Spring Security Overview

By Ramesh Fadatare (Java Guides)

Spring Security

- Spring Security is a framework that provides authentication, authorization, and protection against common attacks. With first class support for securing both imperative and reactive applications, it is the de-facto standard for securing Spring-based applications.

- Spring Boot provides a **spring-boot-starter-security** starter that aggregates Spring Security-related dependencies together.

Blog Comments Management Feature

1. Create Comment JPA Entity
2. Create CommentRepository
3. Create CommentDto
4. Create CommentMapper
5. Create Comment Form Handling
6. Create Handler Method to Save Comment
7. Adding Validation to Create Comment Form
8. Display List of Comments for Blog Post

User Registration Feature

1. User and Role JPA Entities (Many to Many Mapping)
2. Spring Data JPA Repositories for User and Role
3. Create Thymeleaf Template and Design Registration Form
4. Save Registration Form Data into MySQL database
5. Add Validation to Registration Form

Login Feature using Spring Security

1. Add Spring Security to Spring Boot Application
2. Spring Boot Auto Configuration for Spring Security
3. Spring Security's default Login and Logout Features
4. Create Custom Login Form Step by Step
5. Implement Database Authentication

Spring Boot Auto Configuration for Spring Security

Spring Boot automatically:

- Enables Spring Security's default configuration, which creates a servlet Filter as a bean named **springSecurityFilterChain**. This bean is responsible for all the security (protecting the application URLs, validating submitted username and passwords, redirecting to the login form, and so on) within your application.
- Creates a **UserDetailsService** bean with a username of user and a randomly generated password that is logged to the console.
- If Spring Security is on the classpath, Spring Boot automatically secures all HTTP endpoints with "basic" authentication.

Spring Boot Auto Configuration for Spring Security

Spring Boot auto configures below features:

- **spring-boot-starter-security** starter that aggregates Spring Security-related dependencies together.
- Enables Spring Security's default configuration, which creates a servlet Filter as a bean named **springSecurityFilterChain**. Provides default login form for you.
- Creates default user with a username as **user** and a randomly generated password that is logged to the console (Ex: 8e557245-73e2-4286-969a-ff57fe326336).
- Spring boot provides properties to cusomize default user's username and password
- Protects the password storage with **BCrypt** algorithm
- Lets the user log out (default logout feature)
- CSRF attack prevention (enabled by default)
- If Spring Security is on the classpath, Spring Boot automatically secures all HTTP endpoints with "basic" authentication.

How form based authentication works within Spring Security

Spring Security provides support for username and password being provided through an html form.

- Step 1: How user is redirected to the log in form.
- Step 2: When the username and password are submitted, the **UsernamePasswordAuthenticationFilter** authenticates the username and password.

Step 1: Redirect to Login Form

The below diagram shows how the user is redirected to the log in form.

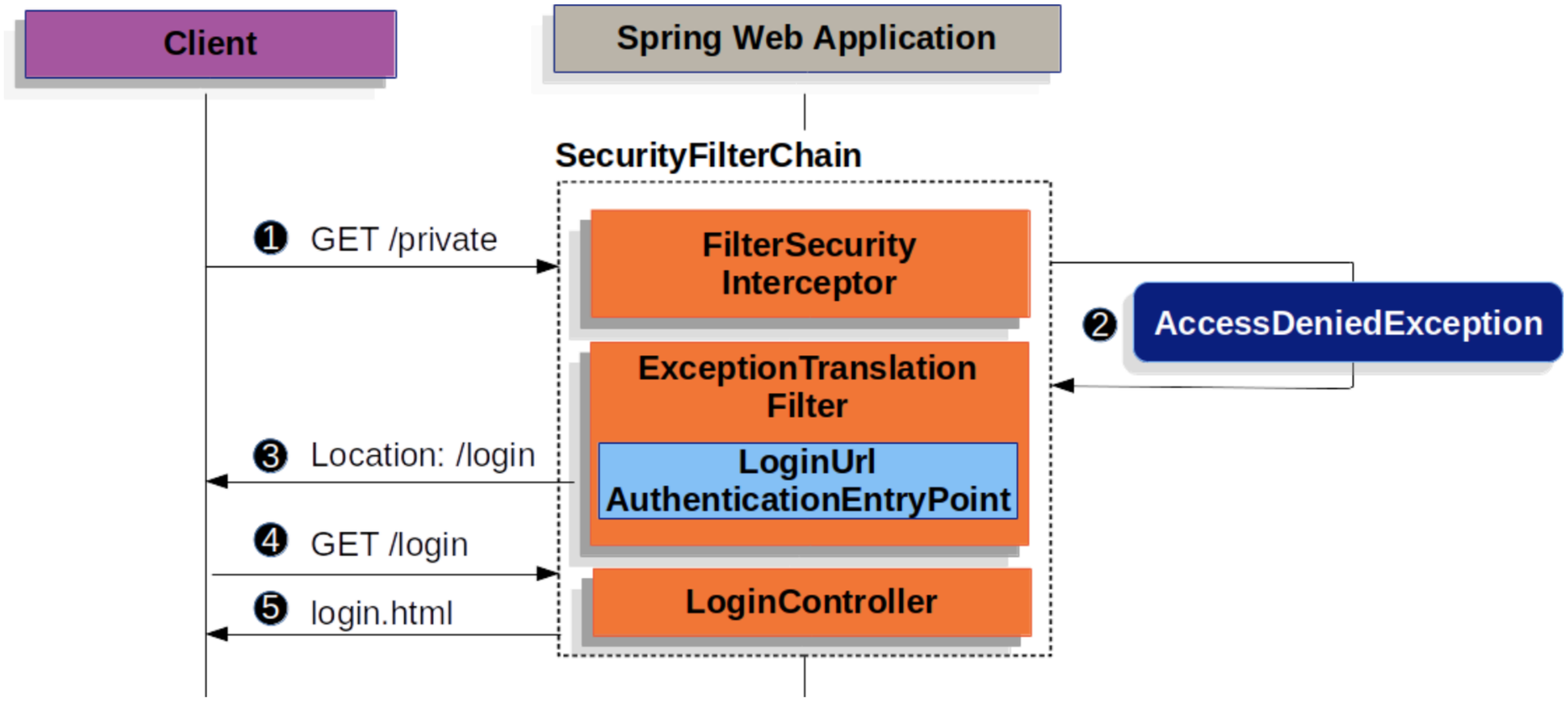

Figure 1. Redirecting to the Log In Page

Step 2: Submit the Form

- When the username and password are submitted, the **UsernamePasswordAuthenticationFilter** authenticates the username and password.

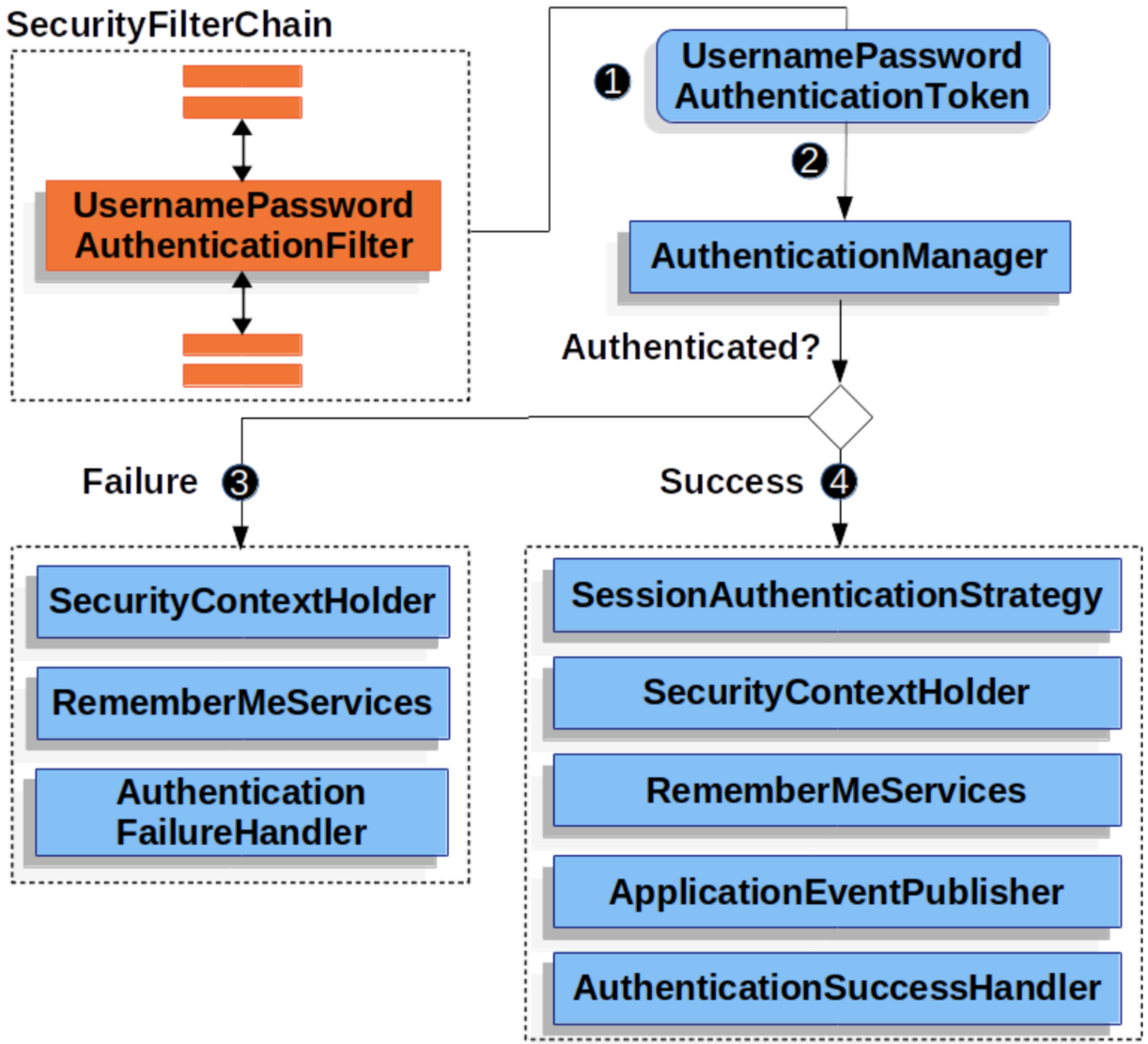

Figure 2. Authenticating Username and Password

Spring Security Default Form Login

- Spring Security form log in is enabled by default. However, as soon as any servlet based configuration is provided, form based log in must be explicitly provided. Spring Boot Auto Configure default configuration.

```
@Configuration
@EnableWebSecurity
public class WebSecurityConfig {

    public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {
        http
                .formLogin(withDefaults());
        return http.build();
        // ...
    }
}
```