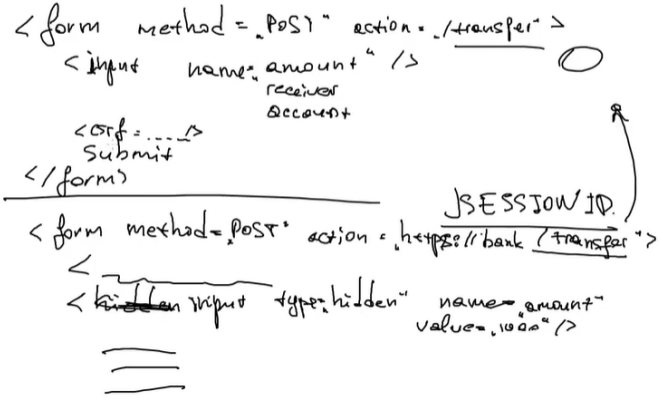
asd


POST
F
username
password
UPAFT
AuthProv.
SecCont.
Cont.
AuthProvider
UserDetailService
equals
by UN

Asd


(1)
UsernamePassword
AuthenticationToken.
Username
Password
UPAT
UserDetails
Authorities
AuthenticationManager
ProviderManager
(2)
(3)
UserDetailServ.
(4)
PassEncoder
DaoAuthProv

Asdf


<form method="POST" action="/transfer">
<input name="amount" />
receiver account
<csrf=...>
Submit
</form>
JSESSIONID
<form method="POST" action="https://bank/transfer">
<input type="hidden" name="amount" value="1000" />

увфы