Despliegue

1. Obtener código a publicar

```
git checkout <rama-a-publicar> && git pull origin <rama-a-publicar>
```

2. Clave de acceso al servidor

Colocar la key `chainz.key` en el directorio donde se encuentra el script `deploy.sh`

3. Ensamblar la solución

Crear archivo `.jar` con todas las dependencias.

```
# ejecutar en raíz de la solución
assembly:assembly -f pom.xml
```

o equivalentemente en *IntelliJ*: Maven → Plugins → `assembly:assembly`

4. Desplegar el código ensamblado

Ejecutar `deploy.sh`

Esto subirá el `.jar` al servidor, ejecutará el script `setup_db.sh` ubicado en el servidor, finalizará el servicio en ejecución y ejecutará otro script `start.sh` ubicado en el servidor que se encargará de volver a levantar el servicio.

Script `setup_db.sh`

Como no pudimos encontrar una solución fácil y rápida para usar variables de entorno que especifiquen los datos de la base de datos a usar en el archivo Java, optamos por reemplazar el `persistance.xml` en el archivo `.jar` como parte de nuestro proceso de deploy ayudándonos del script `setup_db.sh`.

Este nuevo `persistance.xml` se encuentra en el servidor y tendrá los secretos de la instancia MySQL que también se encuentra en ejecución allí.

Script `start.sh`

Este script es el encargado de declarar las variables de entorno necesarias (como por ejemplo la configuración del servidor SMTP) y de volver a correr el servicio de Java en el servidor.