

# GUÍA DE DESPLIEGUE

STORM PRINT 3D

**Juan José Librero Cansino**

Técnico Superior en Desarrollo de Aplicaciones Web



## Índice

Instalación de Docker	2
<b>Activar la virtualización en nuestra máquina</b>	2
Instalación de otras herramientas	4
<b>Instalación de Maven</b>	4
<b>Instalación de JDK</b>	4
Descargar el proyecto	5
Iniciar la aplicación	6
Ver o editar el código	8
Si queremos ver los datos registrados en la base de datos	11
Desplegar un proyecto en heroku	13



## Instalación de Docker

La aplicación Storm Print 3D se ha dockerizado al completo para poder instalar y usar fácilmente.

Lo primero que debemos hacer es instalar **Docker**, para ello vamos a la siguiente url para descargarlo: <https://www.docker.com/products/docker-desktop>

# Docker Desktop

The fastest way to containerize applications on your desktop

Download for Windows

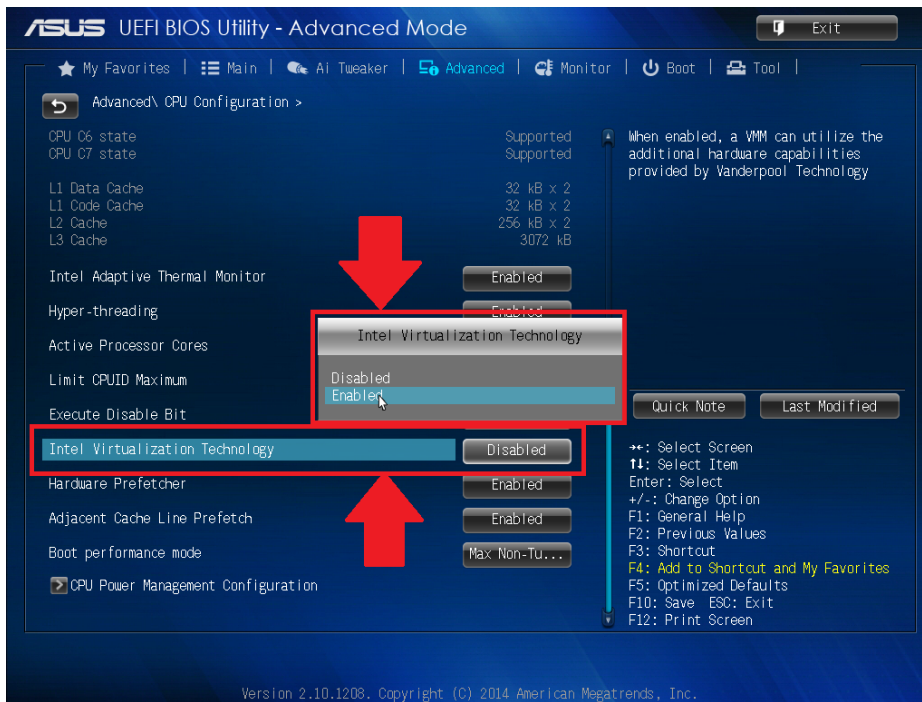
Also available for [Mac](#) and [Linux](#)

By downloading this, you agree to the terms of the [Docker Software End User License Agreement](#) and the [Docker Data Processing Agreement \(DPA\)](#).

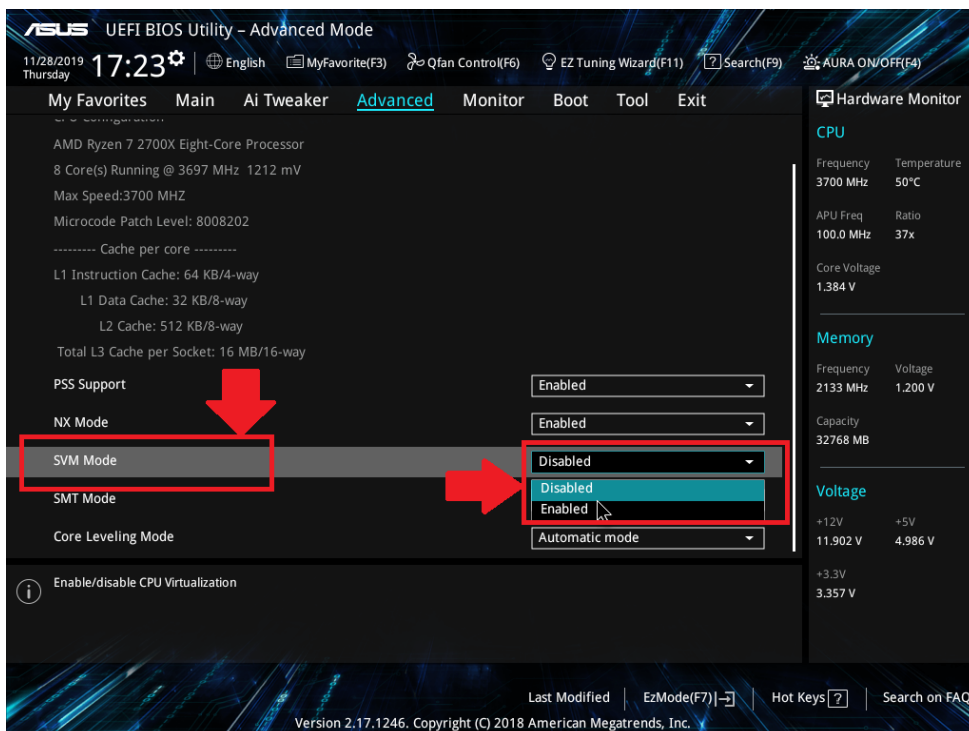
## Activar la virtualización en nuestra máquina

Para utilizar Docker en nuestro ordenador es muy importante activar la **virtualización** en nuestro sistema.

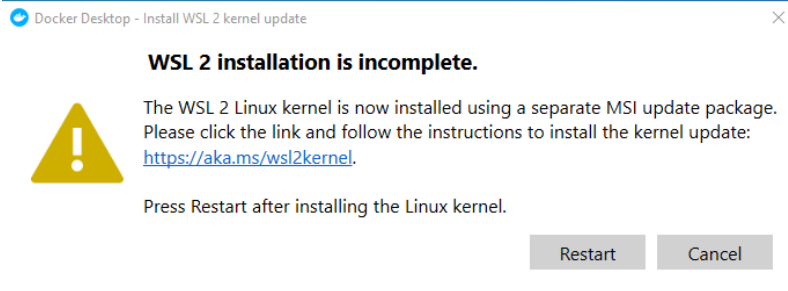
Si nuestro ordenador tiene un procesador Intel, tenemos que activar **Intel Virtualization Technology** en nuestro BIOS.



En el caso de que nuestro procesador sea AMD debemos activar **SVM Mode** en nuestro BIOS.



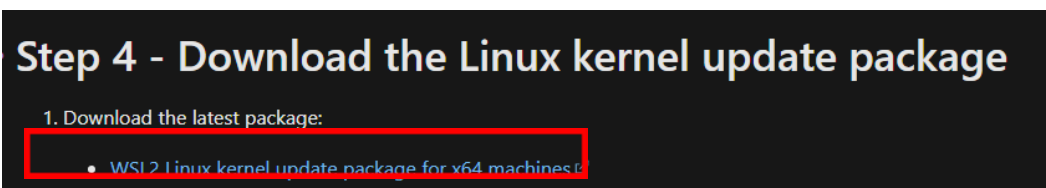
Si al iniciar Docker nos salta este error:



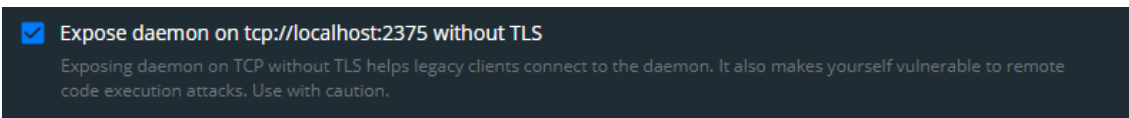
Debemos ir al enlace

<https://docs.microsoft.com/en-us/windows/wsl/install-win10#step-4---download-the-linux-kernel-update-package>

Y descargar e instalar **WSL2 Linux kernel update package for x64 machines**



Cuando tengamos Docker iniciado vamos a Settings y activamos **Expose daemon on tcp://localhost:2375 without TLS.**



## Instalación de otras herramientas

### Instalación de Maven

Apache Maven es una herramienta necesaria para ejecutar nuestra aplicación. Lo descargamos de la web oficial y lo instalamos. URL:

<https://maven.apache.org/download.cgi?Preferred=https%3A%2F%2Fftp.cixug.es%2Fapache%2F>

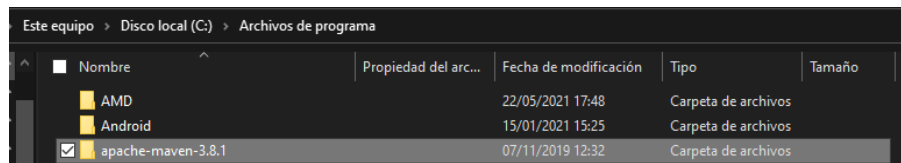
### Files

Maven is distributed in several formats for your convenience. Simply pick a ready-made binary

In order to guard against corrupted downloads/installations, it is highly recommended to [verify the](#)

	Link
Binary tar.gz archive	<a href="#">apache-maven-3.8.1-bin.tar.gz</a>
Binary zip archive	<a href="#">apache-maven-3.8.1-bin.zip</a>
Source tar.gz archive	<a href="#">apache-maven-3.8.1-src.tar.gz</a>
Source zip archive	<a href="#">apache-maven-3.8.1-src.zip</a>

Se descarga un .zip, tendremos que extraer la carpeta en **C:\Program Files**




## Instalación de JDK

Necesitaremos instalar el JDK de Java para el funcionamiento de nuestra aplicación.

Vamos a la url: <https://www.oracle.com/java/technologies/javase-jdk16-downloads.html>

Y descargamos el instalador.

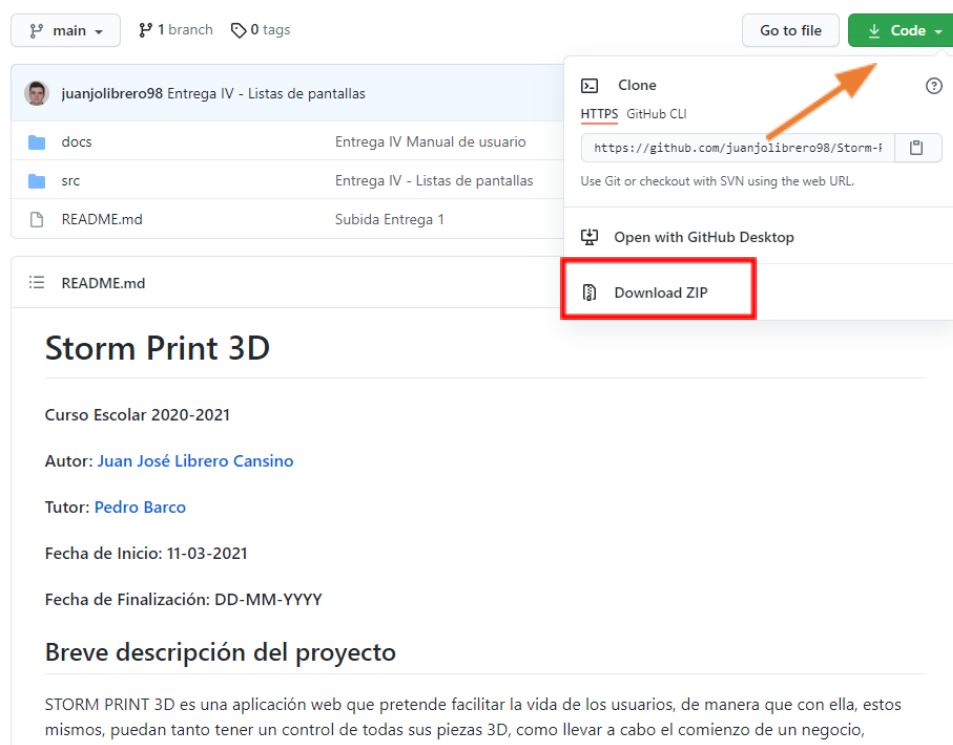
Windows x64 Installer	150.56 MB	 jdk-16.0.1_windows-x64_bin.exe
-----------------------	-----------	--

## Descargar el proyecto

Ahora descargamos el proyecto de la aplicación.

Descargamos el proyecto con Git, se puede utilizar la aplicación de GitHub Desktop para descargarlo o también ir directamente al enlace del repositorio y descargar el .zip

URL del repositorio: [Storm Print 3D JJLC](https://github.com/juanjolibrero98/Storm-Print-3D-JJLC)



## Iniciar la aplicación

Cuando tengamos el proyecto descargado, abrimos un CMD y nos dirigimos a la ruta donde se ha descargado nuestro proyecto.

```

C:\Users\juanj\Documents\WorkspacePrueba\StormPrint3D>mvn install -DskipTests
[INFO] Scanning for projects...
[INFO]
[INFO] -----< org.iesalixar.jjoselibreroc:StormPrint3D >-----
[INFO] Building StormPrint3D 0.0.1-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-resources-plugin:3.1.0:resources (default-resources) @ StormPrint3D ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 1 resource
[INFO] Copying 41 resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ StormPrint3D ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:3.1.0:testResources (default-testResources) @ StormPrint3D ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\Users\juanj\Documents\WorkspacePrueba\StormPrint3D\src\test\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:testCompile (default-testCompile) @ StormPrint3D ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:2.22.2:test (default-test) @ StormPrint3D ---
[INFO] Tests are skipped.
[INFO]
[INFO] --- maven-jar-plugin:3.2.0:jar (default-jar) @ StormPrint3D ---
[INFO] Building jar: C:\Users\juanj\Documents\WorkspacePrueba\StormPrint3D\target\StormPrint3D-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- spring-boot-maven-plugin:2.4.3:repackage (repackage) @ StormPrint3D ---
[INFO] Replacing main artifact with repackaged archive
[INFO]
[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ StormPrint3D ---
[INFO] Installing C:\Users\juanj\Documents\WorkspacePrueba\StormPrint3D\target\StormPrint3D-0.0.1-SNAPSHOT.jar to C:\Users\juanj\.m2\repository\org\iesalixar\jjoselibreroc\StormPrint3D\0.0.1-SNAPSHOT.jar
[INFO] Installing C:\Users\juanj\Documents\WorkspacePrueba\StormPrint3D\pom.xml to C:\Users\juanj\.m2\repository\org\iesalixar\jjoselibreroc\StormPrint3D\0.0.1-SNAPSHOT.pom
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 2.139 s
[INFO] Finished at: 2021-06-15T09:47:12+02:00
[INFO]
C:\Users\juanj\Documents\WorkspacePrueba\StormPrint3D>

```

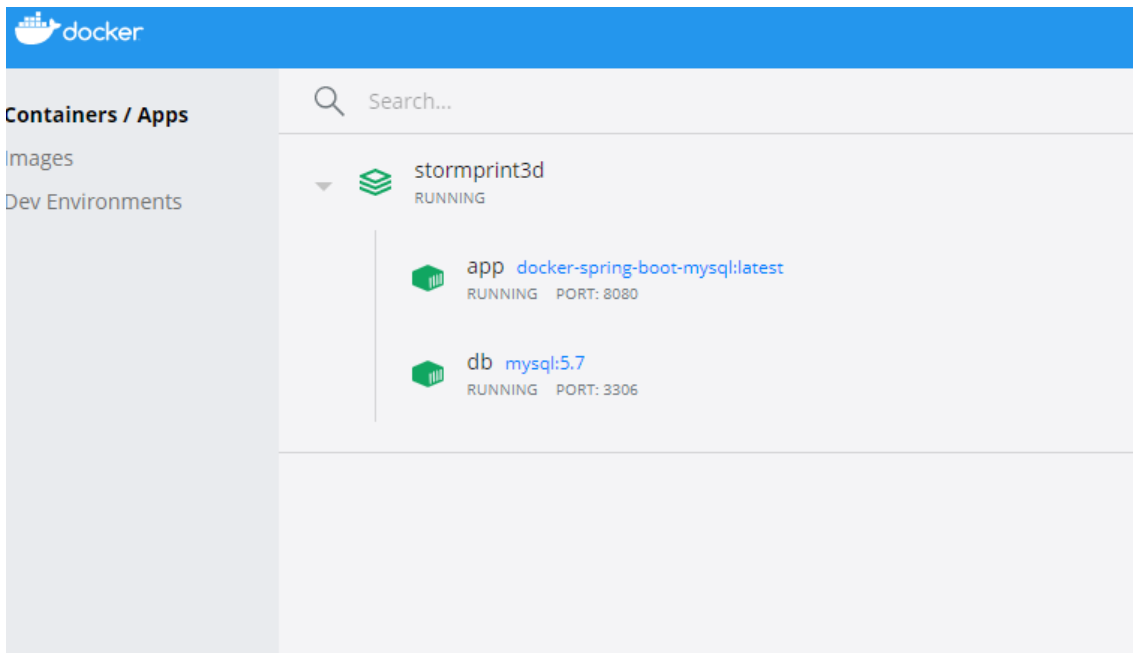
Cuando hayamos hecho este proceso ejecutamos **docker-compose up --build**

```
C:\Users\juan\Documents\Workspace\Prueba\StormPrint3D>docker-compose up --build
Docker Compose is now in the Docker CLI, try 'docker compose up'

Building app
[*] Building 0.7s (7/7) FINISHED
=> [internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 32B 0.0s
=> [internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [internal] load metadata for docker.io/library/openjdk:8-jdk-alpine 0.6s
=> [internal] load build context 0.0s
=> => transferring context: 88B 0.0s
=> [1/2] FROM docker.io/library/openjdk:8-jdk-alpine@sha256:94792824df2df33402f201713f932b58cb9de9a40cd524164a0f 0.0s
=> CACHED [2/2] COPY target/StormPrint3D-0.0.1-SNAPSHOT.jar app.jar 0.0s
=> exporting to image 0.0s
=> exporting layers 0.0s
=> writing image sha256:06ceef29f181cd441f7bc74979aa4491b4841f16b596418296db2f3bc81e9f 0.0s
=> naming to docker.io/library/docker-spring-boot-postgres:latest 0.0s

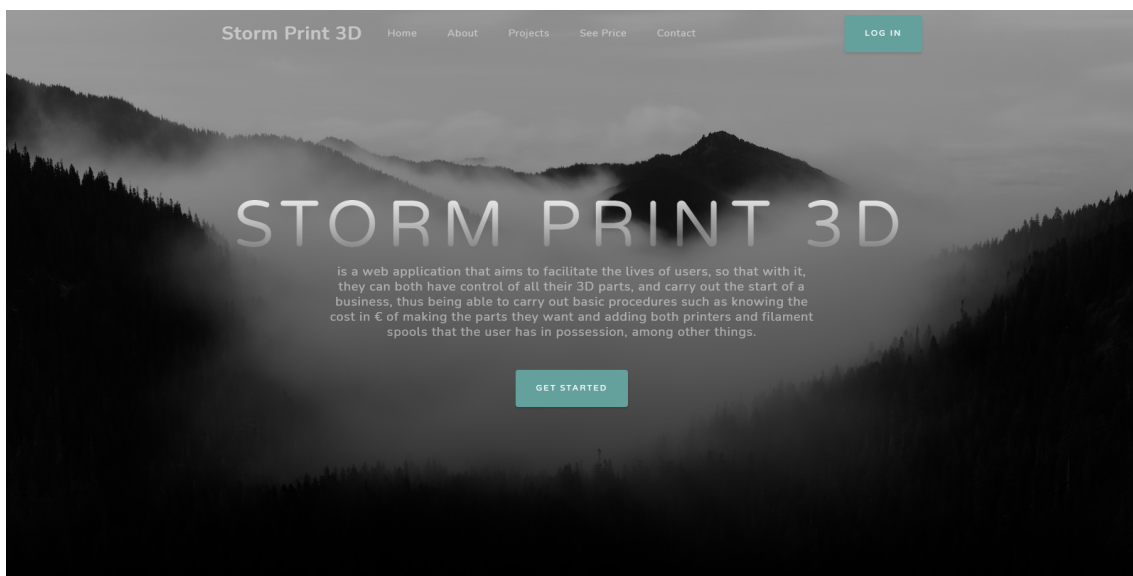
Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
Starting db ... done
Starting app ... done
Attaching to db, app
db      | 2021-06-14 18:12:54:00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 5.7.34-1debian10 started.
db      | 2021-06-14 18:12:54:00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
db      | 2021-06-14 18:12:54:00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 5.7.34-1debian10 started.
db      | 2021-06-14 18:12:54:45099272 [Warning] TIMESTAMP with implicit DEFAULT value is deprecated. Please use --explicit_defaults_for_timestamp server option (see d
db      | 2021-06-14 18:12:54:4516972 [Note] mysqld (mysqld 5.7.34) starting as process 1 ...
db      | 2021-06-14 18:12:54:4533992 [Note] InnoDB: PUNCH HOLE support available
db      | 2021-06-14 18:12:54:4534322 [Note] InnoDB: Mutexes and rw_locks use GCC atomic builtins
db      | 2021-06-14 18:12:54:4534372 [Note] InnoDB: Uses event mutexes
db      | 2021-06-14 18:12:54:4534392 [Note] InnoDB: GCC builtin __atomic_thread_fence() is used for memory barrier
db      | 2021-06-14 18:12:54:4534412 [Note] InnoDB: Compressed tables use zlib 1.2.11
db      | 2021-06-14 18:12:54:4534422 [Note] InnoDB: Using Linux native AIO
db      | 2021-06-14 18:12:54:4535642 [Note] InnoDB: Number of pools: 1
db      | 2021-06-14 18:12:54:4536602 [Note] InnoDB: Using CPU crc32 instructions
db      | 2021-06-14 18:12:54:4545102 [Note] InnoDB: Initializing buffer pool, total size = 128M, instances = 1, chunk size = 128M
db      | 2021-06-14 18:12:54:4588432 [Note] InnoDB: Completed initialization of buffer pool
db      | 2021-06-14 18:12:54:4601072 [Note] InnoDB: If the mysqld execution user is authorized, page cleaner thread priority can be changed. See the man page of setp
db      | 2021-06-14 18:12:54:4718772 [Note] InnoDB: Highest supported file format is Barracuda.
db      | 2021-06-14 18:12:54:4723742 [Note] InnoDB: Log scan progressed past the checkpoint lsn 13011322
db      | 2021-06-14 18:12:54:4724002 [Note] InnoDB: Doing recovery: scanned up to log sequence number 13011331
db      | 2021-06-14 18:12:54:4724052 [Note] InnoDB: Database was not shutdown normally!
db      | 2021-06-14 18:12:54:4724062 [Note] InnoDB: Starting crash recovery.
db      | 2021-06-14 18:12:54:5830932 [Note] InnoDB: Removed temporary tablespace data file: "ibtmp1"
db      | 2021-06-14 18:12:54:5831252 [Note] InnoDB: Creating shared tablespace for temporary tables
db      | 2021-06-14 18:12:54:5831612 [Note] InnoDB: Setting file './ibtmp1' size to 12 MB. Physically writing the file full; Please wait ...
db      | 2021-06-14 18:12:54:5953992 [Note] InnoDB: file './ibtmp1' size is now 12 MB.
db      | 2021-06-14 18:12:54:5960372 [Note] InnoDB: 96 redo rollback segment(s) found. 96 redo rollback segment(s) are active.
db      | 2021-06-14 18:12:54:5960642 [Note] InnoDB: 32 non-redo rollback segment(s) are active.
db      | 2021-06-14 18:12:54:5967012 [Note] InnoDB: 5.7.34 started; log sequence number 13011331
db      | 2021-06-14 18:12:54:5969622 [Note] InnoDB: Loading buffer pool(s) from ./var/lib/mysql/ib_bufferpool
db      | 2021-06-14 18:12:54:5972272 [Note] Plugin 'FEDERATED' is disabled.
db      | 2021-06-14 18:12:54:5988802 [Note] InnoDB: Buffer pool(s) load completed at 210614 18:12:54
```

pág. 6



Si todo ha salido correctamente, nuestra aplicación junto a las bases de datos deberían estar funcionando.

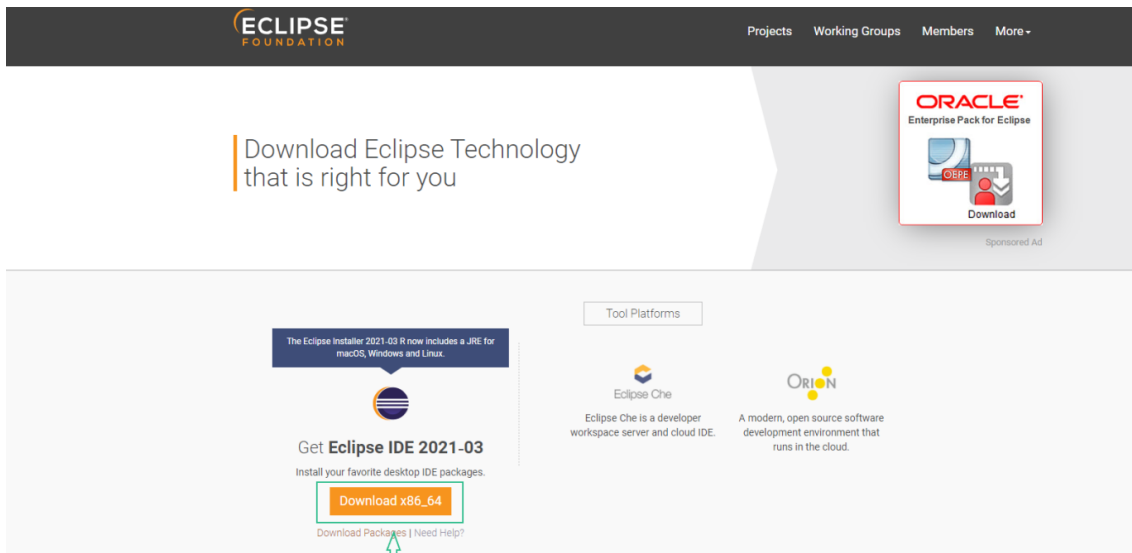
Con el proyecto iniciado ya podremos entrar en nuestra aplicación desde la dirección <http://localhost:8080/>



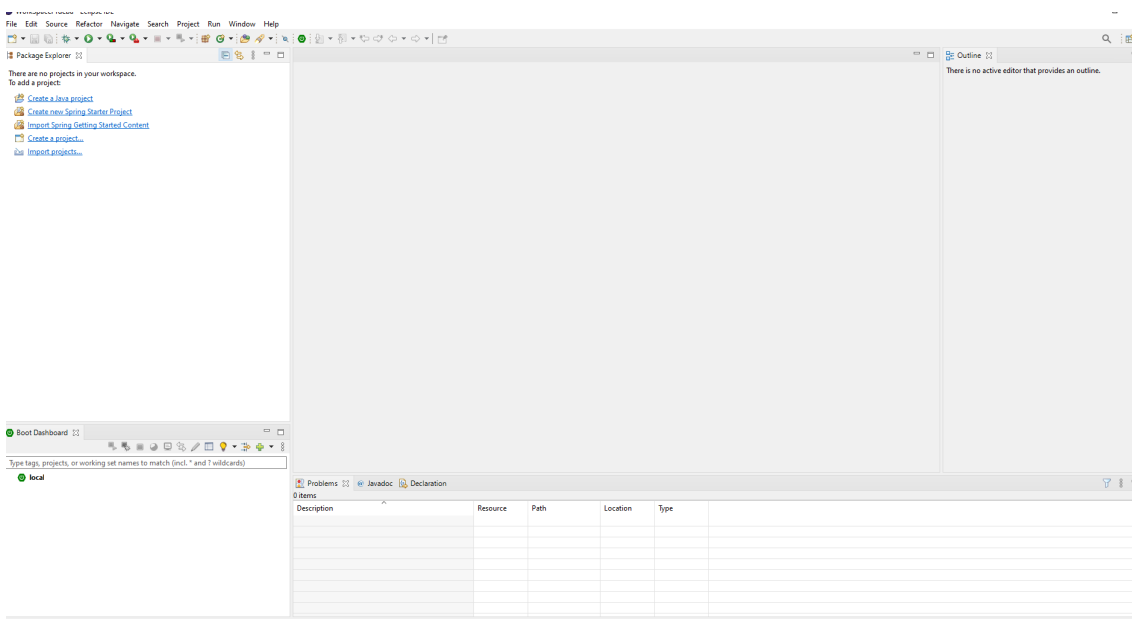
## Si queremos ver el código o editar algo

Primero tendremos que descargar Eclipse desde este [enlace](#).



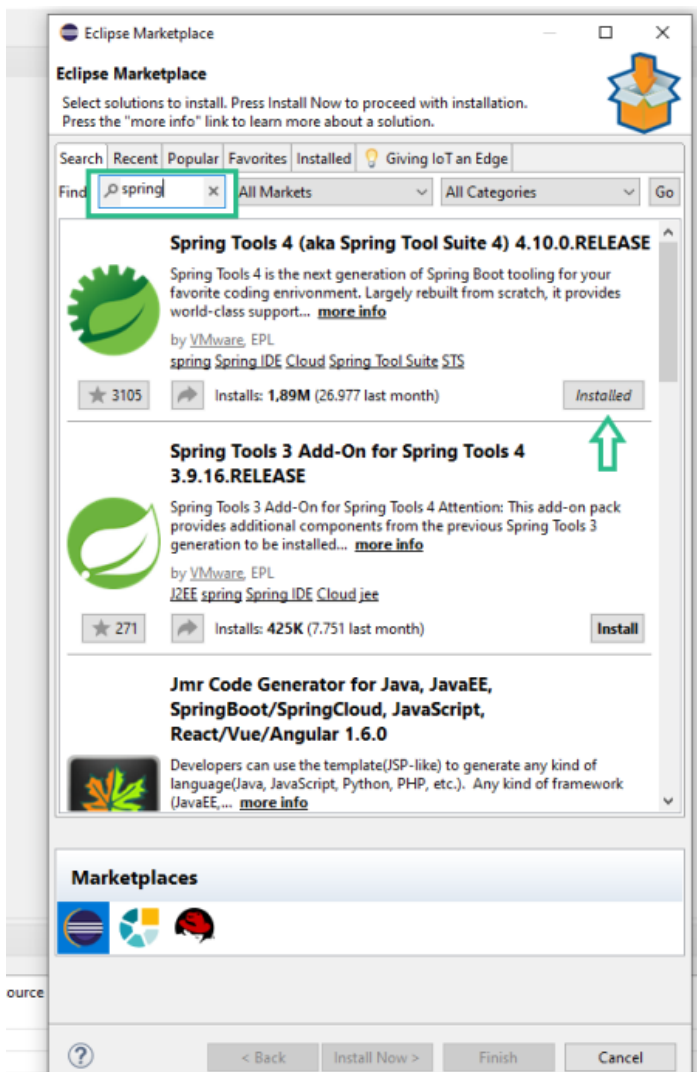
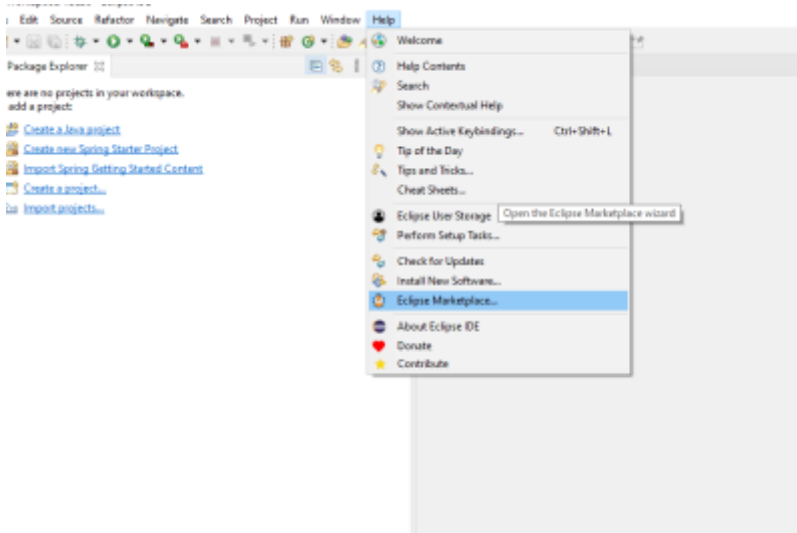


Una vez esté descargada e instalada Eclipse en nuestro equipo, la abrimos y se verá de esta manera.

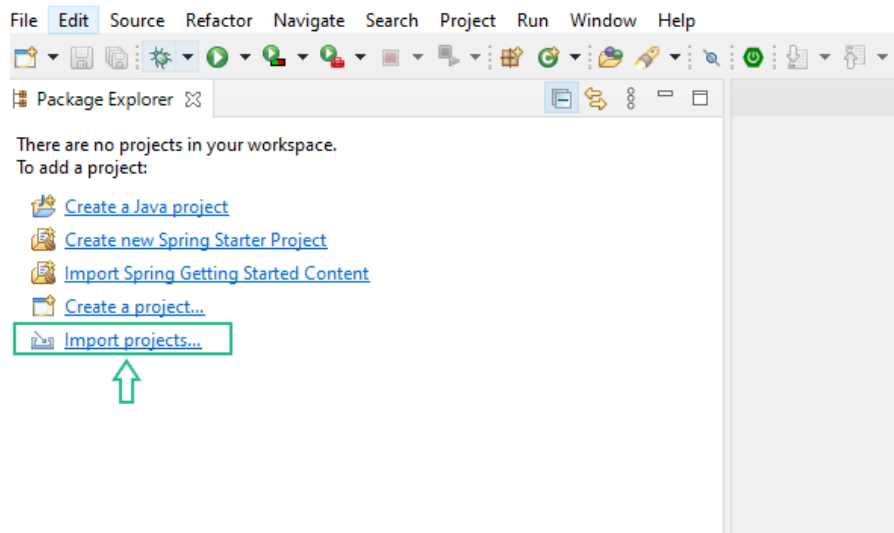


Ahora tendremos que proceder a instalar una herramienta para poder usar y editar proyectos realizados en Spring Boot, para ello seguiremos estos pasos:

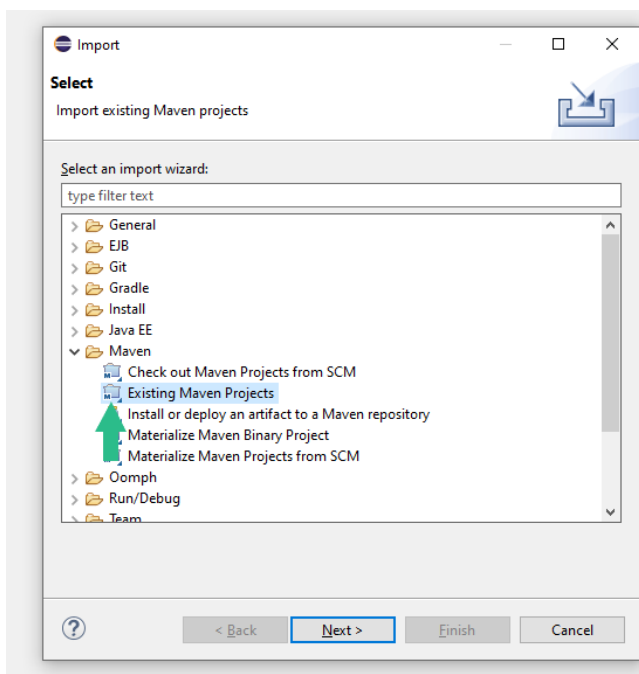
Help / Eclipse Marketplace / buscar spring boot / instalar herramienta.



Una vez que tengamos instalada Spring Tools en nuestro eclipse podemos proceder a editar o correr nuestro proyecto desde Eclipse, procedemos a importar el proyecto.

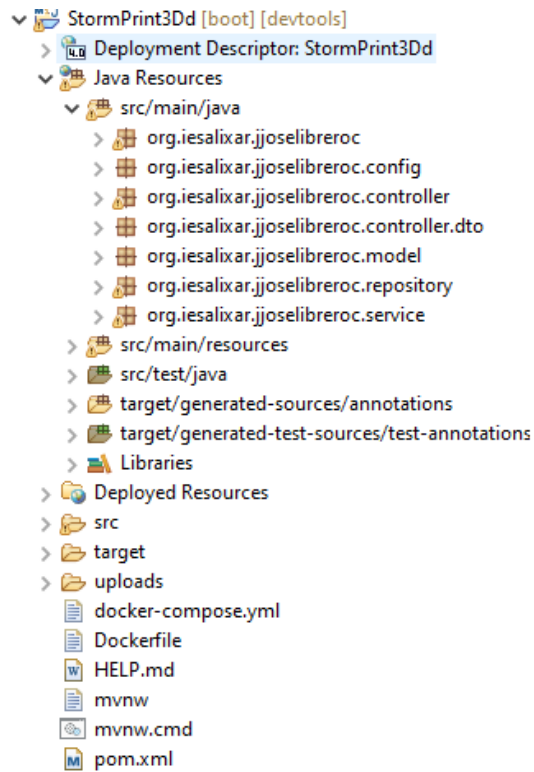


Abrimos la pestaña de Maven y le damos a proyecto existente de maven.



Ahora seleccionamos la ruta donde se encuentra lo que descargamos anteriormente de GitHub, abrimos la carpeta src y seleccionamos la carpeta que pone StormPrint3D y le damos a importar.

Se cargará el proyecto y nos saldrá a la izquierda con el nombre StormPrint3D, lo abrimos y ya podemos ver el código o modificarlo.

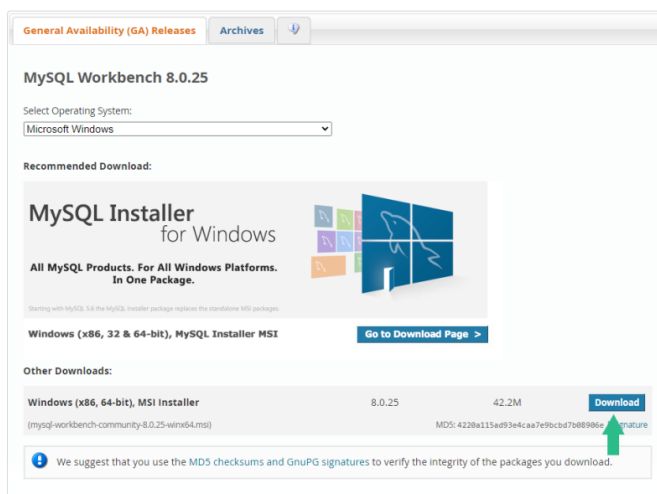


## Si queremos ver la base de datos

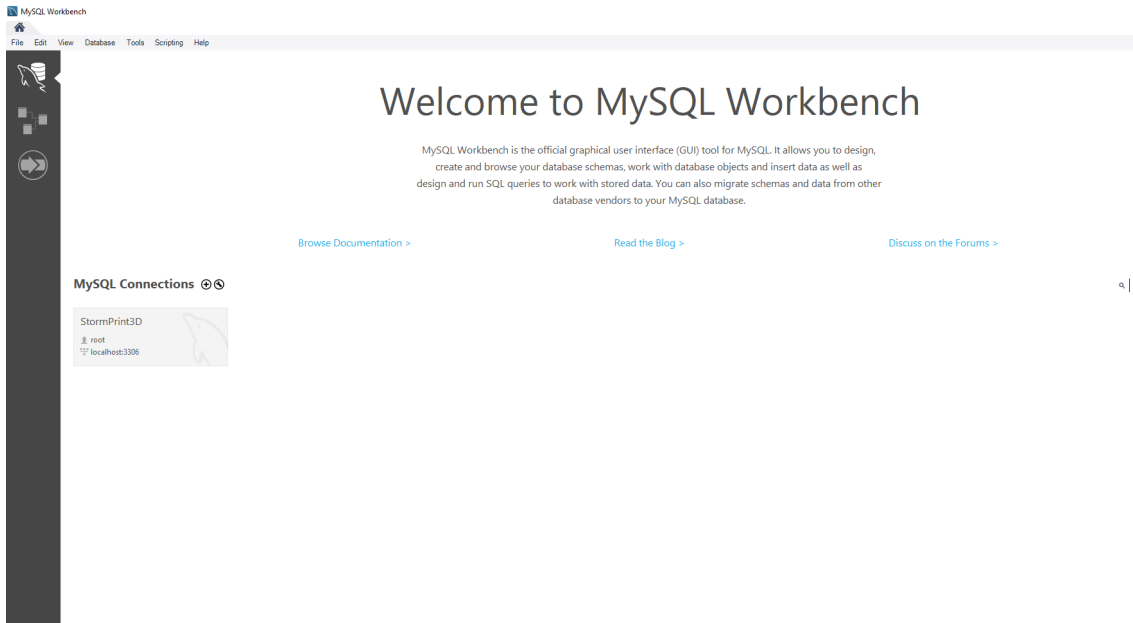
Primero tenemos que descargarnos MySQL Workbench desde este [enlace](#).

### MySQL Community Downloads

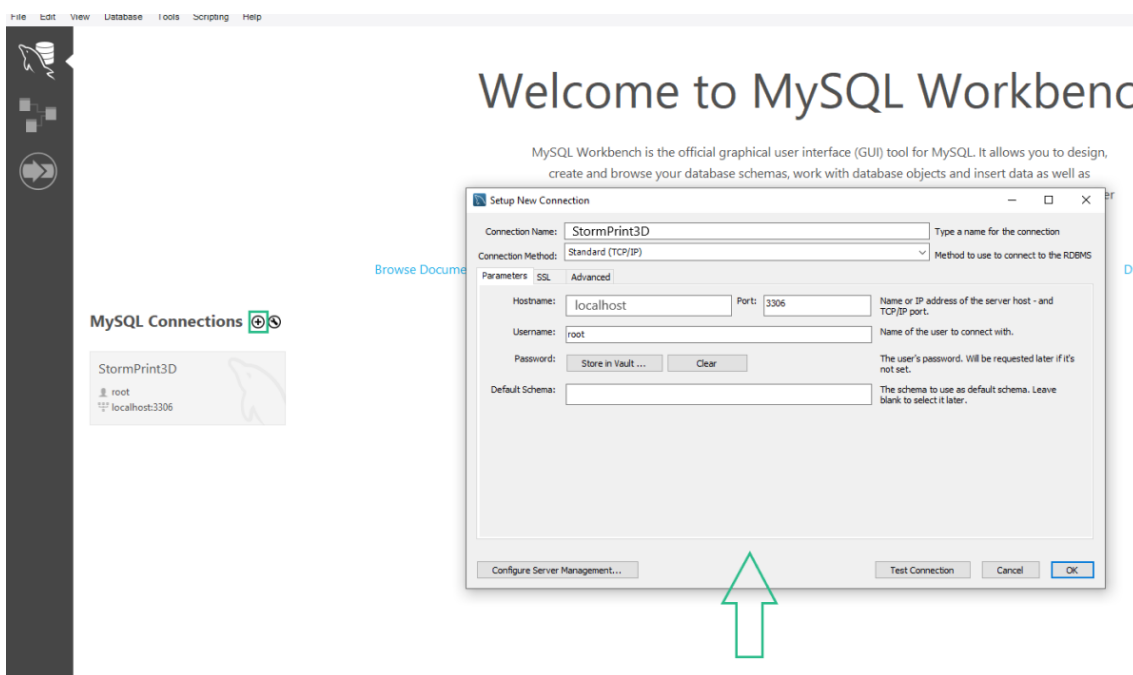
MySQL Workbench



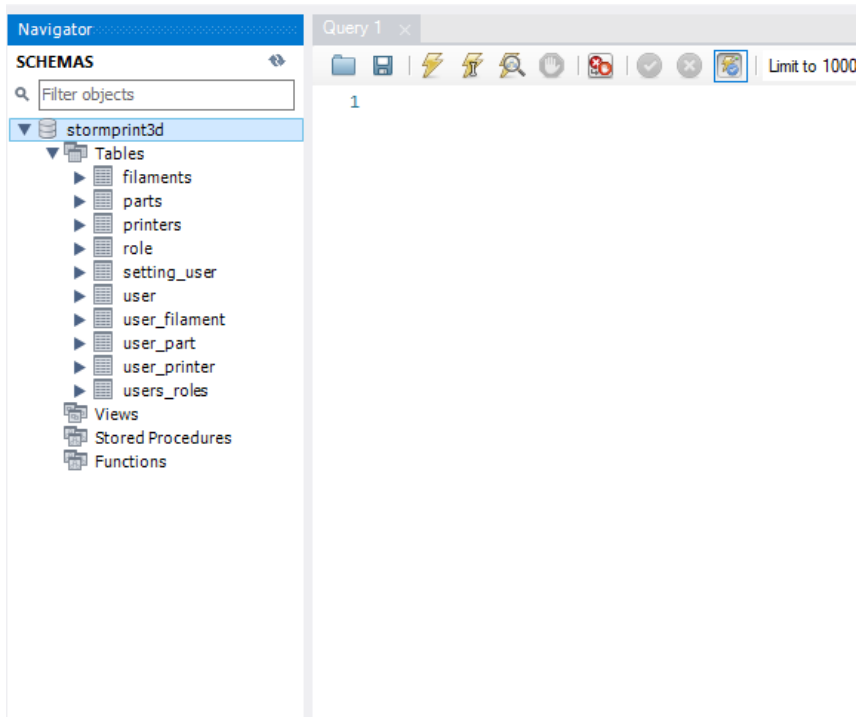
Una vez descargado e instalado lo abrimos y se verá así.



Os saldrá sin esa conexión, es lo que tenemos que crear ahora, así que le damos al + y vamos a crear una nueva conexión.



Tendremos que ponerle nombre a la conexión → **StormPrint3D**, luego el hostname que pondremos **localhost**, el puerto por defecto **3306**, usuario **user** y contraseña **password**. Le damos a aceptar y se creará una conexión idéntica a la que sale en la foto a la izquierda. Tras esto tendremos que tener el proyecto levantado, como se explica en el paso [anterior](#). Cuando esté levantado podremos acceder a la base de datos, pinchamos sobre la conexión creada y se verá algo así.



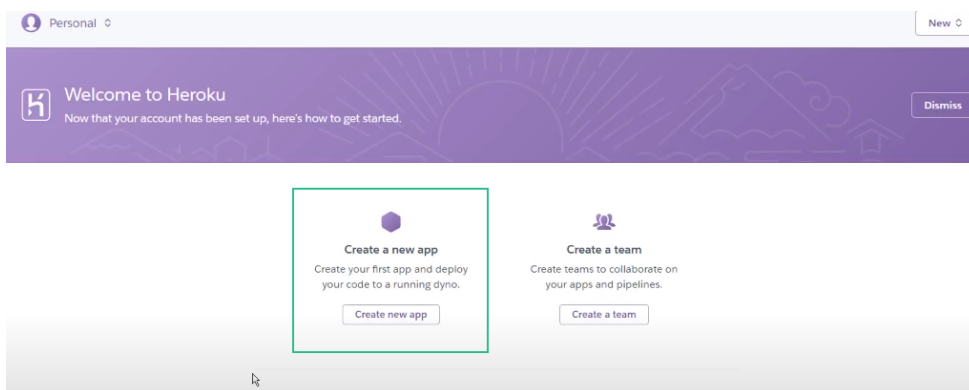
## Desplegar un proyecto en heroku

Lo primero es crearnos una cuenta en [heroku](https://heroku.com).

Importante poner de lenguaje Java, ya que la aplicación que vamos a desplegar está construida en Java. Luego nos llegará un correo para confirmar la cuenta y crear la contraseña de nuestra cuenta.



Una vez tengamos la cuenta, nos saldrá una pestaña con la opción de crear una nueva app, le damos.



y nos saldrá esto:



## Create New App

App name

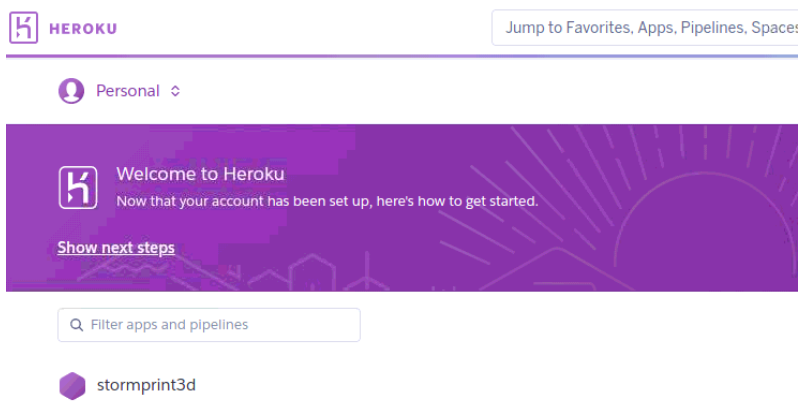
Choose a region

[Add to pipeline...](#)

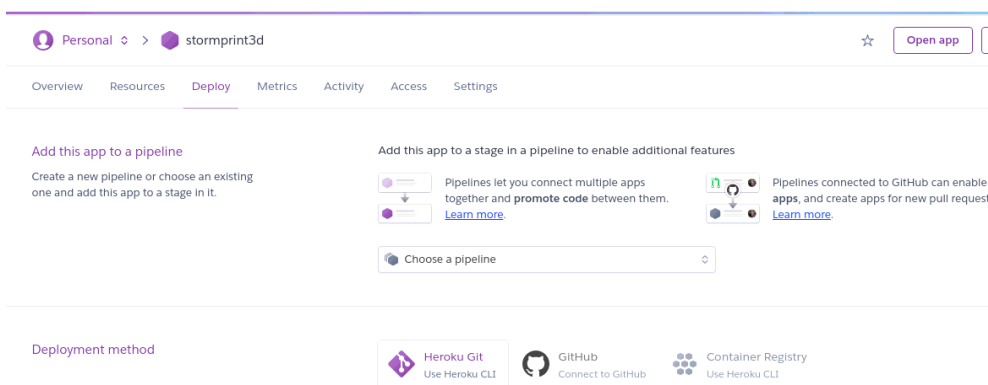
[Create app](#)

Añadimos nombre de la aplicación y la región y le damos a intro, sin añadir pipeline.

Una vez que se haya creado la aplicación, nos vamos a la pestaña principal de heroku, y nos saldrá la app creada.



Pinchamos sobre ella y entraremos en la app que hemos creado.



Ahora necesitamos crear una base de datos para guardar los datos que se vayan registrando en la app. Para ello nos vamos a la pestaña resources.





Personal > stormprint3d

Overview Resources Deploy Metrics Activity Access Settings

Free Dynos [Change Dyno Type](#)

web java -Dserver.port=\$PORT \$JAVA\_OPTS -jar target/StormPrint3D-0.0.1-SNAPSHOT.jar \$0.00

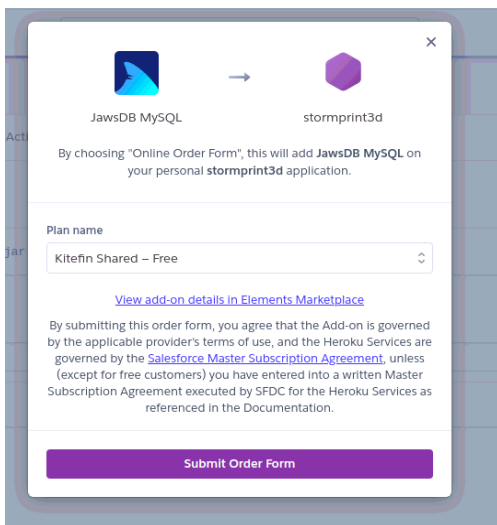
Add-ons [Find more add-ons](#)

Quickly add add-ons from Elements

**JawsDB MySQL** Attached as JAWSDB Kitefin Shared Free

Estimated Monthly Cost \$0.00

Añadimos en add-ons mysql y nos saldrá la que tenemos ahí ya desplegada.



Le damos a añadir, pero nos pedirá que añadamos una cuenta de crédito/débito para poder añadir una base de datos distinta a postgresql, es solo añadir la tarjeta, no se paga nada, para ello tenemos que irnos a account setting que está en la foto de perfil de la derecha de la pantalla y nos saldrá esta pestaña, luego le damos a billing, y ahí añadimos la tarjeta. Cuando esté registrada, volvemos a resources y hacemos los mismos pasos, comprobarás que ya se puede añadir la base de datos mysql a nuestro proyecto de heroku.

HEROKU Jump to Favorites, Apps, Pipelines, Spaces...

Manage Account

Account Applications Billing

**Profile**  
Your email address is your identity on Heroku and is used to log in.

**Email Address**  
alum.jjosellbreroc@lesalixar.org

**Name (Optional)**  
Juan José Librero  
Manage your avatar using [Gravatar](#)

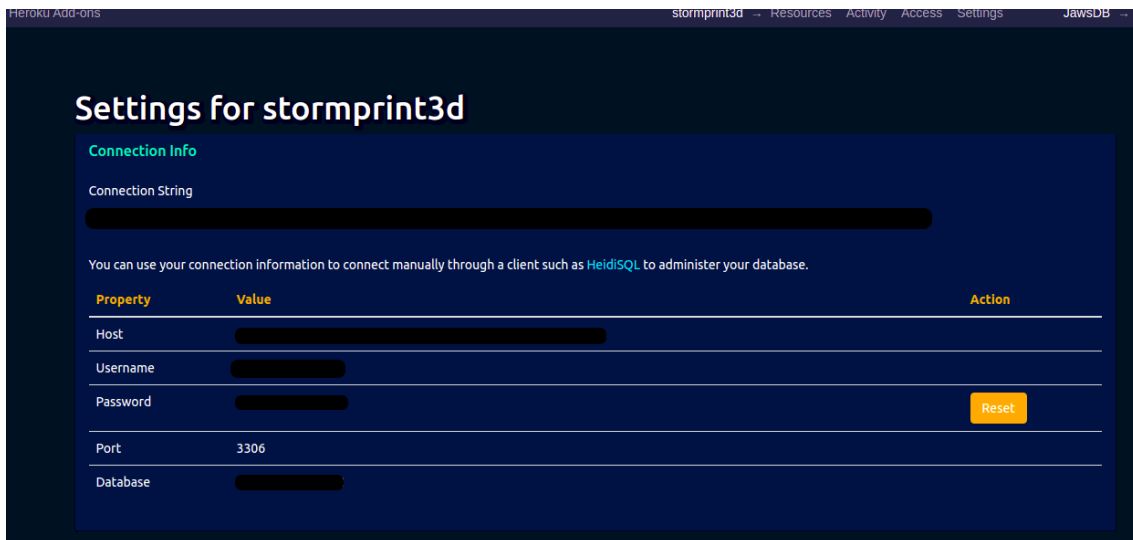
**Password**  
Changing your password will also reset your API key.

**Current Password**  
enter your current password

**New Password**



Ahora tenemos que pinchar sobre la base de datos mysql y se nos debe abrir una pestaña con la información de la base de datos y la conexión.



Lo necesitamos para configurar la conexión del proyecto con esta base de datos. Abrimos nuestro proyecto con Eclipse, abrimos la carpeta /src/main/resources/ y abrimos el archivo application.properties, en el cual veremos algo como esto.

```

1  server.port=8080
2
3  #server.port=${PORT:8080}
4  #spring.datasource.url=jdbc:mysql://....:3306/.....?useSSL=true
5  #spring.datasource.username=...
6  #spring.datasource.password=...
7  #spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
8  #spring.jpa.database-platform=org.hibernate.dialect.MySQL57Dialect
9  #spring.jpa.generate-ddl=true
10 #spring.jpa.hibernate.ddl-auto=update
11 #logging.level.org.hibernate.SQL=debug
12
13 ##Para tamaño maximo de la subida de la img de perfil de cliente
14 #spring.servlet.multipart.max-file-size = 10MB
15 #spring.servlet.multipart.max-request-size = 10MB

```

Deberemos de descomentar estas líneas e ir cogiendo la información de la base de datos y añadirla a este archivo, como en el username, en la password, y luego en el datasource va → en los primeros '...' va el host, y en los segundos '...' va la database. Cuando tengamos este archivo terminado estamos listos para subir nuestro proyecto a heroku.

Para ello volvemos a la plataforma de heroku y le damos a la pestaña de deploy.



HEROKU

Jump to Favorites, Apps, Pipelines, Spaces...

Personal > stormprint3d

Overview Resources **Deploy** Metrics Activity Access Settings

**Add this app to a pipeline**  
Create a new pipeline or choose an existing one and add this app to a stage in it.

**Add this app to a stage in a pipeline to enable additional features**

Pipelines let you connect multiple apps together and **promote code** between them. [Learn more.](#)

Pipelines connected to Git apps, and create apps for r [Learn more.](#)

Choose a pipeline

**Deployment method**

**Heroku Git**  
Use Heroku CLI

GitHub  
Connect to GitHub

Container Registry  
Use Heroku CLI

Tenemos que tener seleccionado Heroku Git, luego hacemos scroll y nos saldrá los pasos a seguir para hacer un deploy de nuestro proyecto en heroku, lo primero es descargarse la consola de heroku, [heroku cli](#).

Una vez esté descargada, hacemos heroku login en nuestra cmd o terminal, según en que sistema operativo estemos. Nos pedirá permisos en el navegador web para logearnos, le damos y volvemos al terminal.

```
jjlc-daw@JJLC:~$ heroku login
> Warning: heroku update available from 7.52.0 to 7.54.0.
heroku: Press any key to open up the browser to login or q to exit:
Opening browser to https://cli-auth.heroku.com/auth/cli/browser/383566b4-8d8e-
JuBgB0c_kTegFiAAFRgA._32jlibbc4QT8lMqzJ7Ln4ARqs8WlWzRhPn4FDdL44s
Logging in... done
Logged in as [redacted]
jjlc-daw@JJLC:~$
```

Ahora entramos en la carpeta donde se encuentra nuestro proyecto Storm Print3D, cuando estemos en esa ruta, ponemos estos 3 comandos en este orden.

```
$ git add .
$ git commit -am "make it better"
$ git push heroku master
```

El último paso finaliza dándote la url de tu app en heroku, pinchas y ya está tu app desplegada y funcionando.