Exercise 21.1 Spring Security

The Setup:

Create a copy of exercise19_1, and call it exercise20_1. Remember to also change the project name inside the pom.xml's `<artifactId>` and `<name>` tags.

We will be using Spring Security 4.0 which is still being tested for release. Although using a pre-release version is a bit of a pain, it's easier than using Spring Security 3 with SpringMVC 4.

Start by adding the following repository to your pom.xml file (right after the `<dependencies>` section) so that you have the possibility of downloading spring-security pre-releases:

```
<repositories>
    <repository>
        <id>spring-snapshots</id>
        <name>Spring Snapshots</name>
        <url>http://repo.spring.io/snapshot</url>
        <snapshots>
            <enabled>true</enabled>
        </snapshots>
    </repository>
</repositories>
```

Add the SpringMVC dependencies outlined in exercise 19_1, and then also add the following additional dependencies for spring security:

```
    <dependency>
        <groupId>org.springframework.security</groupId>
        <artifactId>spring-security-web</artifactId>
        <version>4.0.0.CI-SNAPSHOT</version>
    </dependency>
    <dependency>
        <groupId>org.springframework.security</groupId>
        <artifactId>spring-security-taglibs</artifactId>
        <version>4.0.0.CI-SNAPSHOT</version>
    </dependency>
    <dependency>
        <groupId>org.springframework.security</groupId>
        <artifactId>spring-security-config</artifactId>
        <version>4.0.0.CI-SNAPSHOT</version>
    </dependency>
    <!-- security also wants the following to be present -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-tx</artifactId>
        <version>4.1.0.BUILD-SNAPSHOT</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-jdbc</artifactId>
        <version>4.1.0.BUILD-SNAPSHOT</version>
```

```
</dependency>
```

The Application:

The application is the same Car application as used in exercise 19.1, running it should still give the same output as before (see screenshot in the previous exercise).

I will also provide an example spring security project, which was used to create the slides.

The Exercise:

Make it so that the car application from the Spring MVC exercise has security. You should need at least USER authorization to see the list of cars, and ADMIN authorization should be required to add/delete/update the list.

The most important steps are:

1. Adding the context loader listener and spring security filter to the web.xml
2. Configuring a plain text authentication manager (in the root springconfig.xml)
3. Creating two users one with ROLE_USER and one with ROLE_ADMIN
4. Using the `<url-intercept>` tags to specify which urls can be
    a. Accessed anonymously
    b. Accessed as a normal user
    c. Accessed as an administrator
5. Using `<form-login>` to specify a login-page, authentication-failure-url, and make sure to set the default-target-url=/cars
6. Seting `<logout logout-success-url="login.jsp" />`
7. Updating `carList.jsp` so that users don't even see the edit and add a car links
8. Adding a logout link to carList.jsp
9. Testing that your admin can log in and go to all the different pages
10. Testing that your user can log in and only visit the list of cars (other pages blocked)

Optional additional exercise:

- Switch to using a JDBC Authenticator instead of plain text
- Try with both the standard tables and the non-standard tables