CONFIGURACIÓN DE NUESTRA APP CON SPRING SECURITY

REQUITOS:

Jar's:

- spring-security-cas-3.1.0-release.jar
- spring-security-config-3.0.1-release.jar
- spring-security-core-3.0.1-release.jar
- spring-security-taglibs-3.0.1-release.jar
- spring-security-web-3.0.5-release.jar
- spring-webmvc-3.0.0-release.jar

GUIAS:

- http://stackoverflow.com/questions/9834710/spring-security-and-cas-integration
- http://docs.spring.io/spring-security/site/docs/3.1.6.RELEASE/reference/cas.html
- https://wiki.jasig.org/display/CASC/Using+the+CAS+Client+3.1+with+Spring+Security
- https://wiki.jasig.org/display/CASC/Configuring+the+JA-SIG+CAS+Client+for+Java+using+Spring
- http://cuandotrasteo.blogspot.com/2010/09/spring-security-y-sso.html

DESARROLLO

1.- Para el desarrollo de la configuración de la nuestra aplicación con el uso de Spring Security se debe agregar las siguientes dependencias en el pom.xml

```
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-web</artifactId>
    <version>3.0.5.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-config</artifactId>
    <version>3.0.1.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-expression</artifactId>
    <version>3.0.3.RELEASE</version>
    <scope>runtime</scope>
</dependency>
```

```
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-webmvc</artifactId>
    <version>3.0.0.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-cas</artifactId>
    <version>3.1.0.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-core</artifactId>
    <version>3.0.1.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-taglibs</artifactId>
    <version>3.0.1.RELEASE</version>
</dependency>
```

2.- En el *web.xml* se agrega los siguientes filtros

```
<filter>
    <filter-name>CAS Authentication Filter</filter-name>
    <filter-class>org.jasig.cas.client.authentication.AuthenticationFilter</filter-class>
    <init-param>
        <param-name>casServerLoginUrl</param-name>
        <param-value>https://gianella09:8443/cas-server-webapp-3.5.2/login</param-value>
    </init-param>
    <init-param>
        <param-name>renew</param-name>
        <param-value>false</param-value>
    </init-param>
    <init-param>
        <param-name>gateway</param-name>
        <param-value>false</param-value>
    </init-param>
</filter>
```

3.- En el mismo *web.xml* se agrega las siguientes líneas.

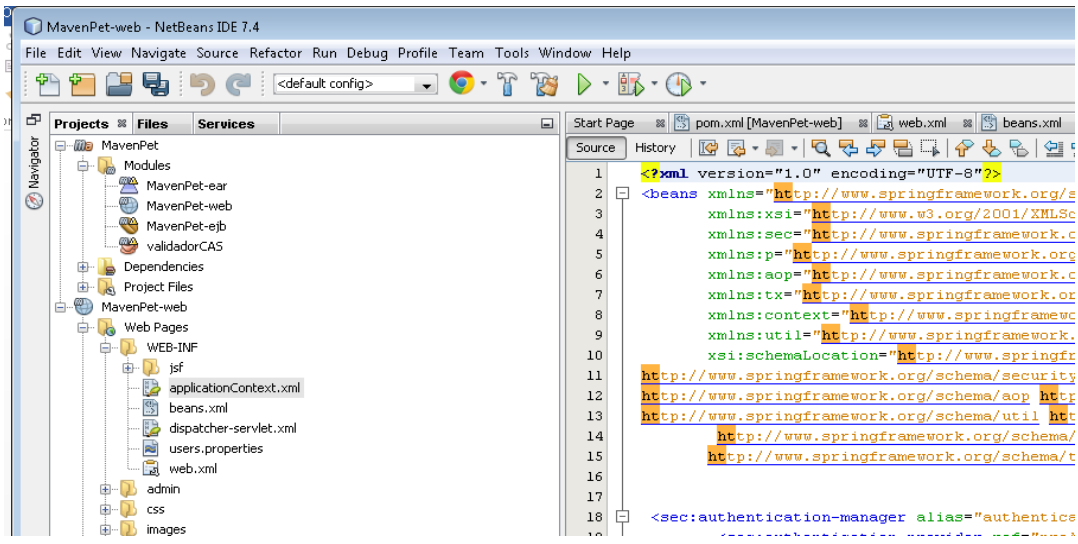
```
<listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>

<servlet>
    <servlet-name>dispatcher</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
    <load-on-startup>2</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>dispatcher</servlet-name>
    <url-pattern>*.html</url-pattern>
</servlet-mapping>

<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/applicationContext.xml</param-value>
</context-param>
```

Nota: puede modificar el nombre del dispatcher, aplicationContext teniendo en cuenta su ubicación

4.- Para los errores en la petición de usuarios con el CAS se debe agregar la siguiente línea dando la ruta de la página de acceso denegado al usuario quien quiera intentar ingresar a una página no autorizada.

```
<error-page>
    <error-code>403</error-code>
    <location>/denied.xhtml</location>
</error-page>
```

5.- Se procede a crear la página aplicationContext.xml como se lo indica en una de las guías vistas anteriormente



Colocando el siguiente código visto como ejemplo en el siguiente link…

https://wiki.jasig.org/display/CASC/Using+the+CAS+Client+3.1+with+Spring+Security

Modificando lo siguiente:

```
<bean class="org.jasig.cas.client.validation.Cas20ServiceTicketValidator" id="ticketValidator">
    <constructor-arg index="0" value="https://qianella09:8443/cas-server-webapp-3.5.2" />
    <property name="proxyGrantingTicketStorage" ref="proxyGrantingTicketStorage" />
    <property name="proxyCallbackUrl" value="https://qianella09:8443/cas-server-webapp-3.5.2/j_spring_cas_security_proxyreceptor" />
</bean>
```

NOTA:

- Colocando en el constructor el URL de nuestro cas
- Y en el proxyCallbackURL el URL del cas al final colocando /j_spring_cas_security_proxyreceptor que viene a ser como un módulo de cas y que debe ser el mismo colocado en el

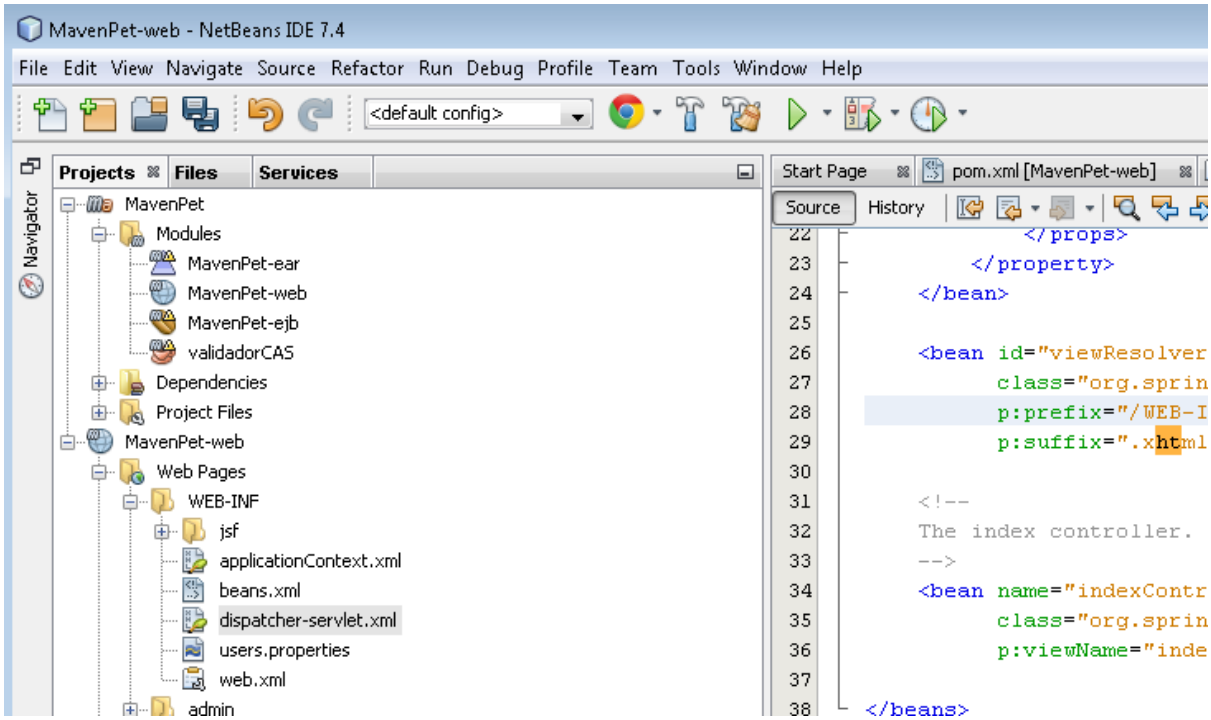
<property name="proxyReceptorUrl" value="/j_spring_cas_security_proxyreceptor"/>

```
<bean id="fsi" class="org.springframework.security.web.access.intercept.FilterSecurityInterceptor">
    <property name="authenticationManager" ref="authenticationManager"/>
    <property name="accessDecisionManager" ref="httpRequestAccessDecisionManager"/>
    <property name="securityMetadataSource">
        <sec:filter-invocation-definition-source >

            <sec:intercept-url pattern="/admin/**" access="ROLE_ADMIN"/>
            <sec:intercept-url pattern="/**" access="ROLE_CLIENTE"/>
            <sec:intercept-url pattern="/cLient/**" access="ROLE_CLIENT"/>
            <sec:intercept-url pattern="/employee/**" access="ROLE_EMPLOYEE"/>
            <sec:intercept-url pattern="/doctor/**" access="ROLE_DOCTOR"/>
            <sec:intercept-url pattern="/css/**" access="ROLE_CLIENT,ROLE_ADMIN,ROLE_EMPLOYEE,ROLE_DOCTOR"/>
            <sec:intercept-url pattern="/images/**" access="ROLE_CLIENT,ROLE_ADMIN,ROLE_EMPLOYEE,ROLE_DOCTOR"/>

        </sec:filter-invocation-definition-source>
    </property>

</bean>

 <bean id="casAuthenticationFilter" class="org.jasig.cas.client.authentication.AuthenticationFilter">
    <property name="casServerLoginUrl" value="https://qianella09:8443/cas-server-webapp-3.5.2/login" />
    <property name="serverName" value="https://qianella09:8443"/>
</bean>
```

NOTA:

En la sección de *intercept-url* se colocar los roles con las rutas a las carpetas a las que solo tiene acceso.

6.- Ahora se debe crear el documento despatcher-servlet.xml



Colocando el siguiente código…

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:p="http://www.springframework.org/schema/p"
       xmlns:aop="http://www.springframework.org/schema/aop"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.1.xsd
       http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-3.1.xsd
       http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.1.xsd">

    <bean class="org.springframework.web.servlet.mvc.support.ControllerClassNameHandlerMapping"/>

    <!--
    Most controllers will use the ControllerClassNameHandlerMapping above, but
    for the index controller we are using ParameterizableViewController, so we must
    define an explicit mapping for it.
    -->
    <bean id="urlMapping" class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
        <property name="mappings">
            <props>
                <prop key="index.xhtml">indexController</prop>
            </props>
        </property>
    </bean>
```

```
    <bean id="viewResolver"
          class="org.springframework.web.servlet.view.InternalResourceViewResolver"
          p:prefix="/WEB-INF/jsf/"
          p:suffix=".xhtml" />

    <!--
    The index controller.
    -->
    <bean name="indexController"
          class="org.springframework.web.servlet.mvc.ParameterizableViewController"
          p:viewName="index" />

/beans>
```

7.- Ahora para crear los roles se debe crear el siguiente documento llamado user.properties

NOTA:

Colocando los roles que se estarán vistos en nuestra aplicación y que cuyos usuario y passwords serán guardados en la BD

Ejemplo:

- admin= random,ROLE_ADMIN, enabled
- admin: usuario
- random: permite coger cualquier contraseña vista en la base
- ROLE_ ADMIN: nombre asignado para nuestro primer ROL

8.- Finalizando se crea por cada ROL sus respectivas carpetas con las páginas a las cuales está autorizado.