DESPLIEGUE DE UNA APLICACIÓN SPRING BOOT SOBRE TOMCAT7 Y MYSQL5 EN LA PLATAFORMA CLOUD OPENSHIFT

1 CONTENIDO

STUCOM
CENTRE D'ESTUDIS

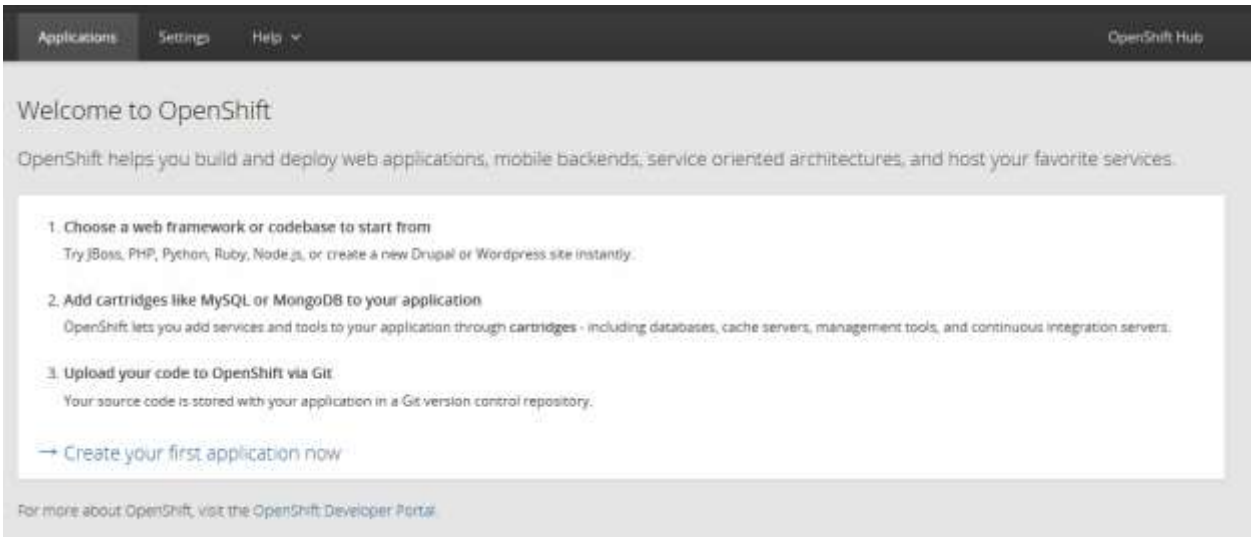
2 CREANDO APLICACIÓN EN OPENSHIFT

En esta guia vamos a desplegar nuestra aplicación creada con Spring utilizando el servicio Cloud de Openshift. Lo primero que debemos hacer es ir a la pagina oficial de Openshift:

https://www.openshift.com/

Una vez estamos en la pagina debemos registrarnos para poder crear nuestra aplicación o mejor dicho nuestro servidor donde desplegaremos nuesra aplicación.

Ya registrados nos logueamos, la primeravez que nos logueamos no tendremos ninguna aplicación creada por lo que debemos proceder a crearla.

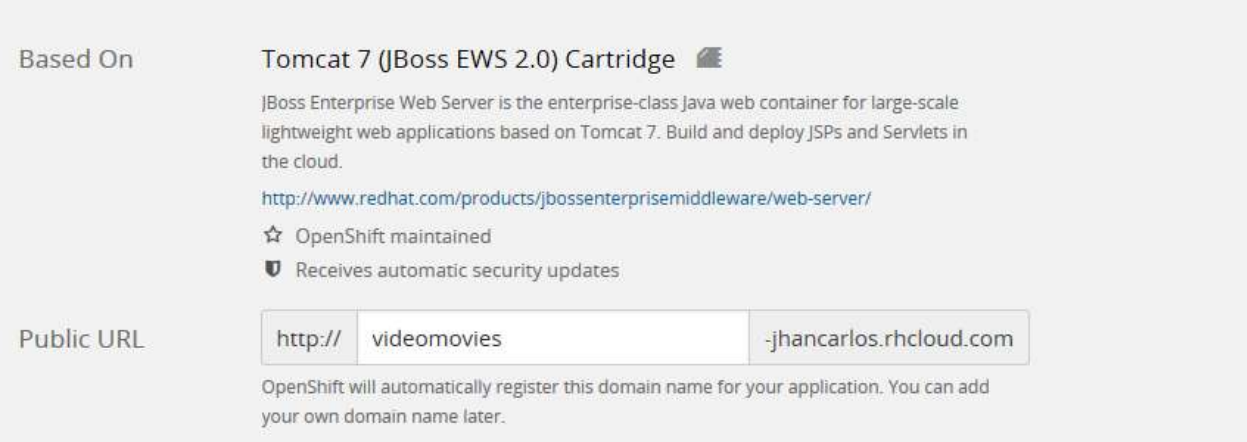


Para crearla debemos hacer clic en donde dice “Create your first application now”. Esto nos dirigirá a la siguiente ventana donde se mostrará un listado de “cartridges”.

En esta lista debemos bajar y en el apartado de Java buscamos el Tomcat 7 (JBoss EWS 2.0).

Hacemos clic y continuamos configurando nuestra aplicación.

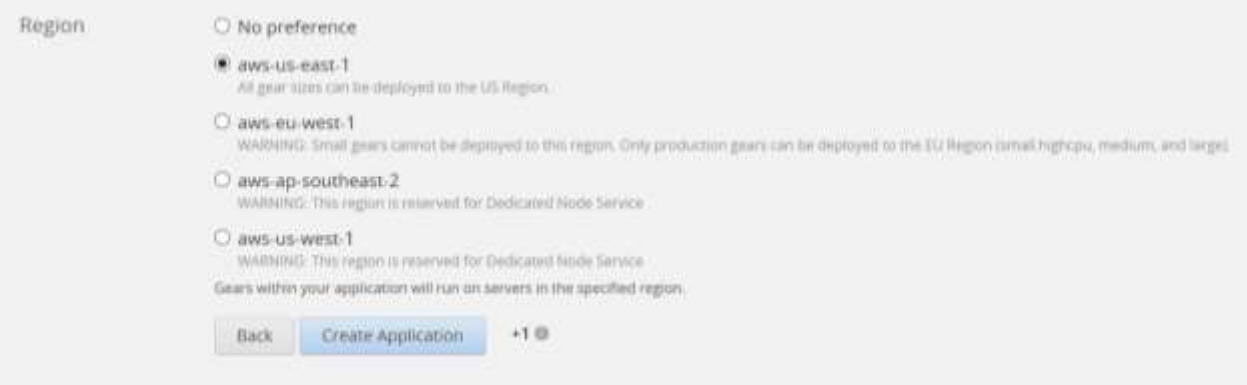
En la siguiente ventana debemos configurar la aplicación seleccionada, veremos que en el Public URL ya hay puesto un nombre por defecto que es “jbossews”, éste es el nombre que tendrá nuestra aplicación y también completará la URL con la que accederemos a ella, otra cosa muy importante es que ese será el nombre de la base de datos de la aplicación una vez añadido el MySQL. Colocamos el nombre que queremos que tenga nuestra aplicación y continuamos.



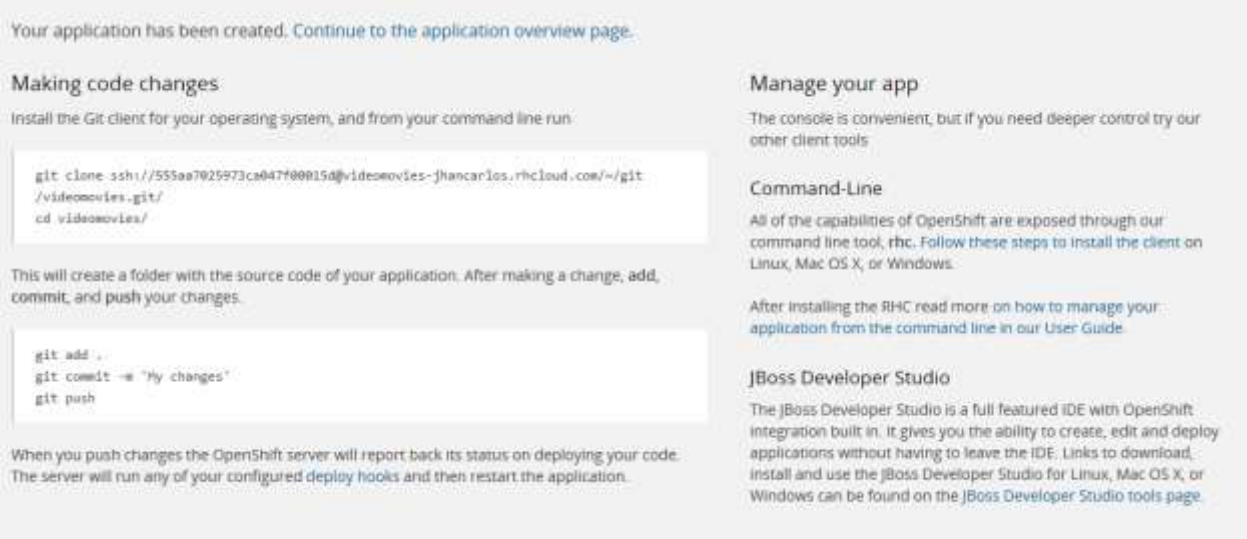
Lo siguiente que veremos será el Source Code, aquí podemos colocar la URL de alguna aplicación que tengamos subida a algún repositorio con las configuraciones correstas aplicadas. Debemos tener en cuenta que si ese repositorio no está bien configurado, la aplicación nos dará error al intentar desplegarse. En este caso lo dejaremos en blanco ya que utilizaremos otro método para desplegar nuestra aplicación.

Si continuamos podremos ver mas información sobre nuestra aplicación el Gear de la aplicación, que en este caso es “small” ya que es gratuita, el Cartridge que se creará, el Scaling el cual se utiliza para temas de balanceo de carga para cuando nuestra aplicación está recibiendo muchas peticiones. Podemos activar el Scaling y automaticamente se creará otra aplicación idéntica a la nuestra en la que se dirigirán las peticiones de los usuarios, así al repartir las peticiones entre las dos aplicaciones podemos mantener un servicio optimo. En principio, no activamos esta opción para esta demostración.

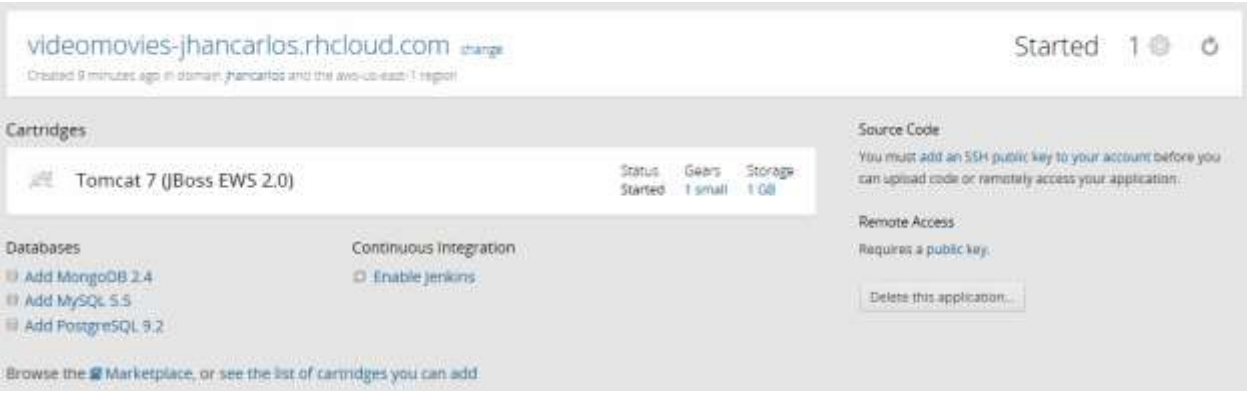
Luego tenemos la región donde estará nuestra aplicación, lo dejamos en el que está por defecto y le damos a “Create Application”.



Después de un rato nuestra aplicación estará lista. Aquí podremos ver la direccion del repositorio de la aplicación por defecto que crea Openshift, esto lo utilizaremos mas adelante para poder desplegar nuestra propia aplicación.



Si hacemos clic en “Continue to the application overview page” podremos ver las características de nuestra aplicación.



A continuacion , vamos a habilitar nuestro servicio de base de datos. Cuando creamos una aplicación Tomcat 7, por defecto tendremos el servicio Tomcat y el servicio Apache habilitados, si queremos tener base de datos tendremos que habilitarlo nosotros mismos.

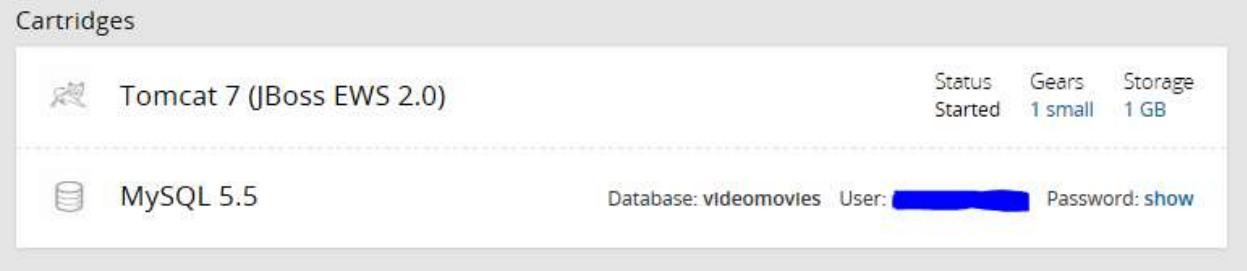
Esto es muy fácil, sólo debemos hacer clic en uno de los servicios del apartado de Databases. Podemos escoger entre diversas tecnologías: MongoDB, MySQL y PostgreSQL. En nuestro caso utilizaremos MySQl, hacemos clic en add MySQL.

Cuando hacemos clic nos dirigirán a otra página donde veremos un resumen del Cartridge, Openshift nos indica que añadirá un nuevo Cartridge a nuestra aplicación. Clicamos en Add Cartridge y continuamos.

Cuando finaliza la configuración de MySQL, Openshift nos redirige a nuestra aplicación y nos muestra una alerta verde donde estan los datos de acceso a nuestro MySQL. Recomiendo que hagáis una captura de estos datos, aunque podréis acceder a ellos desde la aplicación.

Tal y como indiqué anteriormente, el nombre de la base de datos será el mismo que el de la aplicación y en la siguiente captura aparece reflejado.

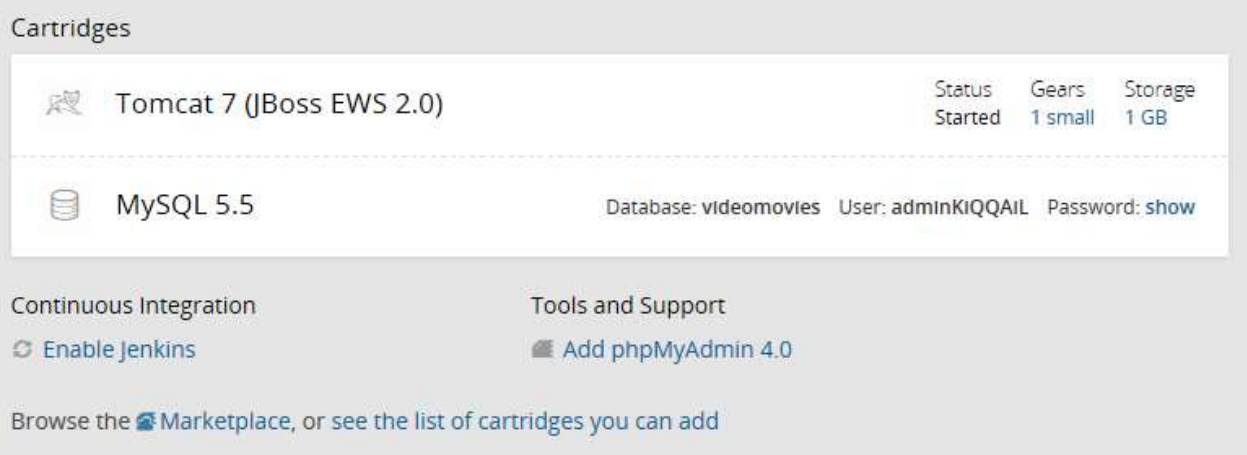
Si refrescamos la página, veremos que la alerta verde ha desaparecido pero, tal y como indiqué anteriormente, en nuestra aplicación podremos ver las credenciales de acceso a nuestro MySQL.



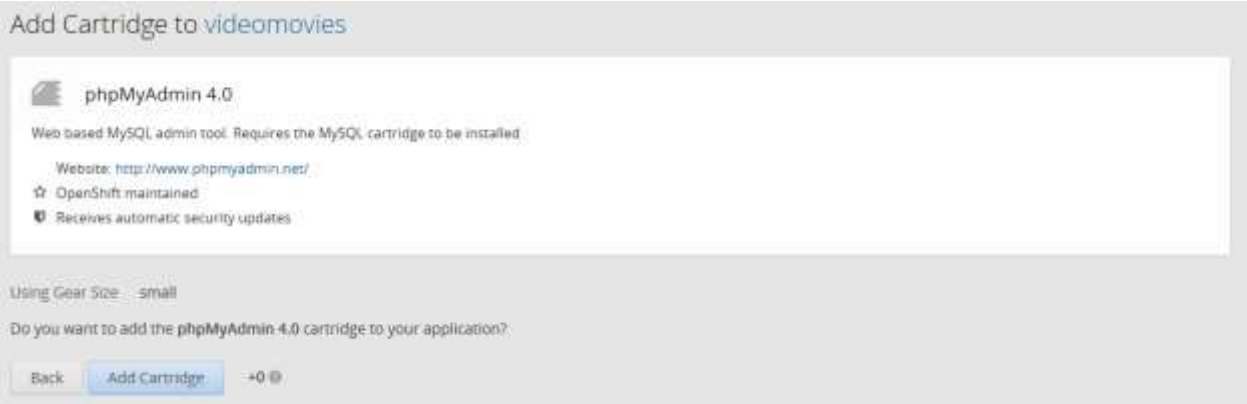
Sólo con hacer clic en show podremos ver la contraseña de nuestro usuario.

Ya tenemos nuestra aplicación con su servicio de base de datos, lo último que nos falta es añadir el phpMyAdmin para poder acceder a nuestra base de datos. Openshift nos permite añadirlo con sólo un clic al igual que con MySQL.

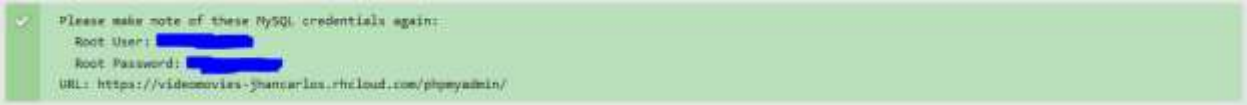
En el apartado de Tools and Support podemos ver que dice Add phpMyadmin. Hacemos clic aquí y lo añadimos.



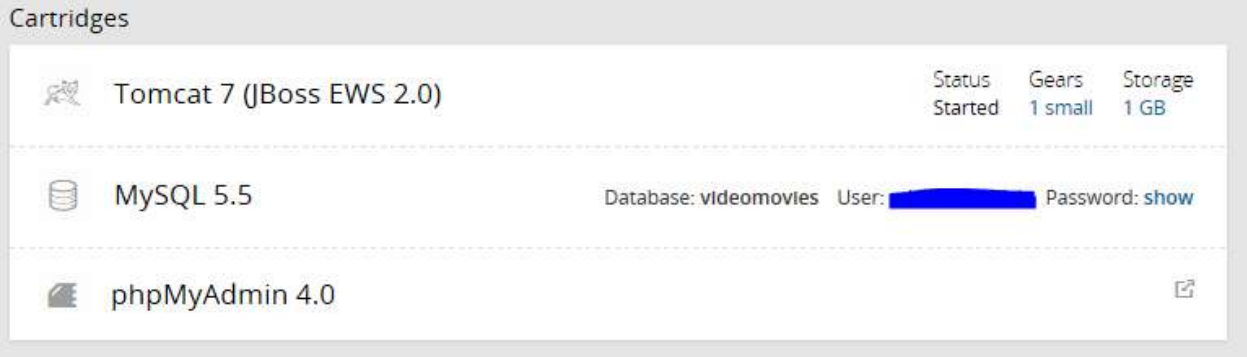
Pasará lo mismo que cuando añadimos el MySQL, hacemos clic en Add Cartridge y ya está disponible.



Una vez añadido nos redirigirán a nuestra aplicación y nos mostrarán otra vez una alerta verde con las credenciales de acceso y la URL de acceso al phpMyAdmin. Utilizamos la dirección que nos han dado para acceder a él.



Así es como queda nuestra aplicación con todos los Cartridges habilitados. Otra forma de acceder a nuestro phpMyAdmin es haciendo clic en el icono que hay a la derecha del texto de phpMyAdmin, es un enlace que nos llevará directamente a él.



Cuando accedemos a esta dirección se nos pedirán las credenciales de acceso a nuestro MySQL, las colocamos y podremos ver que accedemos al phpMyAdmin sin ningún problema. También podemos ver que tenemos creada nuestra base de datos.

Con esto ya tenemos la parte de Openshift completamente configurada. A continuacion procederemos a configurar las Openshift Client Tools (rhc) en nuestra máquina. Más adelante volveremos a nuestra aplicación de Openshift pero sólo para copiar la URL del repositorio git de nuestra aplicación.

3 INSTALANDO Y CONFIGURANDO LAS OPENSHIFT CLIENT TOOLS

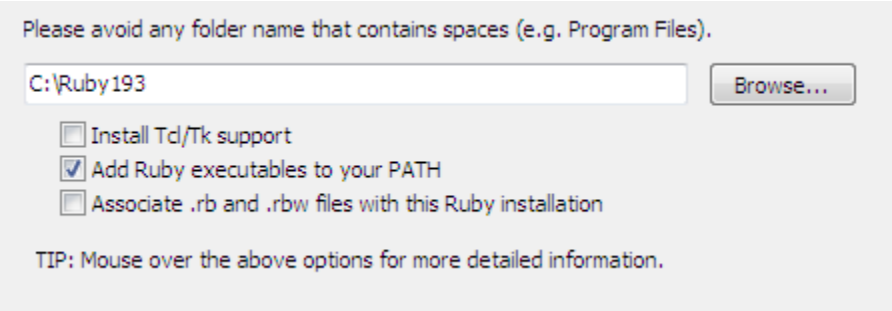
Antes de instalar las Client Tools debemos tener instalado en nuestro sistema Ruby y Git, en la siguiente URL está la guía oficial de OpenShift para instalar las Client Tools. Este enlace es para windows pero tambien hay guias para los demás sistemas operativos.

https://developers.openshift.com/en/getting-started-windows.html#client-tools

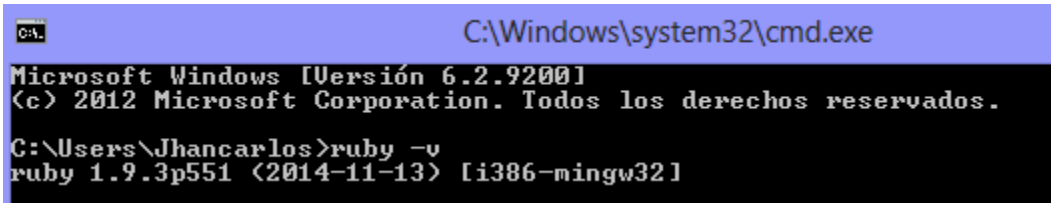
Si seguimos la guía podremos instalar las Client Tools sin ningun problema, lo primero que debemos hacer es instalar Ruby, la versión que recomienda Openshift es la 1.9.3.

http://dl.bintray.com/oneclick/rubyinstaller/rubyinstaller-1.9.3-p551.exe

Cuando estamos instalando Ruby debemos marcar la opcion que dice Add Ruby executables to your PATH, para que se añada al Path del sistema.



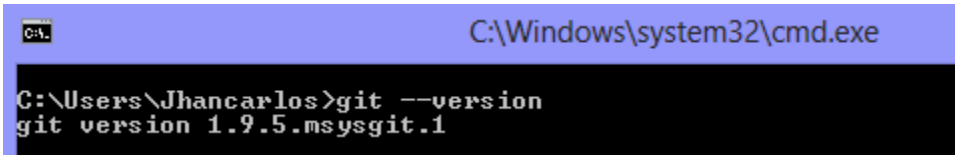
Para saber si tenemos Ruby añadido correctamente al Path, abrimos un terminal y ejecutamos **ruby –v**, debería mostrarnos la versión de ruby que tenemos instalada.



Luego debemos instalar Git, con git no hay ningun problema, podeis instalar la última versión.

https://github.com/msysgit/msysgit/releases/download/Git-1.9.5-preview20150319/Git-1.9.5-preview20150319.exe

lo mismo para comprobar si git tambien lo esta, pero con **git --version**.



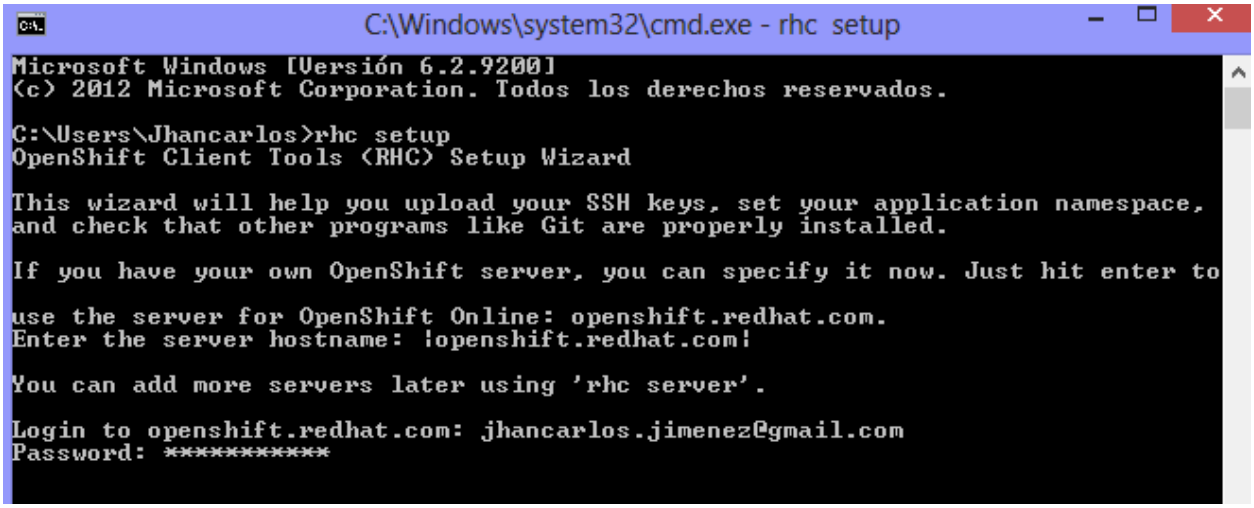
Una vez instaladas estas dos cosas procedemos a instalar las Client Tools o como las llama OpenShift **rhc**.

Para instalar las rhc sólo debemos escribir en nuestro terminal **gem install rhc**.

Yo ya las tenia instaladas pero las borré para que podáis ver el proceso de instalación, aquí solo me nuestra que se ha instalado una gem pero a vosotros os saldran nueve o más, si todo ha ido bien os saldrá vuestra línea de comandos otra vez, ahora sólo debemos ejecutar **rhc setup** para proceder con la configuración de las rhc.

```
C:\Users\Jhancarlos>gem install rhc
Fetching: rhc-1.35.3.gem (100%)
================================================================================

If this is your first time installing the RHC tools, please run 'rhc setup'

================================================================================
Successfully installed rhc-1.35.3
1 gem installed
Installing ri documentation for rhc-1.35.3...
Installing RDoc documentation for rhc-1.35.3...

C:\Users\Jhancarlos>
```

Ejecutamos rhc setup para proceder con la instalación, esto nos pedirá la dirección del host de Openshift, sólo debemos presionar Enter y lo pondrá automaticamente, luego nos pedirá nuestros datos de acceso a OpenShift, los colocamos y continuamos.



Una vez introducidos los datos, si son correctos el sistema de configuración nos indicará que va a crear un Token para que podamos acceder a OpenShift sin necesidad de colocar nuestras credenciales de acceso, este Token se guardará en nuestra carpeta de usuario en una carpeta llamada .openshift, escribimos yes y continuamos.

```
OpenShift can create and store a token on disk which allows to you to access the
server without using your password. The key is stored in your home directory and
should be kept secret.  You can delete the key at any time by running 'rhc
logout'.
Generate a token now? (yes|no) yes
Generating an authorization token for this client ... lasts about 1 month

Saving configuration to C:\Users\Jhancarlos\.openshift\express.conf ... done
```

A continuación se buscará en nuestro sistema nuestra key ssh, la cual será subida a openshift para poder acceder a nuestras aplicaciones, si el sistema no encuentra ninguna llave te avisará de que tienes que crear una. En mi caso yo ya la tengo porque la había generado antes para otro proyecto, en caso de que no la tengas solo debes seguir el sistema de configuracion. Te dirá que te creara un par de claves.

```
No SSH keys were found. We will generate a pair of keys for you.
Created: C:\Users\User1\.ssh\id_rsa.pub
```

Si te pregunta algo escribes yes a todo y continuas, luego llegará a un punto donde te preguntará si quieres subir tu clave publica a Openshift para poder acceder a tu codigo, escribes yes y continuas.

Tu Clave publica se sube a OpenShift y al momento queda todo completamente configurado, podrás ver que te muestras tus datos, chequea tu dominio y te dice el numero de aplicaciones que tienes creadas.

```
Your public SSH key must be uploaded to the OpenShift server to access code.
Upload now? (yes|no)
yes

Since you do not have any keys associated with your OpenShift account, your new
key will be uploaded as the 'default' key.

Uploading key 'default' ... done

Checking common problems

Your private SSH key file should be set as readable only to yourself.  Please
run 'chmod 600 C:\Users\Jhancarlos\.ssh\id_rsa'

Checking for a domain ... jhancarlos

Checking for applications ... found 1

  videomovies http://videomovies-jhancarlos.rhcloud.com/

  You are using 1 of 3 total gears
  The following gear sizes are available to you: small

Your client tools are now configured.

C:\Users\Jhancarlos>
```

Con esto finalizamos la instalación y configuración de las Client Tools (rhc), ahora sólo nos falta configurar nuestro proyecto para desplegarlo en Openshift.

Si queremos ver datos sobre las aplicaciones que tenemos instaladas sólo debemos escribir en nuestro terminal **rhc apps**, nos mostrará todas las aplicaciones que tenemos y sus datos al detalle.

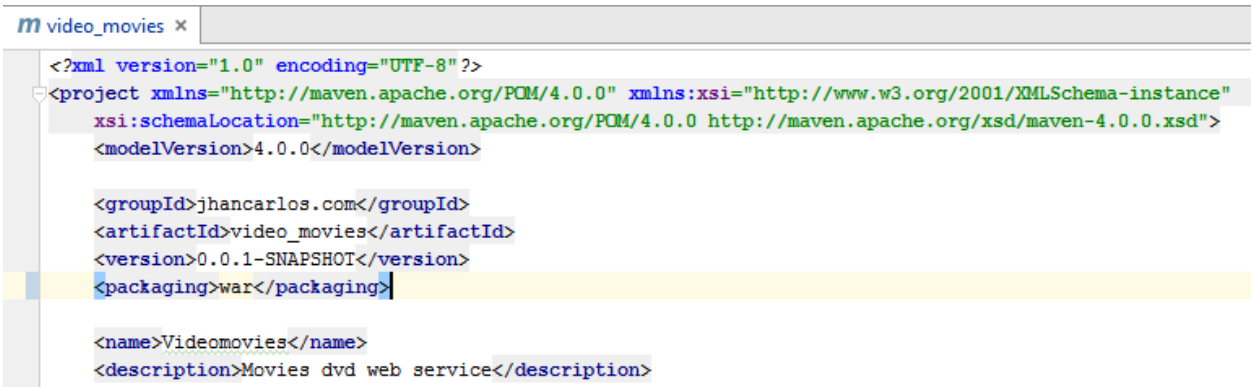
4 GENERANDO WAR DESDE INTELLIJ IDEA PARA DESPLEGAR EN OPENSHIFT

Una vez tenemos nuestra aplicación creada en Openshift y nuestro equipo configurado para poder interactuar con nuestra aplicación de Openshift podemos proceder a generar el War de nuestra propia aplicación para poder cambiar la de Openshift por la nuestra.

Antes de eso debemos tener algunas cosas claras. El primer punto importante es que debemos tener nuestro proyecto correctamente configurado. Una vez el proyecto está listo debemos tener el codigo subido a algun repositorio como Github o Bitbucket, ya que realizaremos cambios en los archivos del proyecto. Para gestionar las diversas configuraciones: entorno local (para desarrollo), entorno Openshift (para preproducción y producción) podríamos utilizar la funcionalidad de perfiles nativa que ofrece Spring.

El segundo punto importante es tener instalado el JDK de java 7, ya que Openshift aunque da soporte a java 8, debemos activarlo nosotros mismos manualmente utilizando DIY, de modo que debemos tener nuestro proyecto configurado para que compile con Java 7.

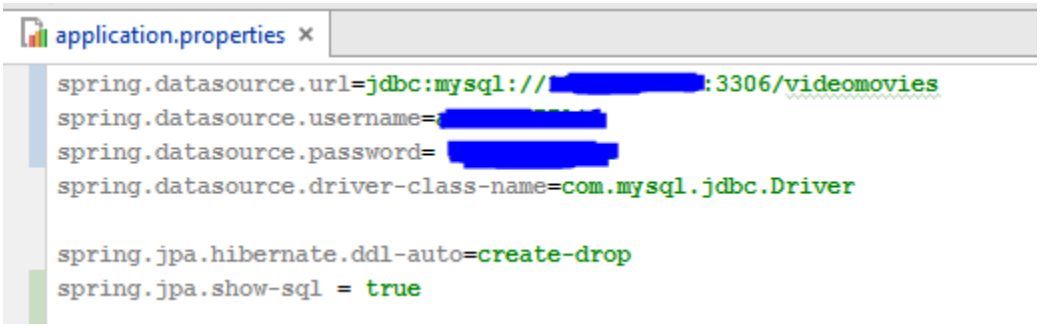
Vamos a empezar lo primero que debemos hacer es abrir el pom.xml de nuestro proyecto y cambiar el packaging de JAR a WAR.



Después debemos añadir la siguiente dependencia al pom.xml:

```
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-tomcat</artifactId>
   <scope>provided</scope>
</dependency>
```

Luego debemos colocar las credenciales de acceso en el application.properties, los datos de MySQL de Openshift. La IP de MySQL de OpenShift, el usuario y la contraseña, como ya sabéis si vais a Openshift podeis obtener esos datos desde vuestra aplicación.

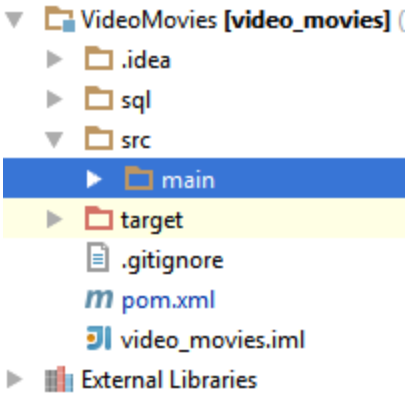


Una vez hecho esto debemos ejecutar en maven un **clean package** para que se genere nuestro WAR, pero antes de eso para no tener ningun error, vamos a eliminar de nuestro proyecto todo lo que esté relacionado con “tests”, esto lo hacemos porque al compilar el war se ejecutan los tests y como estos no se podran ejecutar correctamente habrá un error y no se podra generar el war.

Borramos del pom.xml todo lo que tenga que ver con tests, en mi caso tengo una dependencia. La borro y continuo.

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
</dependency>
```

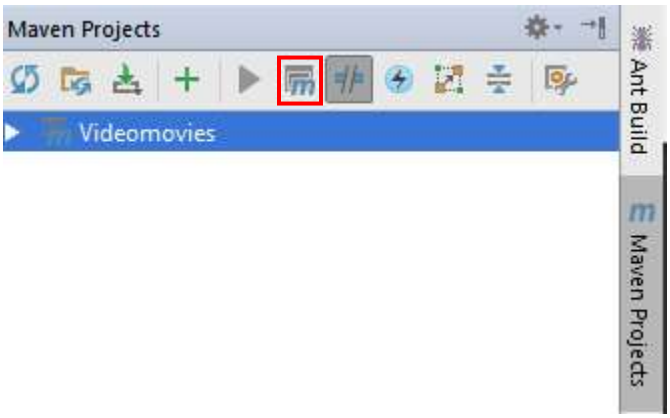
Ahora debemos borrar la carpeta de test que esta en el src de nuestro proyecto. Ahora sólo tenemos la carpeta main, con esto finalizamos el borrado de los tets.



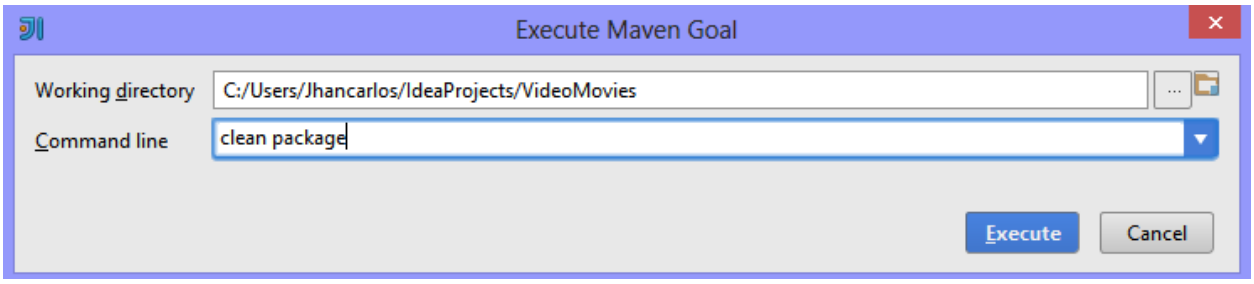
El último paso antes de hacer el clean package es crear una clase en nuestro package que herede de **SpringBootServletInitializer**. Esta es la clave para que cuando se despliegue el WAR en Openshift nuestra aplicación se inicialice. Con esto ya podemos proceder a ejecutar un clean package para generar el WAR.

```
package videomovies;

import org.springframework.boot.builder.SpringApplicationBuilder;
import org.springframework.boot.context.web.SpringBootServletInitializer;

public class VideoMoviesInitializer extends SpringBootServletInitializer {
    @Override
    protected SpringApplicationBuilder configure(SpringApplicationBuilder application){
        return application.sources(VideomoviesApplication.class);
    }
}
```

Desplegamos la pestaña de Maven Projects y hacemos clic en el icono que tiene una “m”.



Luego escribimos clean package y esto generara el WAR en la carpeta target.



Aquí vemos como finaliza la compilacion y se genera el WAR.

```
[INFO] Webapp assembled in [1084 msecs]
[INFO] Building war: C:\Users\Jhancarlos\IdeaProjects\VideoMovies\target\video_movies-0.0.1-SNAPSHOT.war
[INFO]
[INFO] --- spring-boot-maven-plugin:1.2.3.RELEASE:repackage (default) @ video_movies ---
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 16.806s
[INFO] Finished at: Mon May 18 16:37:21 CEST 2015
[INFO] Final Memory: 25M/177M
[INFO] ------------------------------------------------------------------------
```

Ahora solo debemos buscar ese WAR y desplegarlo en OpenShift.

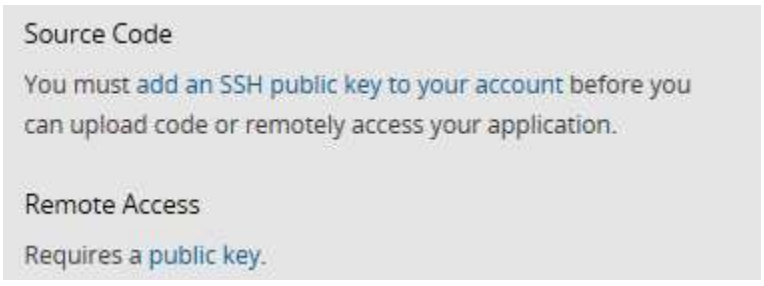
5 DESPLEGANDO WAR EN OPENSHIFT

Ahora que tenemos nuestro WAR generado solo nos falta desplegarlo en Openshift, para poder hacer esto debemos clonar el repositorio de nuestra aplicación de OpenShift a nuestro equipo y realizar algunos cambios.

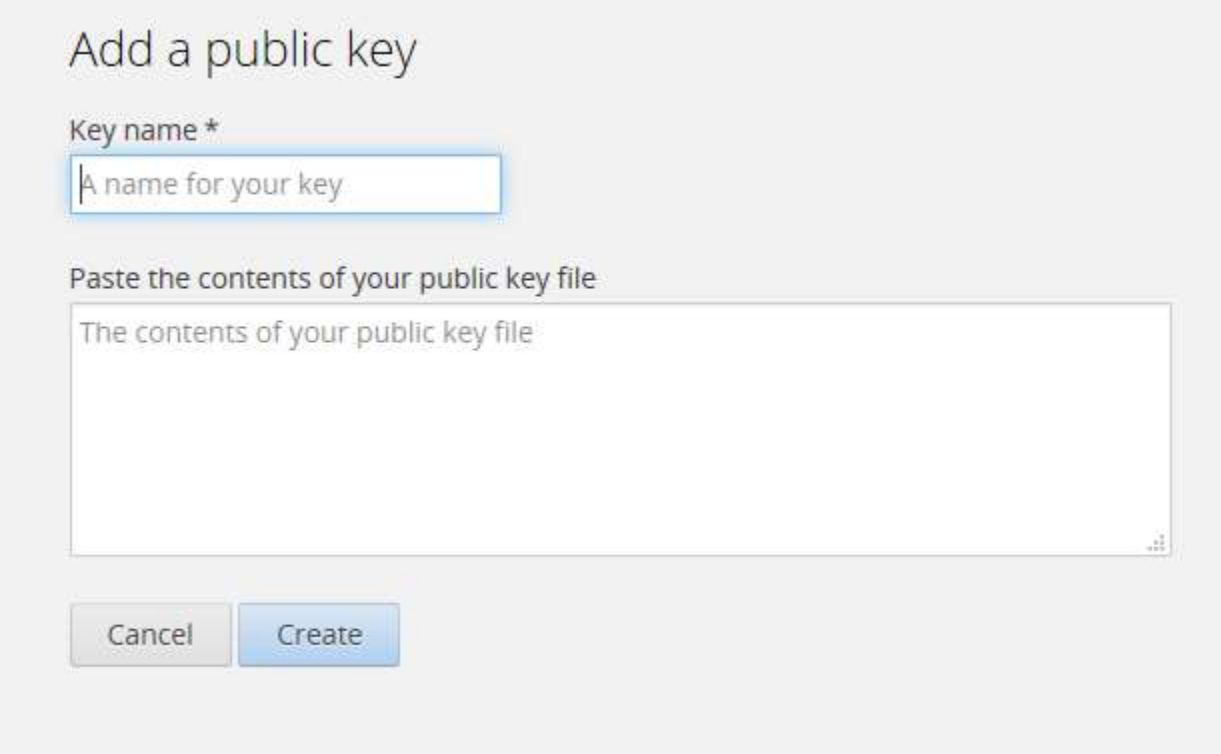
Para poder obtener la URL de nuestro repositorio lo podemos hacer de dos maneras, la primera es yendo al terminal y escribiendo rhc apps, aquí se mostraran los datos de nuestras aplicaciones incluyendo el repositorio de cada aplicación.

El segundo es via web, entramos a nuestra aplicación en Openshift por el navegador una vez alli, si miramos a la derecha hay un apartado que pone Source Code, aquí debemos añadir nuestra clave pública, será rapido porque ya está subida a Openshift, la subimos cuando configuramos las Client Tools.

Clicamos en add an SSH public key to your account.

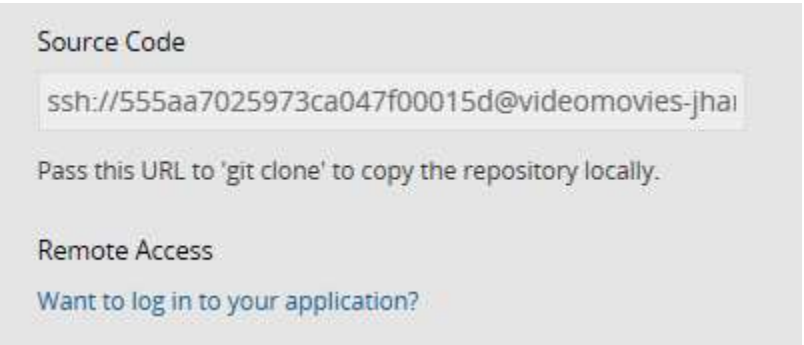


Nos saldra la siguiente pagina. Aquí no hacemos nada, volvemos atrás y entramos a Settings.

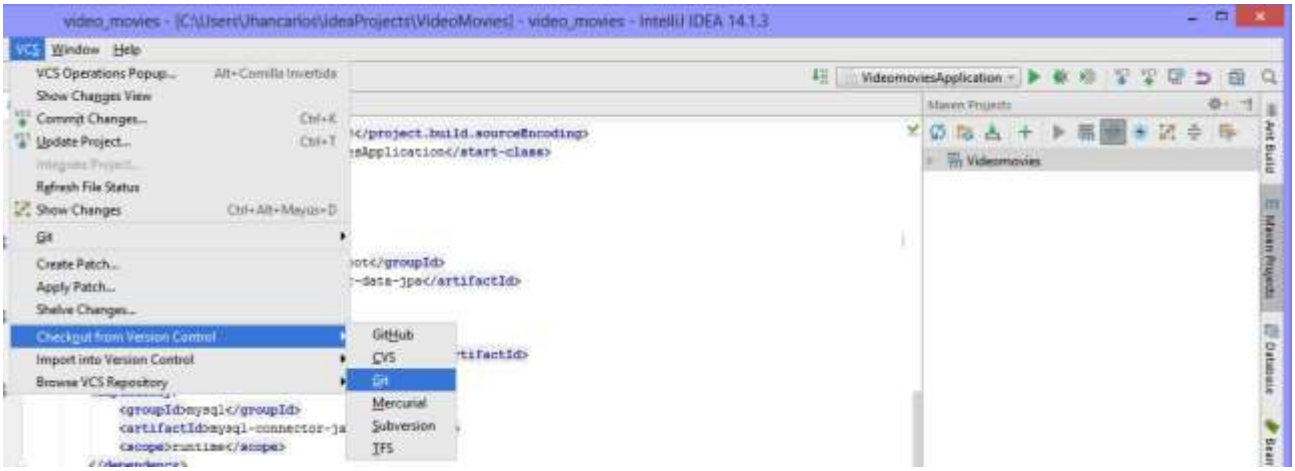


En Settings aparece un apartado que dice Public Keys, aquí estaba la clave pública que subió el sistema de configuracion de las Client Tools.

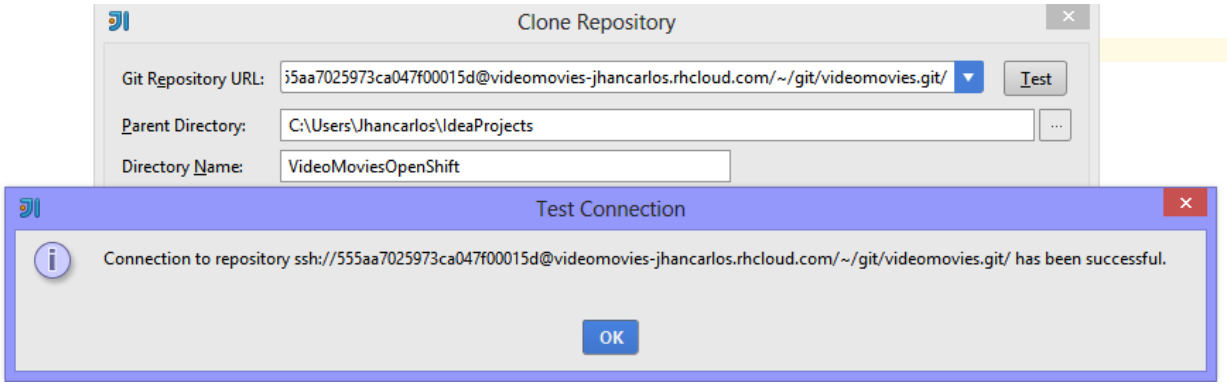
Volemos a nuestra aplicación y ahora veremos que nos muestra en el enlace de git para poder clonar el repositorio de nuestra aplicación.



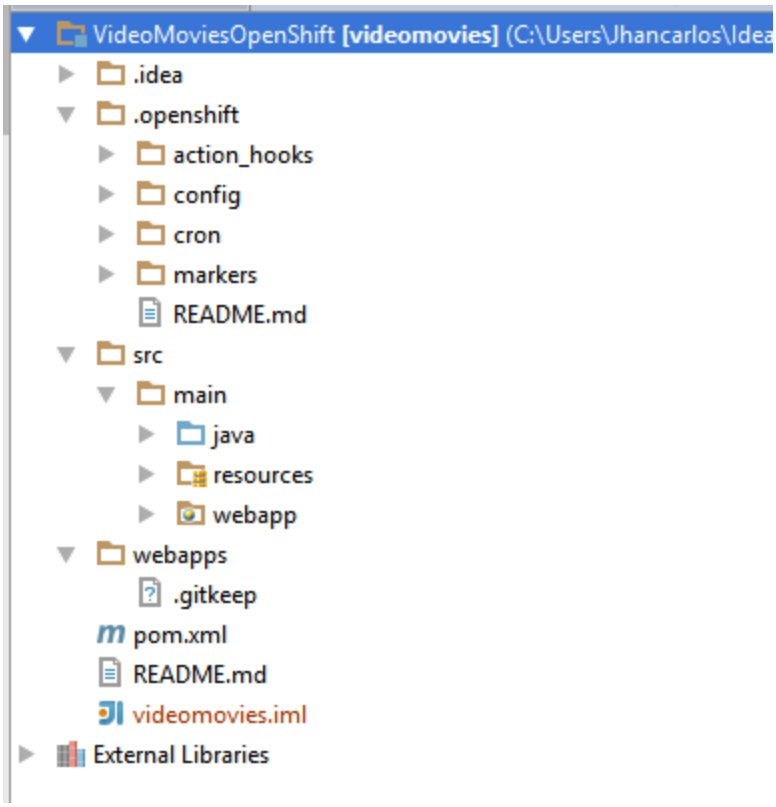
Copiamos la URL y procedemos a clonarlo en nuestra maquina. Para clonarlo desde IDEA lo podemos hacer muy facilmente de la siguiente manera. Vamos a VCS, luego a Checkout from Version Control y Clicamos Git.



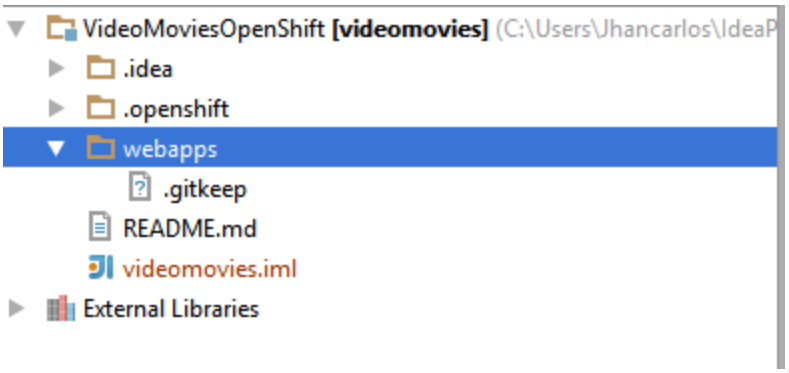
Colocamos la URL, hacemos un test de conectividad, debería de pasarlo, colocamos el nombre del proyecto y hacemos clic en Clone. Lo abrimos en una nueva ventana y procedemos a realizar los cambios necesarios.



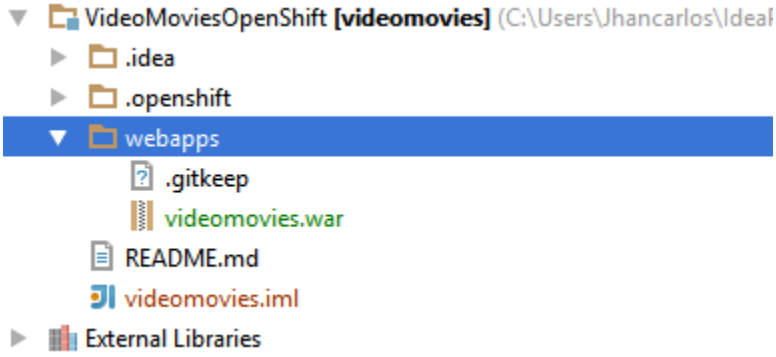
Éste es el proyecto que esta funcionando en Openshift.



Lo primero que debemos hacer es borrar la carpeta src y el pom.xml.

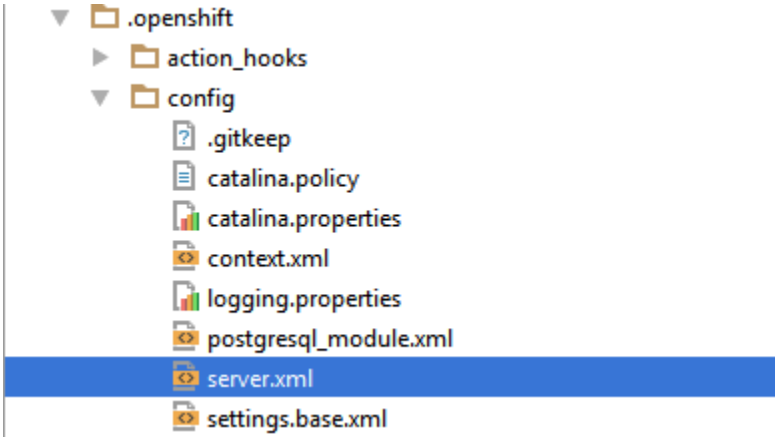


Una vez borradas esas carpetas, colocamos nuestro WAR en la carpeta webapps, aquí podemos colocar todos los WARS que queramos, cada uno será una aplicación independiente. Asi tambien podemos crear para cada WAR su base de datos si dichas aplicaciones las necesitan. Yo le he cambiado el nombre al WAR que ha generado el IDEA ya que necesito un nombre cómodo para poder acceder a él.



Antes de hacer el commit and push para desplegar nuestro WAR en Openshift debemos añadir algunas configuraciones más para que el acceso a nuestra aplicación sea segura por HTTPS.

Lo primero es editar el archivo server.xml que esta en .openshift/config/server.xml. Buscamos el siguiente conector y en el redirectPort pone 8443, le quitamos el 8 y guardamos.



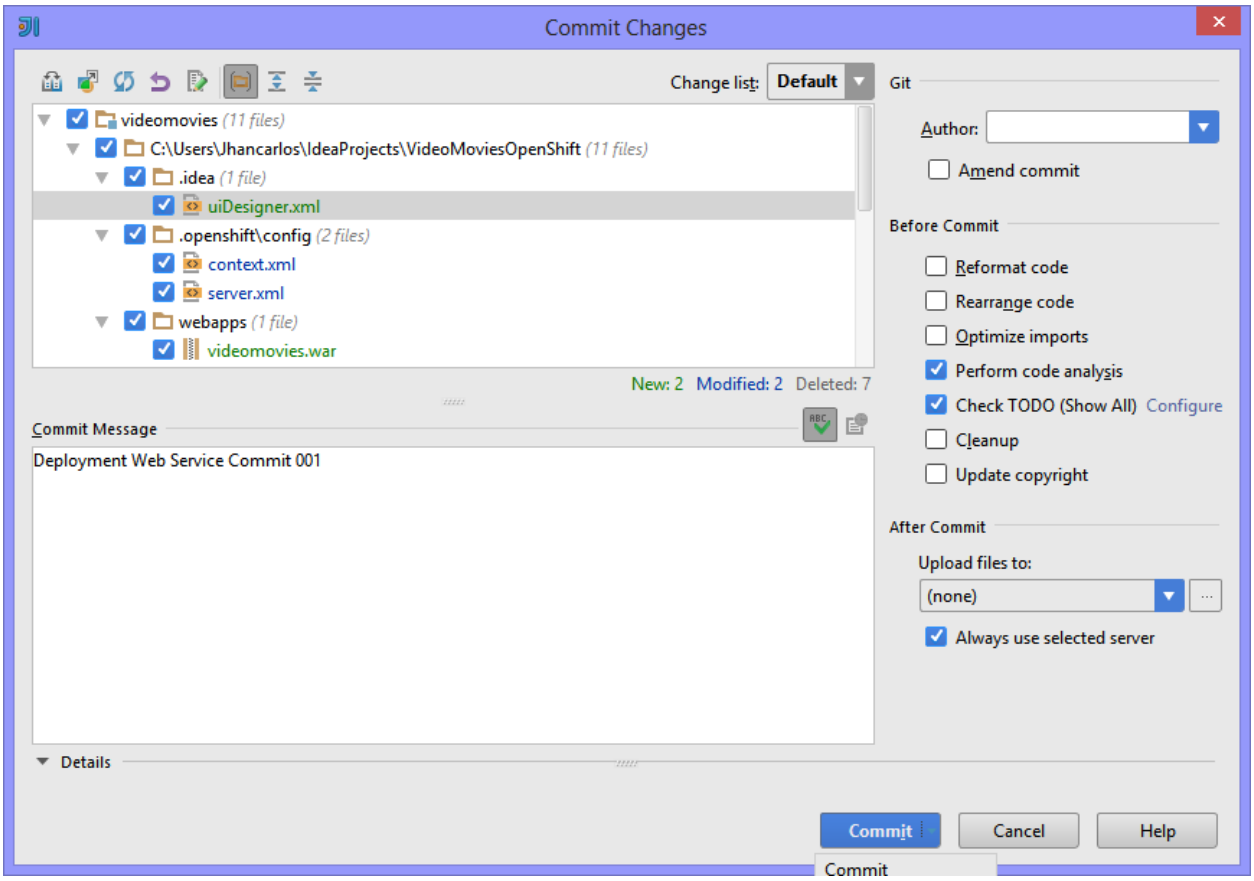

```
<Connector address="${OPENSHIFT_JBOSSEWS_IP}"
           port="${OPENSHIFT_JBOSSEWS_HTTP_PORT}"
           protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="443"/>
```

Luego vamos al archivo context.xml y añadimos la siguiente Valve. Este archivo esta en la misma carpeta que el server.xml en la carpeta config.

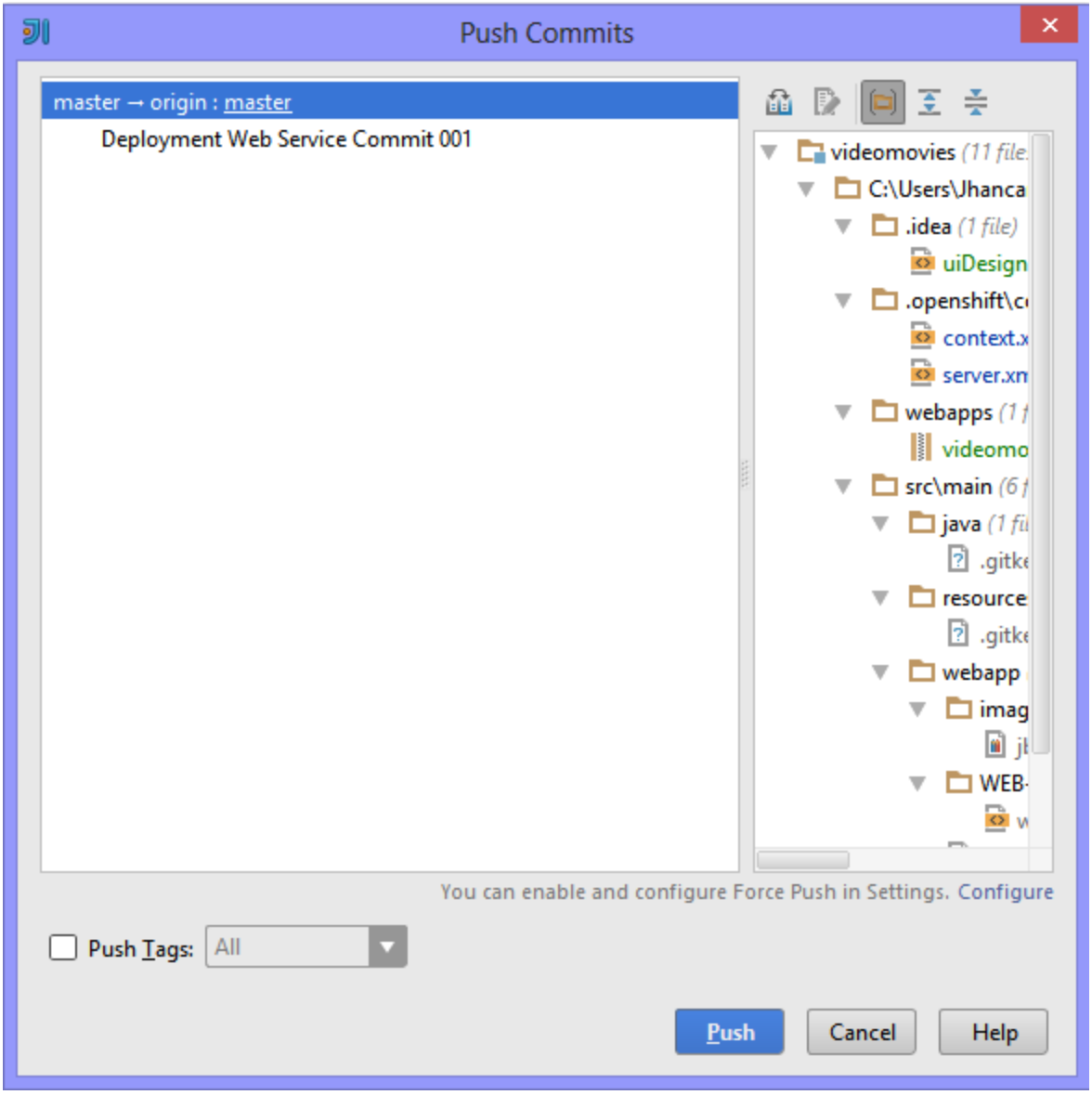
<Valve className="org.apache.catalina.valves.RemoteIpValve" protocolHeader="x-forwarded-proto"/>

```
<Valve className="org.apache.catalina.valves.RemoteIpValve" protocolHeader="x-forwarded-proto"/>
```

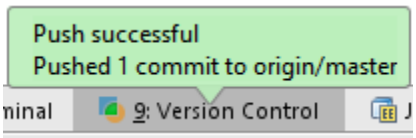
Ahora que ya tenemos todo correctamente configurado hacemos un commit and push.



Confirmamos el Push.



El push tardará un buen rato pero cuando haya finalizado, podremos acceder al servidor para comprobar que todo ha ido bien.



Para conectarnos a nuestra aplicación y comprobar que todo ha ido bien mirando el log de tomcat, debemos ir a un terminal y escribir, “rhc ssh –a nombreapp”.

Luego nos movemos a la carpeta de logs con cd app-root/logs y aquí hacemos un tail –f al archivo jbossws.log o un cat.

Aquí podremos ver que se genera el mapeo de nuestra aplicación, eso es una buena señal ya que es la parte que se ejecuta al final cuando el web service está finalizando su ejecucion para poder mostrar los datos.

Aquí teneis la URL para poder consumir este web service, tiene algunos datos de puebra pero podeis hacer inserts, updates y deletes utilizando el complemento Advanced Rest Client de Chrome.

Es posible que la primera petición tarde unos minutos, ya que hemos utilizado la cuenta básica gratuita de Openshift, y esta plataforma Cloud detiene las máquinas en el caso de que no se hayan utilizado durante 48 horas (y por tanto, debe reiniciar las máquinas y los servicios necesarios antes de poder servir la petición)

http://videomovies-jhancarlos.rhcloud.com/videomovies

Con esto finalizamos esta guía, todo está explicado al detalle y no creo que os falle nada, mucha suerte y Happy Coding.

Centro de estudios Stucom

Ciclo Formativo: Desarrollo de Aplicaciones Multiplataforma

Alumno: Jhancarlos Marte Jiménez

Profesor responsable: Alfredo Rueda Unsain
www.linkedin.com/in/alfredorueda