**Core Spring Security Component**

**UserDetails**

The UserDetails interface represents a user in the Spring Security framework.

It provides methods to get user information such as username, password, and authorities. **Purpose:** To encapsulate user information, including authentication and authorization details. **Implementation:** You can use it to extend your User Entity.

## **UserDetailsService**

The UserDetailsService interface is a core component in Spring Security that is used to retrieve user-related data. It has a single method: loadUserByUsername

**Purpose:** To fetch user details from a datasource (e.g., database) based on the username.

**Implementation:** You typically implement this interface to load user details, such as username, password, and roles, from your own user repository.

## **InMemoryUserDetailsManager**

The InMemoryUserDetailsManager is a Spring Security provided implementation of UserDetailsService that stores user information in memory.

**Purpose:** To store user details in memory, typically for testing or small applications. You define users directly in the configuration.

**PasswordEncoder**

The PasswordEncoder interface is used for encoding and validating passwords. It has methods for encoding raw passwords and matching encoded passwords.

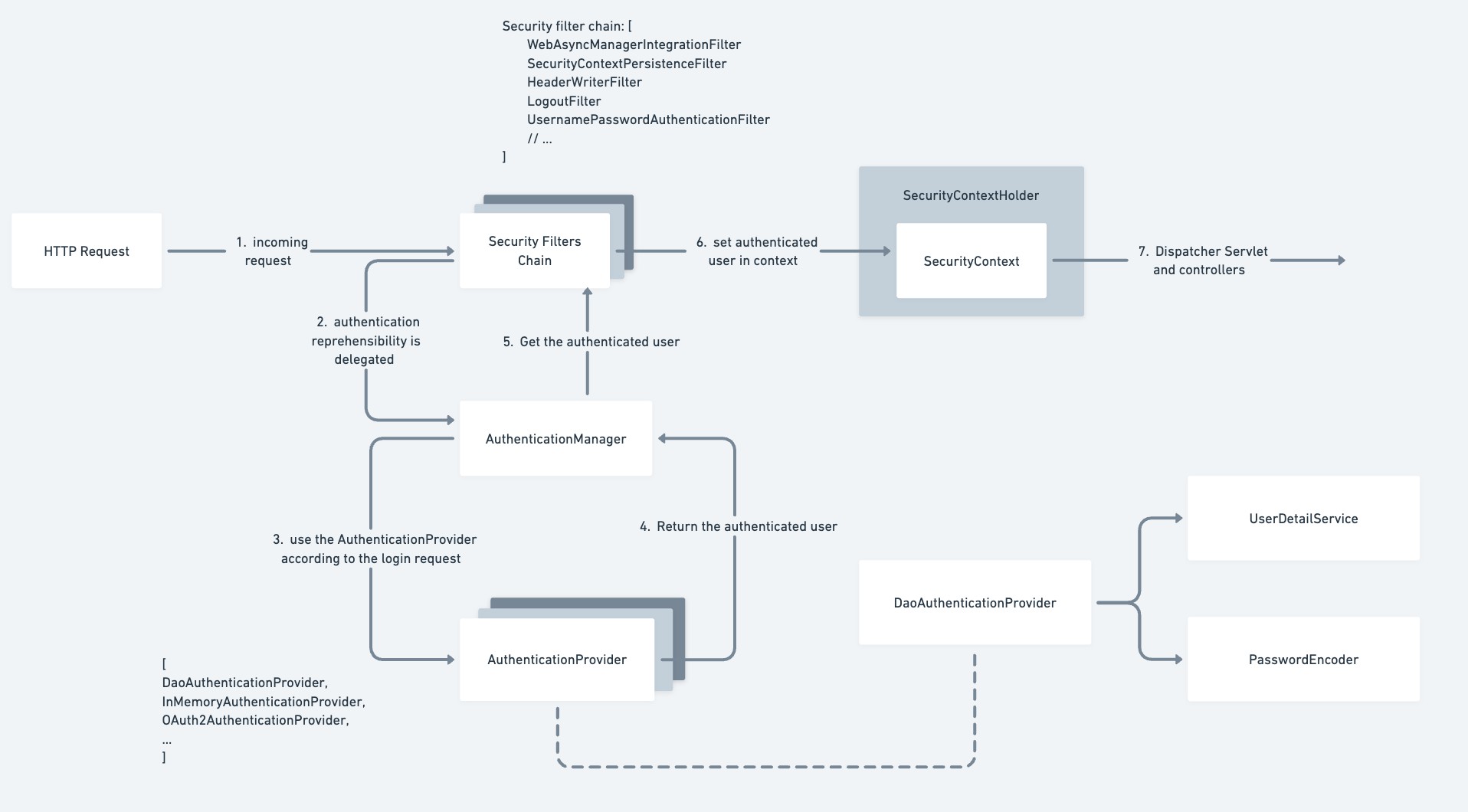
**Purpose:** To securely hash passwords before storing them and to verify hashed passwords during authentication.

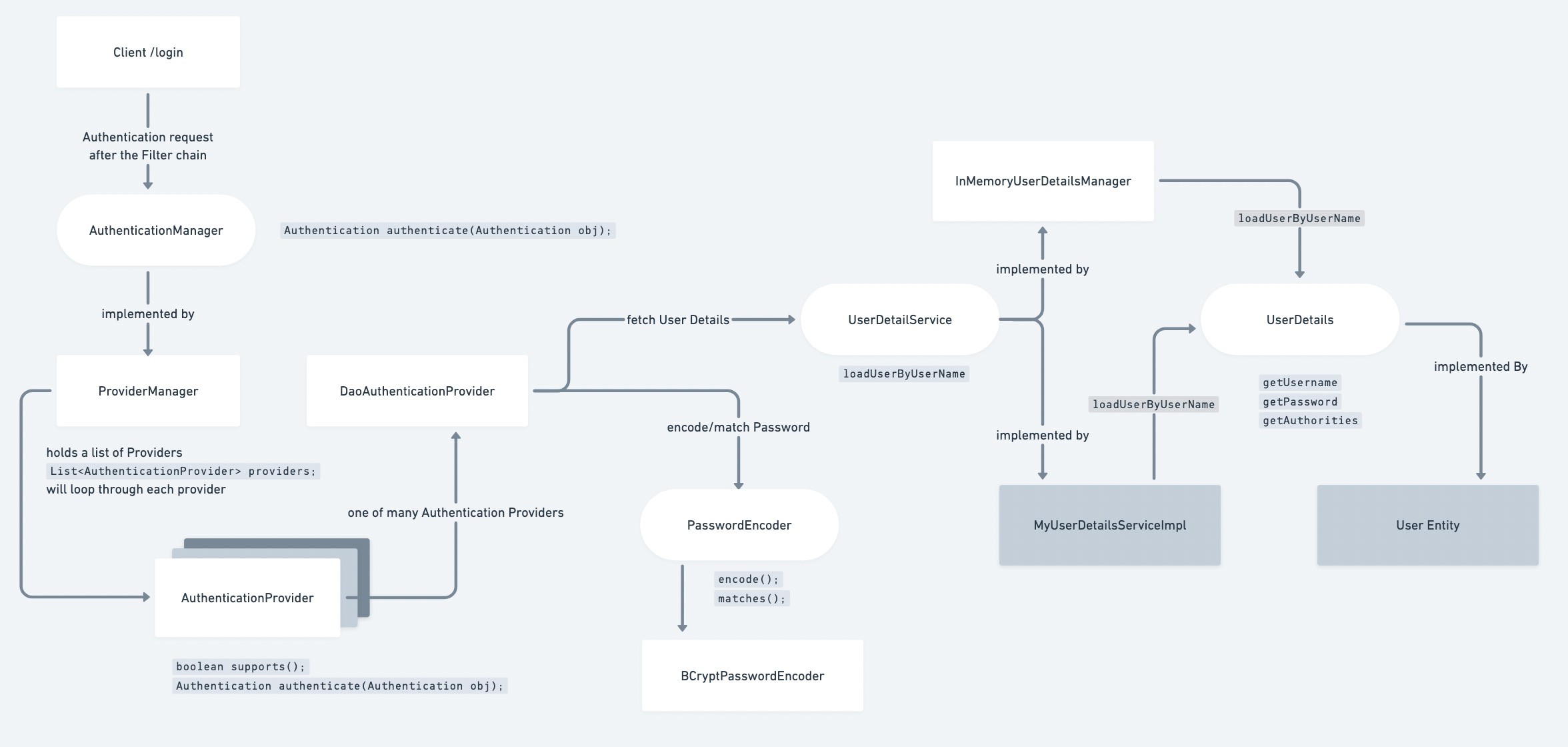
**Common Implementations:**

BCryptPasswordEncoder

Pbkdf2PasswordEncoder

SCryptPasswordEncoder





**Configuring SecurityFilterChain:**

Default SecurityFilterChain Config

• **authorizeRequests()** restricts access based on RequestMatcher implementations.

• **authenticated()** requires that all endpoints called be authenticated before proceeding in the filter chain.

• **formLogin()** calls the default FormLoginConfigurer class that loads the login page to authenticate via username-password and accordingly redirects to corresponding failure or success handlers. • **csrf()** to cofigure the csrf protection