Now I have implemented the basic structure for the 3 microservices. In Registration microservice a basic Get and Post rest service is implemented as well.

Now I should implement login service.

I will use spring security, OAuth and JWT for authentication, authorization and exception handling.

Note: Do not do the mistake of putting Boot Application java class in package other than the maven or gradle group name.

In config(HttpSecurity http) method where you extend WebSecurityConfigrerAdapator if you do not want default spring’s login page the do not add .formLogin() in the http.. chain of antMatchers().

Hey “Optional” in java 8 is basically to avoid NullPointerException. It is just a wrapper around the class it refers. With helper methods and support of functional programming abilities of java8.

Note about @Autowired and @Bean

By @Bean we tell the spring to manage this class so that whenever I ask for it give it to me.

@Autowired is asking for a managed class or bean.

Struggled to get Authorization part of spring working.

Steps

1: extend for the @Configuration class were we have @WebsecurityConfigurer with WebSecurityConfigurerAdaptar.

2: here in this class we can use the overloaded config method for HttpSecurity as argument were we give antMatchers.

3: to tell spring what roles are available for a user we use argument in one of the config methods AuthenticationManagerBuilder and we call our Custom UserService which has extends Springs provided UserService. There the important method is loadByUserName, in here we actually return the particular user.

4: important note is the overridden getAuthorities method need to be provded with the actuall Role text appended with “Role\_”