

```

1 InMemory Authentication
2 =====pom.xml=====
3 <?xml version="1.0" encoding="UTF-8"?>
4 <project xmlns="http://maven.apache.org/POM/4.0.0"
5   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
7       https://maven.apache.org/xsd/maven-4.0.0.xsd">
8   <modelVersion>4.0.0</modelVersion>
9   <parent>
10     <groupId>org.springframework.boot</groupId>
11     <artifactId>spring-boot-starter-parent</artifactId>
12     <version>2.3.0.RELEASE</version>
13     <relativePath/> <!-- lookup parent from repository -->
14   </parent>
15   <groupId>com.ameya</groupId>
16   <artifactId>006-SpringSecurityBasicInMemory</artifactId>
17   <version>0.0.1-SNAPSHOT</version>
18   <name>006-SpringSecurityBasicInMemory</name>
19   <description>Demo project for Spring Boot</description>
20   <properties>
21     <java.version>1.8</java.version>
22   </properties>
23   <dependencies>
24     <dependency>
25       <groupId>org.springframework.boot</groupId>
26       <artifactId>spring-boot-starter-security</artifactId>
27     </dependency>
28     <dependency>
29       <groupId>org.springframework.boot</groupId>
30       <artifactId>spring-boot-starter-web</artifactId>
31     </dependency>
32     <dependency>
33       <groupId>org.springframework.boot</groupId>
34       <artifactId>spring-boot-starter-test</artifactId>
35       <scope>test</scope>
36       <exclusions>
37         <exclusion>
38           <groupId>org.junit.vintage</groupId>
39           <artifactId>junit-vintage-engine</artifactId>
40         </exclusion>
41       </exclusions>
42     </dependency>
43     <dependency>
44       <groupId>org.springframework.security</groupId>
45       <artifactId>spring-security-test</artifactId>
46       <scope>test</scope>
47     </dependency>
48   </dependencies>
49   <build>
50     <plugins>
51       <plugin>
52         <groupId>org.springframework.boot</groupId>
53         <artifactId>spring-boot-maven-plugin</artifactId>
54       </plugin>
55     </plugins>
56   </build>
57 </project>
58 =====application.properties=====
59
60 server.port=9001
61 logging.level.web=trace
62
63 =====AppSecurityConfig.java=====
64 package com.ameya.configs;
65
66 import org.springframework.context.annotation.Configuration;
67 import
68 org.springframework.security.config.annotation.authentication.builders.AuthenticationM
anagerBuilder;
69 import org.springframework.security.config.annotation.web.builders.HttpSecurity;

```

```

70 import
org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
71 import
org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurer
Adapter;
72
73 @Configuration
74 @EnableWebSecurity
75 public class AppSecurityConfig extends WebSecurityConfigurerAdapter {
76
77     @Override
78     protected void configure(AuthenticationManagerBuilder auth) throws Exception {
79         auth.inMemoryAuthentication()
80             .withUser("ameya").password("{noop}1234").roles("ADMIN")
81             .and()
82             .withUser("avani").password("{noop}1234").roles("USER");
83     }
84
85     @Override
86     protected void configure(HttpSecurity http) throws Exception {
87         http.authorizeRequests()
88             .antMatchers("/").permitAll()
89             .antMatchers("/rest/hello*").access("hasRole('ROLE_ADMIN')")
90             .and().httpBasic();
91         http.csrf().disable();
92     }
93
94
95 }
96 =====HelloController=====
97 package com.ameya.controllers;
98
99 import org.springframework.security.core.context.SecurityContextHolder;
100 import org.springframework.security.core.userdetails.UserDetails;
101 import org.springframework.web.bind.annotation.GetMapping;
102 import org.springframework.web.bind.annotation.RestController;
103
104 @RestController
105 public class HelloController {
106
107     @GetMapping(path="/")
108     public String rootUrl() {
109         return "This is permitted to ALL !!";
110     }
111     @GetMapping("/rest/hello")
112     public String sayHello() {
113         String userName=getUserName();
114         return "Hi There - This is permitted only to Role Admin :: "+userName;
115     }
116     public String getUserName() {
117         String userName=null;
118         Object
119         principal=SecurityContextHolder.getContext().getAuthentication().getPrincipal(
120         );
121         userName=((UserDetails)principal).getUsername();
122         return userName;
123     }
124 }
125
126 =====
127 JdbcAuthentication
128 =====
129 <?xml version="1.0" encoding="UTF-8"?>
130 <project xmlns="http://maven.apache.org/POM/4.0.0"
131 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
132 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
133 https://maven.apache.org/xsd/maven-4.0.0.xsd">
134 <modelVersion>4.0.0</modelVersion>
135 <parent>
136     <groupId>org.springframework.boot</groupId>
137     <artifactId>spring-boot-starter-parent</artifactId>
138     <version>2.3.0.RELEASE</version>
139     <relativePath/> <!-- lookup parent from repository -->

```

```

136     </parent>
137     <groupId>com.ameya</groupId>
138     <artifactId>006-SpringSecurityBasicJdbc</artifactId>
139     <version>0.0.1-SNAPSHOT</version>
140     <name>006-SpringSecurityBasicJdbc</name>
141     <description>Demo project for Spring Boot</description>
142     <properties>
143         <java.version>1.8</java.version>
144     </properties>
145     <dependencies>
146         <dependency>
147             <groupId>org.springframework.boot</groupId>
148             <artifactId>spring-boot-starter-jdbc</artifactId>
149         </dependency>
150         <dependency>
151             <groupId>org.springframework.boot</groupId>
152             <artifactId>spring-boot-starter-security</artifactId>
153         </dependency>
154         <dependency>
155             <groupId>org.springframework.boot</groupId>
156             <artifactId>spring-boot-starter-web</artifactId>
157         </dependency>
158
159         <dependency>
160             <groupId>mysql</groupId>
161             <artifactId>mysql-connector-java</artifactId>
162             <scope>runtime</scope>
163         </dependency>
164         <dependency>
165             <groupId>org.springframework.boot</groupId>
166             <artifactId>spring-boot-starter-test</artifactId>
167             <scope>test</scope>
168             <exclusions>
169                 <exclusion>
170                     <groupId>org.junit.vintage</groupId>
171                     <artifactId>junit-vintage-engine</artifactId>
172                 </exclusion>
173             </exclusions>
174         </dependency>
175         <dependency>
176             <groupId>org.springframework.security</groupId>
177             <artifactId>spring-security-test</artifactId>
178             <scope>test</scope>
179         </dependency>
180     </dependencies>
181
182     <build>
183         <plugins>
184             <plugin>
185                 <groupId>org.springframework.boot</groupId>
186                 <artifactId>spring-boot-maven-plugin</artifactId>
187             </plugin>
188         </plugins>
189     </build>
190
191 </project>
192 =====application.properties=====
193
194 server.port=9001
195 logging.level.web=trace
196
197 spring.datasource.url=jdbc:mysql://localhost:3306/nineleapsdb
198 spring.datasource.username=root
199 spring.datasource.password=root
200 spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
201 #spring.datasource.initialization-mode=always
202 =====schema.sql=====
203
204 CREATE TABLE users(username VARCHAR(45) NOT NULL,password VARCHAR(200) NOT
205 NULL,enabled TINYINT(4) NOT NULL DEFAULT '1',PRIMARY KEY (username));
206
207 CREATE TABLE user_roles (user_role_id INT(11) NOT NULL AUTO_INCREMENT,username

```

```

VARCHAR(45) NOT NULL,role VARCHAR(45) NOT NULL,PRIMARY KEY (user_role_id),UNIQUE
INDEX uni_username_role (role, username),INDEX fk_username_idx (username),CONSTRAINT
fk_username FOREIGN KEY (username) REFERENCES users (username));
=====data.sql=====
=====
207 INSERT INTO users (username, password, enabled) VALUES('ameya', '1234', 1);
208 INSERT INTO users (username, password, enabled) VALUES('avani', '1234', 1);
209
210 INSERT INTO user_roles (user_role_id, username, role) VALUES(2, 'ameya',
'ROLE_ADMIN');
211 INSERT INTO user_roles (user_role_id, username, role) VALUES(1, 'ameya', 'ROLE_USER');
212 INSERT INTO user_roles (user_role_id, username, role) VALUES(3, 'avani', 'ROLE_USER');
213 =====SecurityConfiguration.java=====
=====
214 package com.ameya.configs;
215
216 import javax.sql.DataSource;
217
218 import org.springframework.beans.factory.annotation.Autowired;
219 import org.springframework.context.annotation.Bean;
220 import org.springframework.context.annotation.Configuration;
221 import
org.springframework.security.config.annotation.authentication.builders.AuthenticationM
anagerBuilder;
222 import org.springframework.security.config.annotation.web.builders.HttpSecurity;
223 import
org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
224 import
org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurer
Adapter;
225 import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
226 import org.springframework.security.crypto.password.PasswordEncoder;
227
228 @Configuration
229 @EnableWebSecurity
230 public class SecurityConfiguration extends WebSecurityConfigurerAdapter{
231
232     @Autowired
233     private DataSource dataSource;
234     @Bean
235     public PasswordEncoder passwordEncoder() {
236         return new BCryptPasswordEncoder();
237     }
238     @Override
239     protected void configure(AuthenticationManagerBuilder auth) throws Exception {
240         auth.jdbcAuthentication()
241             .usersByUsernameQuery("select username,password,enabled from users where
username=?")
242             .authoritiesByUsernameQuery("select username,role from user_roles where
username=?")
243             .passwordEncoder(passwordEncoder()).dataSource(dataSource);
244
245     }
246     @Override
247     protected void configure(HttpSecurity http) throws Exception {
248         http
249             .authorizeRequests()
250             .antMatchers("/").permitAll()
251             .antMatchers("/rest/hello*")
252             .access("hasRole('ROLE_ADMIN')")
253             .and().httpBasic();
254         http.csrf().disable();
255
256     }
257
258 }
259 =====HelloController.java=====
260 package com.ameya.controllers;
261
262 import org.springframework.security.core.context.SecurityContextHolder;
263 import org.springframework.security.core.userdetails.UserDetails;
264 import org.springframework.web.bind.annotation.GetMapping;
265 import org.springframework.web.bind.annotation.RestController;

```

```

266
267 @RestController
268 public class HelloController {
269
270     @GetMapping(path="/")
271     public String rootUrl() {
272         return "This is permitted to ALL !!";
273     }
274     @GetMapping("/rest/hello")
275     public String sayHello() {
276         String userName=getUserName();
277         return "Hi There - This is permitted only to Role Admin :: "+userName;
278     }
279     public String getUserName() {
280         String userName=null;
281         Object
            principal=SecurityContextHolder.getContext().getAuthentication().getPrincipal(
                );
            userName=((UserDetails)principal).getUsername();
            return userName;
284     }
285 }
286 =====PasswordEncoder.java=====
287 package com.ameya;
288
289 import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
290
291 public class PasswordEncoder {
292
293     public static void main(String[] args) {
294         String encoded=new BCryptPasswordEncoder().encode("1234");
295         System.out.println(encoded);
296     }
297
298 }
299
300

```