Spring Security -using Boot

In this lab, u will be working on 06-mvc-security-boot-start project

1) Open pom.xml and observe that we have already configured spring-boot-starter-security

2)Open Way2LearnApplication.java and observe that we have used @EnableAutoConfiguration

3) Open SecurityInitializer.java and SecurityConfig.java and observe that we have already configured Spring security using java configuration

4) Deploy the application and observe that deployment fails because of duplicate filter configuration.

Delete SecurityInitializer.java which is configuring DelegatingFilterProxy.

Now deploy and observe that it is deployed.

Now login using the credentials configured in SecurityConfig.java

5) Since we have enabled auto configuration, SecurityAutoConfiguration will configure all the defaults.

Open AuthenticationManagerConfiguration by pressing ctrl+shift+T. Observe that `DefaultInMemoryUserDetailsManagerConfigurer` is applied. It Configures a user with name "user" and randomly generated password.

We want to work with defaults. So, comment all the methods in SecurityConfig.java

By default, Basic Authentication is configured and all urls require that user must be authenticated.

Now deploy the application and observe that browser prompts you for user name and password.

Copy the default password printed on console and login with user name "user" and the copied password

6) We dont want default user name and password.

So, configure following in application.properties

```
security.user.name=siva
security.user.password=secret
security.user.role=ADMIN,USER
```

Now Deploy the application and login with given username and password

7) We want configure more users. So, now Open SecurityConfig.java and write the below method :

```
	@Autowired
	protected void configureGlobal(AuthenticationManagerBuilder auth) throws Exception {
		auth.inMemoryAuthentication()
			.withUser("siva").password("secret").roles("ADMIN")
			.and()
			.withUser("prasad").password("secret").roles("USER");
	}
```

8) Also, we want to use form login and map urls to different roles,etc. So, override the following method inside SecurityConfig

```
	protected void configure(HttpSecurity http) throws Exception {
		http.authorizeRequests()
				//.antMatchers("/login**").permitAll()
				.antMatchers("/x/**").hasRole("ADMIN")
				.anyRequest().authenticated()
				.and()
				.formLogin();


	}
```

Congratulations !! U have configured spring security using Boot