Nome: _____ Data: ____/____/____

Projeto – Utilizando Spring Security (Loja Virtual)

1. Adicionar ao arquivo **pom.xml** as seguintes dependências:

```
<!-- Spring Security -->
<dependency>
	<groupId>org.springframework.security</groupId>
	<artifactId>spring-security-web</artifactId>
	<version>3.2.6.RELEASE</version>
</dependency>
<dependency>
	<groupId>org.springframework.security</groupId>
	<artifactId>spring-security-config</artifactId>
	<version>3.2.6.RELEASE</version>
</dependency>
<dependency>
	<groupId>org.springframework</groupId>
	<artifactId>spring-jdbc</artifactId>
	<version>3.2.6.RELEASE</version>
</dependency>
<dependency>
	<groupId>commons-logging</groupId>
	<artifactId>commons-logging</artifactId>
	<version>1.1.1</version>
</dependency>
<dependency>
	<groupId>org.springframework.security</groupId>
	<artifactId>facelets-taglib-jsf20-spring-3</artifactId>
	<version>0.5</version>
</dependency>
```

Nome: _____ Data: ____/____/____

2. Adicionar também ao arquivo **pom.xml** os seguintes repositórios:

```
 <repositories>
<repository>
<id>jvnet-nexus-releases</id>
<name>jvnet-nexus-releases</name>
<url>https://maven.java.net/content/repositories/releases/</url>
</repository>
<repository>
<id>org.springframework.security.taglibs.facelets</id>
<url>http://spring-security-facelets-taglib.googlecode.com/svn/repo/</url>
        </repository>
 </repositories>
```

3. Alterar o projeto LojaVirtual, adicionando uma nova página chamada: login.xhml.

```
<?xml version="1.0" encoding="UTF-8"?>
<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:h="http://xmlns.jcp.org/jsf/html"
    xmlns:f="http://xmlns.jcp.org/jsf/core">
<h:head>
    <title>Login</title>
</h:head>
<h:body>
    <h:panelGroup rendered="#{!empty param.login_error}">
        Erro ao efetuar o login.<br />
        Motivo: #{SPRING_SECURITY_LAST_EXCEPTION.message}
    </h:panelGroup>

    <form id="login" method="post" action="#{request.contextPath}/j_spring_security_check">
    <table>
        <tr><td>Login</td>
            <td><input type='text' name='j_username'/></td></tr>
        <tr><td>Senha</td>
            <td><input type='password' name='j_password'/></td></tr>
        <tr><td align="right"><input type="checkbox" name="_spring_security_remember_me"/></td>
            <td>Logar automaticamente</td></tr>
        <tr><td></td>
            <td><input type="submit" value="Logar"/></td>
        </tr>
    </table>
    <script>
        document.getElementById("login").j_username.value = "#{SPRING_SECURITY_LAST_USERNAME}";
    </script>
    </form>
</h:body>
</html>
```

Nome: _____ Data: ____/____/____

4. Criar uma pasta dentro de **webapp** com o nome: **cliente**. A seguir, crie uma nova página xhtml com o nome: **Principal.xhtml**:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:ui="http://java.sun.com/jsf/facelets"
    xmlns:h="http://java.sun.com/jsf/html"
    xmlns:f="http://java.sun.com/jsf/core">

<h:head>
    <title>Área do Cliente</title>
</h:head>
<h:body>
    <h1><h:outputText value="Área - Cliente"/></h1>

</h:body>

</html>
```

5. Criar uma pasta dentro de **webapp** com o nome: **admin**. A seguir, crie uma nova página xhtml com o nome: **Principal.xhtml**:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:ui="http://java.sun.com/jsf/facelets"
    xmlns:h="http://java.sun.com/jsf/html"
    xmlns:f="http://java.sun.com/jsf/core">

<h:head>
    <title>Área Administrativa</title>
</h:head>
<h:body>
    <h1><h:outputText value="Área - Administrador"/></h1>
    <h:form>
        <lu>
            <li><h:commandLink value="Produto" action="/admin/lista_produto"/></li>
            <li><h:commandLink value="Cliente" action="/admin/lista_cliente"/></li>
            <li>
                <h:outputLink
                    onclick="document.location='#{request.contextPath}/j_spring_security_logout'">
                    Logout
                </h:outputLink>
            </li>
        </lu>
    </h:form>
</h:body>

</html>
```

Nome: _____ Data: ____/____/____

6. Altere a página **index.xhtml** para que fique com o seguinte conteúdo:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:ui="http://java.sun.com/jsf/facelets"
    xmlns:h="http://java.sun.com/jsf/html"
    xmlns:f="http://java.sun.com/jsf/core">

<h:head>
    <title>Exemplo</title>
</h:head>
<h:body>
    <h:form>
        <lu>
            <li><h:outputLink value="publico/form_cliente.xhtml">Cadastro de Cliente</h:outputLink></li>
            <li><h:outputLink value="cliente/principal.xhtml">Acessar Área do Cliente</h:outputLink></li>
            <li><h:outputLink value="admin/principal.xhtml">Acessar Área Administrativa</h:outputLink></li>
        </lu>
    </h:form>
</h:body>

</html>
```

7. Criar uma nova pasta com o nome: **META-INF**, dentro da pasta **webapp**.

8. Criar um novo arquivo xml dentro da pasta **META-INF** com o nome: **Context.xml** e com o seguinte conteúdo (Observe o nome do seu banco de dados):

```
<?xml version="1.0" encoding="UTF-8"?>
<Context reloadable="true">
    <Resource name="jdbc/hibernatedb"
              auth="Container"
              type="javax.sql.DataSource"
              maxActive="100"
              maxIdle="30"
              maxWait="10000"
              username="postgres"
              password="123"
              driverClassName="org.postgresql.Driver"
              url="jdbc:postgresql://localhost:5432/hibernatedb?autoReconnect=true"/>
</Context>
```

9. Criar dois arquivos de contexto (xml) necessários para o funcionamento do Spring Security. Estes arquivos deverão estar dentro da pasta WEB-INF. São eles:

Nome: _____ Data: ____/____/____

applicationContext.xml (Observe o nome do seu banco de dados)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
            http://www.springframework.org/schema/beans/spring-beans-3.2.xsd">
    <bean id="hibernateDbDataSource" class="org.springframework.jndi.JndiObjectFactoryBean">
        <property name="jndiName">
            <value>java:comp/env/jdbc/hibernatedb</value>
        </property>
    </bean>
</beans>
```

applicationContext-security.xml (Observe o nome do seu banco de dados)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/security"
    xmlns:beans="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
        http://www.springframework.org/schema/security
        http://www.springframework.org/schema/security/spring-security.xsd">
    <http auto-config="true">
        <intercept-url pattern="/admin/**" access="ROLE_ADMINISTRADOR" />
        <form-login login-page="/publico/login.xhtml"
            always-use-default-target="true"
            default-target-url="/admin/principal.xhtml"
            authentication-failure-url="/publico/login.xhtml?login_error=1" />
        <logout logout-success-url="/publico/index.xhtml"/>
        <remember-me />
    </http>
    <authentication-manager>
        <authentication-provider>
            <jdbc-user-service data-source-ref="hibernateDbDataSource"
                authorities-by-username-query="SELECT pes_nome, 'ROLE_ADMINISTRADOR'
                                                FROM pessoa
                                                WHERE pes_email = ?"
                users-by-username-query="SELECT pes_email, pes_senha, '1' FROM pessoa WHERE pes_email = ?" />
        </authentication-provider>
    </authentication-manager>
</beans:beans>
```

10. Alterar a forma de conexão com o BD configurada anteriormente no Hibernate, pois, tanto o Spring Security quanto o Hibernate utilizarão a mesma conexão com o Banco de Dados. Acessar o arquivo: **hibernate.cfg.xml**:

Adicionar a propriedade:

```xml
<property
      name="connection.datasource">java:/comp/env/jdbc/hibernatedb
</property>
```

Nome: _____ Data: ____/____/____

E **remover** as seguintes propriedades:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
        "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
    <session-factory>
        <!-- Configuração da conexão com o banco PostgreSQL e dialeto -->
        <property name="hibernate.dialect">org.hibernate.dialect.PostgreSQLDialect</property>
        <property name="hibernate.connection.driver_class">org.postgresql.Driver</property>
        <property name="hibernate.connection.username">postgres</property>
        <property name="hibernate.connection.password">123</property>
        <property name="hibernate.connection.url">jdbc:postgresql://localhost:5432/hibernatedb</property>

        <!-- Usando as configurações do C3PO para pool de conexões -->
        <property name="c3po.min_size">5</property>
        <property name="c3po.max_size">20</property>
        <property name="c3po.timeout">300</property>
        <property name="c3po.max_statements">50</property>
        <property name="c3po.idle_test_period">3000</property>
        <!-- Configurações de debug -->
        <property name="show_sql">true</property>
        <property name="format_sql">true</property>
        <property name="generate_statistics">true</property>
        <property name="use_sql_comments">true</property>

        <!-- Mapeamento das Classes -->
        <mapping class="beans.Produto"/>
    </session-factory>

</hibernate-configuration>
```

Retirar estas linhas

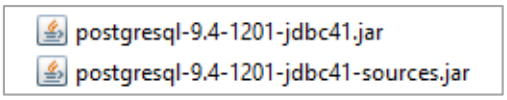
11. Adicionar ao arquivo **web.xml** a conexão com o banco de dados e os filtros de páginas que o Spring Security irá atuar.

```xml
<resource-ref>
    <description>DataSource hibernatedb</description>
    <res-ref-name>jdbc/hibernatedb</res-ref-name>
    <res-type>javax.sql.DataSource</res-type>
    <res-auth>Container</res-auth>
</resource-ref>


<!-- Spring Security -->
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>
        /WEB-INF/applicationContext.xml
        /WEB-INF/applicationContext-security.xml
    </param-value>
</context-param>
<filter>
    <filter-name>springSecurityFilterChain</filter-name>
    <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
</filter>
<filter-mapping>
    <filter-name>springSecurityFilterChain</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
<listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>
```

Nome: _____ Data: ____/____/____

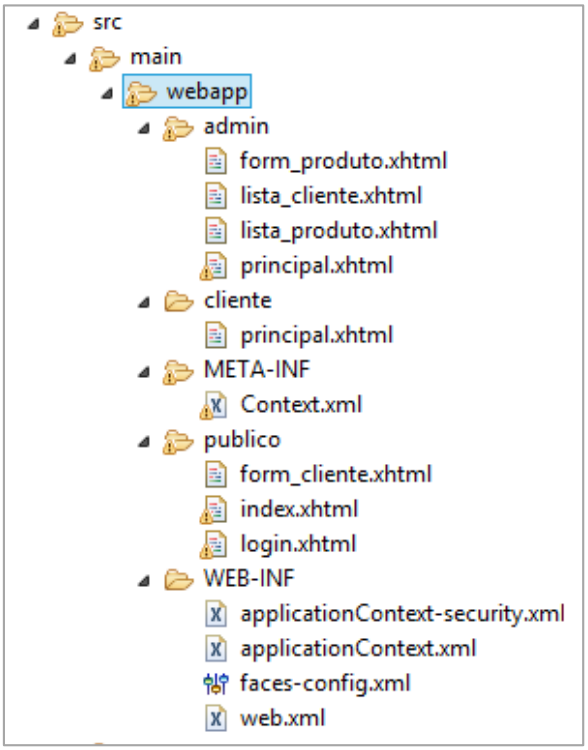
12. Copiar os arquivos .jars do postgresql para a pasta lib do servidor web, no nosso caso o Tomcat.



13. Altear o arquivo pom.xml, na dependência do postgreesql e adicionar a seguinte linha:

```
<!-- Postgresql -->
<dependency>
    <groupId>org.postgresql</groupId>
    <artifactId>postgresql</artifactId>
    <version>9.4-1201-jdbc41</version>
    <scope>provided</scope>
</dependency>
```

14. E para finalizar, verificar ou ajustar os diretórios e arquivos .xhtml para que fiquem com a seguinte estrutura.



15. Ao testar, verifique que o usuário poderá acessar o cadastro de clientes e a Área de Clientes (por enquanto) e ao acessar a Área Administrativa, o usuário será direcionado automaticamente para a página de login.