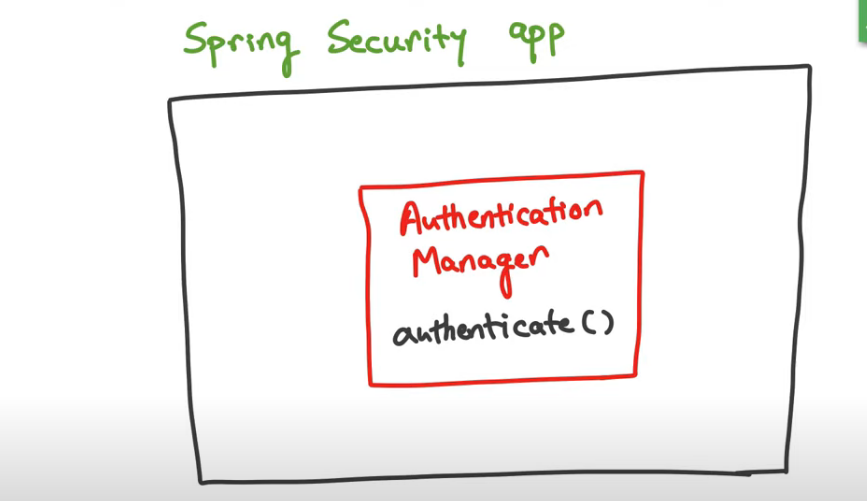
Configure Spring Security in your Spring Boot Application

The way to configure Spring security is by affecting what’s called **AuthenticationManager.** The Authentication Manager is true to this name something it manages authentication in a spring Application.



AuthenticationManager actually has a method called **authenticate** that either returns a successful authentication or throws an exception authentication that it cannot authenticate. We don’t work with authentication manager directly for most part what we do instade is work with a build class called AuthenticationManagerBuilder we use AuthenticationManagerBuilder to configure what the authentication actually do,. So there are two steps

Step 1: Get hold of **AuthenticationManagerBuilder**

Step 2: Set the configuration on it.

How to get hold of **AuthenticationManagerBuilder**?

In the first place, the way to do this is by leveraging a hook that’s already available in the spring security app. The thing is in a spring security app there is a class that’s sitting there which has a method call **configure** and takes in the argument the authentication manager builder and spring security framework calls that method and passes in the **AuthenticationManagerBuilder**. The reason that class is there is so that it gives developers the opportunity to extend the class and overrides the **configure** method to do the configuration that we want if we don’t override the method then the default configuration happens.

Image for Spring Authentication

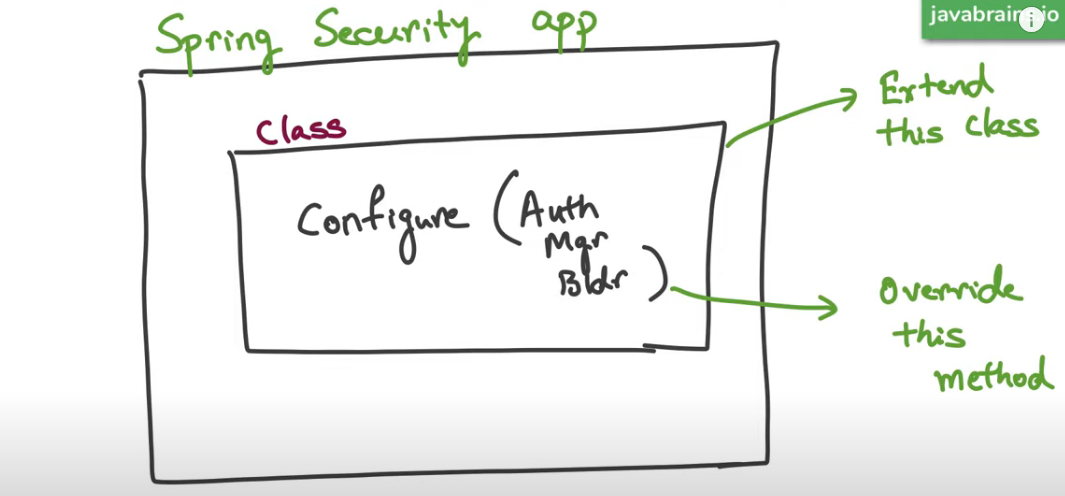
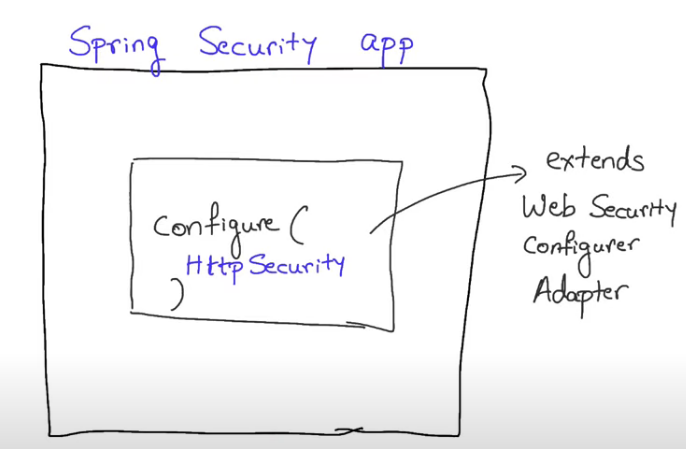
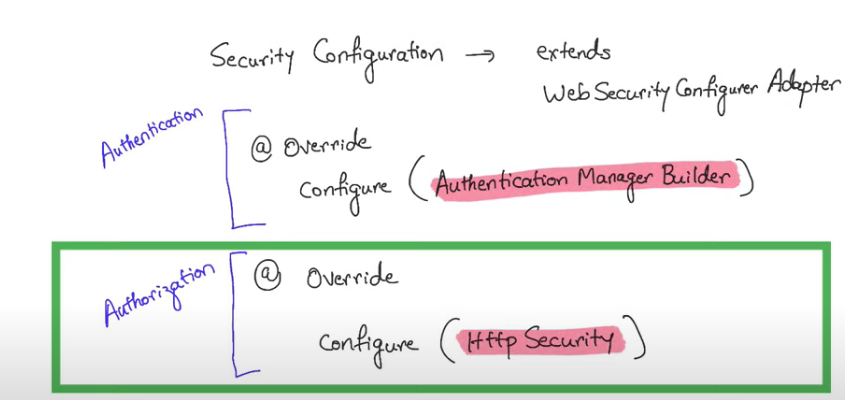


Image for Spring Authorization





Spring Security Authentication and Authorization example

**package** com.skg.springbootjwt.config;

**import** org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;

**import** org.springframework.security.config.annotation.web.builders.HttpSecurity;

**import** org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

**import** org.springframework.security.config.annotation.web.configuration.~~WebSecurityConfigurerAdapter~~;

@EnableWebSecurity

**public** **class** SpringSecurityConfig **extends** ~~WebSecurityConfigurerAdapter~~ {

@Override

**protected** **void** configure(AuthenticationManagerBuilder auth) **throws** Exception {

// Adding users inside in memory

auth.inMemoryAuthentication().withUser("khawja").password("{noop}password").roles("USER").a nd().withUser("khawja").password("{noop}password").roles("ADMIN");

}

@Override

**protected** **void** configure(HttpSecurity http) **throws** Exception {

http.authorizeRequests().antMatchers("/\*\*").hasRole("ADMIN").and().formLogin();

}

}

Simple REST controller

package com.skg.springbootjwt.controller;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class HomeController {

@GetMapping("/")

public String home() {

return "Home Page!!";

}

@GetMapping("/user")

public String user() {

return "Welcome User !!";

}

@GetMapping("/admin")

public String admin() {

return "Welcome Admin!!";

}

}