# Configuraciones básicas Firebase/Springboot/CleverCloud/Heroku

En base a errores comunes surgidos a la hora de realizar el release o deploy de nuestro proyecto Full Stack se formulan una serie de modificaciones, las cuales detallo a continuación:

1. **Error: Incompatibilidad de versiones de Java y/o Maven**

*[ERROR] Failed to execute goal org.apache.maven.plugins:maven-compiler-plugin:3.8.1:compile (default-compile) on project NOMBRE\_DE\_TU\_PROYECTO: Fatal error compiling: invalid target release: 8 -> [Help 1]* (tambien puede darse en las versions 11 y 17)

Esto se debe a la incompatibilidad de las versiones de Java y/o Maven que tienes en tu PC Local con las que intenta instalar heroku remotamente.

Para solucionarlo debes indicarle a heroku las versiones que usas en tu proyecto, creando un archivo de propiedades de sistema (***system.properties***) especificando las versiones de Java y/o Maven que usas localmente, y luego subirlos para heroku los tenga en cuenta. Básicamente debes seguir los siguientes pasos:

***\*\*\*\*\*\*(Aclaración: las versiones son de ejemplo, por favor coloca las versiones que tienes en tu pc)\*\*\*\*\*\*\*\*\****

*1- Crear el archivo****system.properties****en la raíz de tu proyecto con el siguiente contenido:*

***java.runtime.version=11***

***maven.version=3.6.2***

*2- Agregar a git el archivo, ejecutando en la raíz de tu proyecto:*

***git add system.properties***

*o*

***git add .***

*3- Hacer commit de los cambios:*

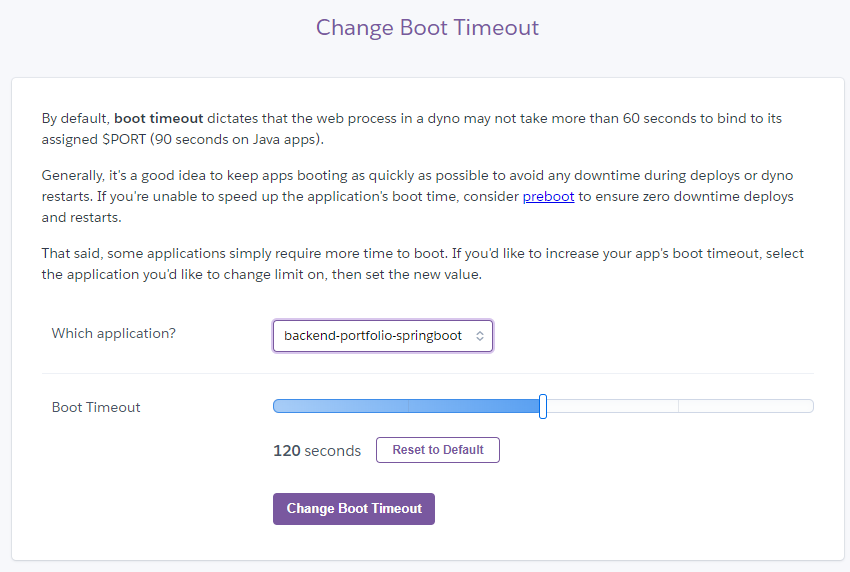
***git commit -m "se agrega archivo de configuraciones de sistema"***

*4- Finalmente realizar:*

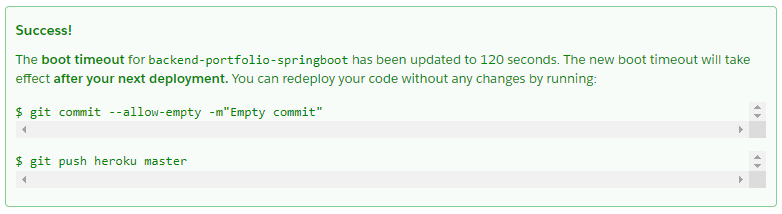
***git push heroku master***

*referencia:*[*https://devcenter.heroku.com/articles/java-support#specifying-a-java-version*](https://devcenter.heroku.com/articles/java-support#specifying-a-java-version)

1. **Error: Error timeout heroku/clevercloud:**
2. Abrir la url: <https://tools.heroku.support/limits/boot_timeout>
3. En la ventana que se abre, seleccionar nuestra app, desplazar el control para aumentar el tiempo de espera y presionar el botón ***Change Boot Timeout***:

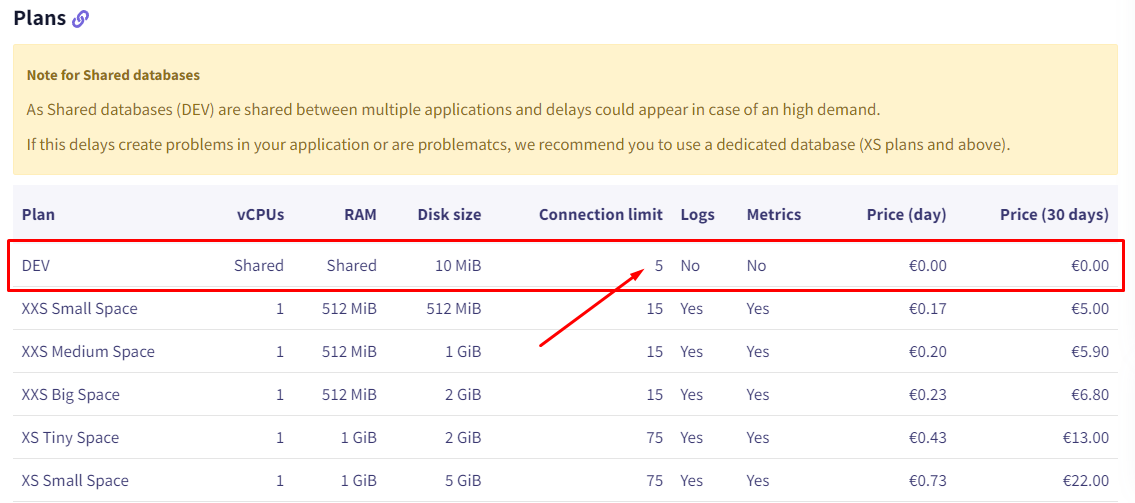


1. Por último, seguir las instrucciones:



1. **Error: CleverCloud Supera el límite de conexiones**

A partir de Spring Boot 2 se agrega por defecto Hikari como pool de conexiones. Por defecto se crean 10 conexiones a la base de datos. El problema es que Clever Cloud en su versión gratuita solo permite 5 conexiones.

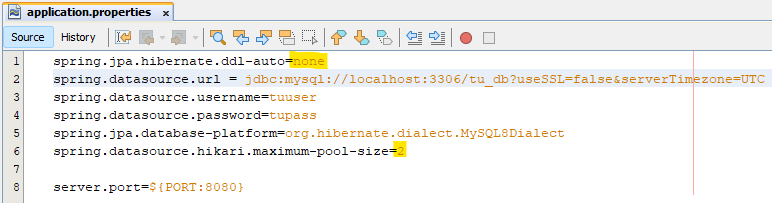


Es por esta razón que una vez que se inicia el backend ya no es posible conectarse a la base desde otra herramienta como MySQL Workbench por ejemplo.  
  
Para solucionar el problema, se puede configurar el pool de conexiones para que el backend no consuma todas las conexiones disponibles.

Para esto, debes agregar al archivo *application.properties*:

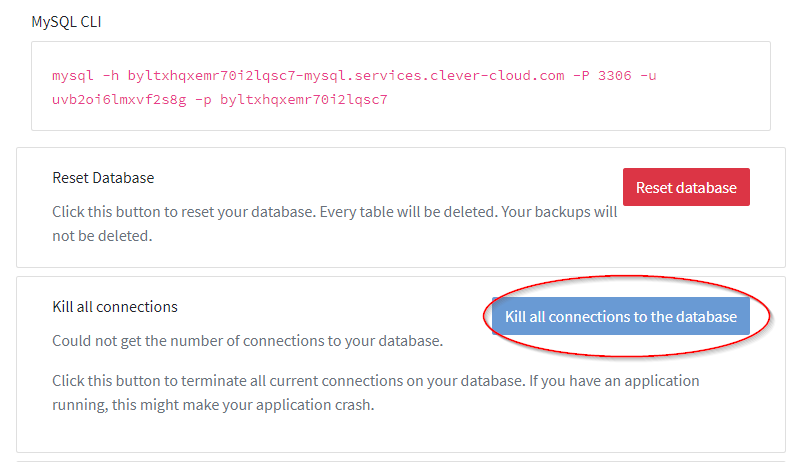
***spring.datasource.hikari.maximum-pool-size=2***

lo cual configura el pool de conexiones para que cree 2 conexiones como máximo a la base de datos, que para el caso del portfolio son suficientes. De esta forma se puede conectar a la base de datos desde una herramienta externa ya que hay conexiones disponibles, teniendo el servicio SpringBoot en ejecución.



Solución alternativa (no tan eficiente):

Antes de ejecutar tu app, desde clevercloud prueba cerrar todas las conexiones (como adjunto en la imagen).



1. **Solución Error: CORS deshabilitados**

*1- Crear el archivo****WebConfig.java*** *en la raíz de tu proyecto con el siguiente contenido:*

***import org.springframework.context.annotation.Configuration;***

***import org.springframework.web.servlet.config.annotation.CorsRegistry;***

***import org.springframework.web.servlet.config.annotation.EnableWebMvc;***

***import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;***

***/\*\****

***\****

***\* @author Ema***

***\*/***

***@EnableWebMvc***

***@Configuration***

***public class WebConfig implements WebMvcConfigurer {***

***@Override***

***public void addCorsMappings(CorsRegistry registry) {***

***registry.addMapping("/\*\*");***

***}***

***}***

*2- Agregar a git el archivo, ejecutando en la raíz de tu proyecto:*

***git add WebConfig.java***

*o*

***git add .***

*3- Hacer commit de los cambios:*

***git commit -m "se agrega archivo que habilita CORS"***

*4- Finalmente realizar:*

***git push heroku master***

PD: Ejemplo de cómo quedaría

