











# Spring Security ke Important Classes aur Interfaces (Simple Version)

---

## 1. UserDetails (Interface)

- Ye ek interface hai jo **user ki details** rakhta hai (username, password, status).
  - **Important Methods (  ):**
    -  getUsername() → User ka naam deta hai.
    -  getPassword() → User ka password deta hai.
    -  getAuthorities() → User ke roles/permissions deta hai.
    -  isAccountNonExpired() → Account expire hua ya nahi.
    -  isAccountNonLocked() → Account lock hai ya nahi.
    -  isCredentialsNonExpired() → Password expire hua ya nahi.
    -  isEnabled() → User active hai ya nahi.
- 

## 2. User (Class)

- Ye **UserDetails interface** ki ready-made implementation hai.
  - Aapko khud methods likhne ki zarurat nahi, sab methods already isme implement hote hain.
- 




## 3. UserDetailsService (Interface)

- Ye **username ke base par user ko load karta hai** (Database/LDAP se).
  - **Important Method (  ):**
    -  loadUserByUsername(String username) → User ki details laata hai.
-

#### 4. UsernameNotFoundException (Class)

- Jab user nahi milta toh ye exception throw hota hai.
- 










#### 5. PasswordEncoder (Interface)

- Ye password ko **encode** aur **match** karne ke liye use hota hai.
  - **Important Methods (  ):**
    -  `encode(rawPassword)` → Plain text password ko secure form me convert karta hai.
    -  `matches(rawPassword, encodedPassword)` → Password verify karta hai.
- 

#### 6. BCryptPasswordEncoder (Class)

- Ye **PasswordEncoder** ki implementation hai jo **BCrypt algorithm** use karti hai.
  - Isse password bahut secure tarike se hash hote hain.
- 

#### 7. HttpSecurity (Class)

- Ye class security ko **configure** karti hai (login, logout, CSRF, authorization rules).
- **Important Methods (  ):**
  -  `authorizeRequests()`
  -  `formLogin()`
  -  `logout()`
  -  `httpBasic()`
  -  `csrf()`
  -  `sessionManagement()`
  -  `rememberMe()`
  -  `exceptionHandling()`

---

## 8. GrantedAuthority (Interface)

- Ye ek permission/role ko represent karta hai.
- **Method (  ):**
  -  `getAuthority()`

---

## 9. UsernamePasswordAuthenticationToken (Class)

- Ye ek **object hai jo authentication attempt ko represent karta hai** (username + password).
- **Important (  ):**
  -  `getPrincipal()`
  -  `getCredentials()`
  -  `getAuthorities()`




---

## 10. AuthenticationManager (Interface)

- Ye **authentication process chalata hai**.
- **Method (  ):**
  -  `authenticate(Authentication authentication)`


---

## 11. Authentication (Interface)


- Ye ek **authenticated user ko represent karta hai**.
- **Methods (  ):**
  -  `getPrincipal()`
  -  `getCredentials()`

- `getAuthorities()`
  - `getDetails()`
- 

## 12. AuthenticationProvider (Interface)

- Ye ek custom way deta hai authentication karne ka.
  - **Method (  ):**
    - `authenticate(Authentication authentication)`
- 

## 13. SecurityFilterChain (Class)

- Ye **Spring Security ka filter chain** represent karta hai.
  - Isme multiple filters hote hain jo authentication/authorization handle karte hain.
  - **Methods (  ):**
    - `matches()`
    - `getFilters()`
    - `addFilter()`
    - `addFilterBefore()`
    - `addFilterAfter()`
- 

## 14. Principal (Interface)

- Ye **currently logged-in user** ko represent karta hai.
  - **Method (  ):**
    - `getName()`
-

👉 Is tarah Classes ( ) , Interfaces ( ) aur Methods ( ) alag color me rakhe gaye hain.

---

Kya aap chahte ho main isko **HTML file** bana kar dikhau (taaki aap STS ya browser me open karke colored output directly dekh sako)?