

# Manual de Implantación

Aplicación Descanso Royal

---

José María de Juan Vázquez

2º DAW 2018/19

24/06/2019

<b>Introducción</b>	<b>2</b>
<b>Base de datos</b>	<b>3</b>
<b>Backend - Tecnologías</b>	<b>5</b>
<b>Frontend - Tecnologías</b>	<b>7</b>
<b>Puesta en marcha del proyecto - Local</b>	<b>8</b>
<b>Puesta en marcha del proyecto - Servidor</b>	<b>11</b>
Crear y configurar el servidor - AWS	11
Conectar con el servidor - PUTTY, Filezilla y SQL Workbench	14
Servidor - Instalaciones y despliegue	20
<b>Bibliografía</b>	<b>22</b>

## Introducción

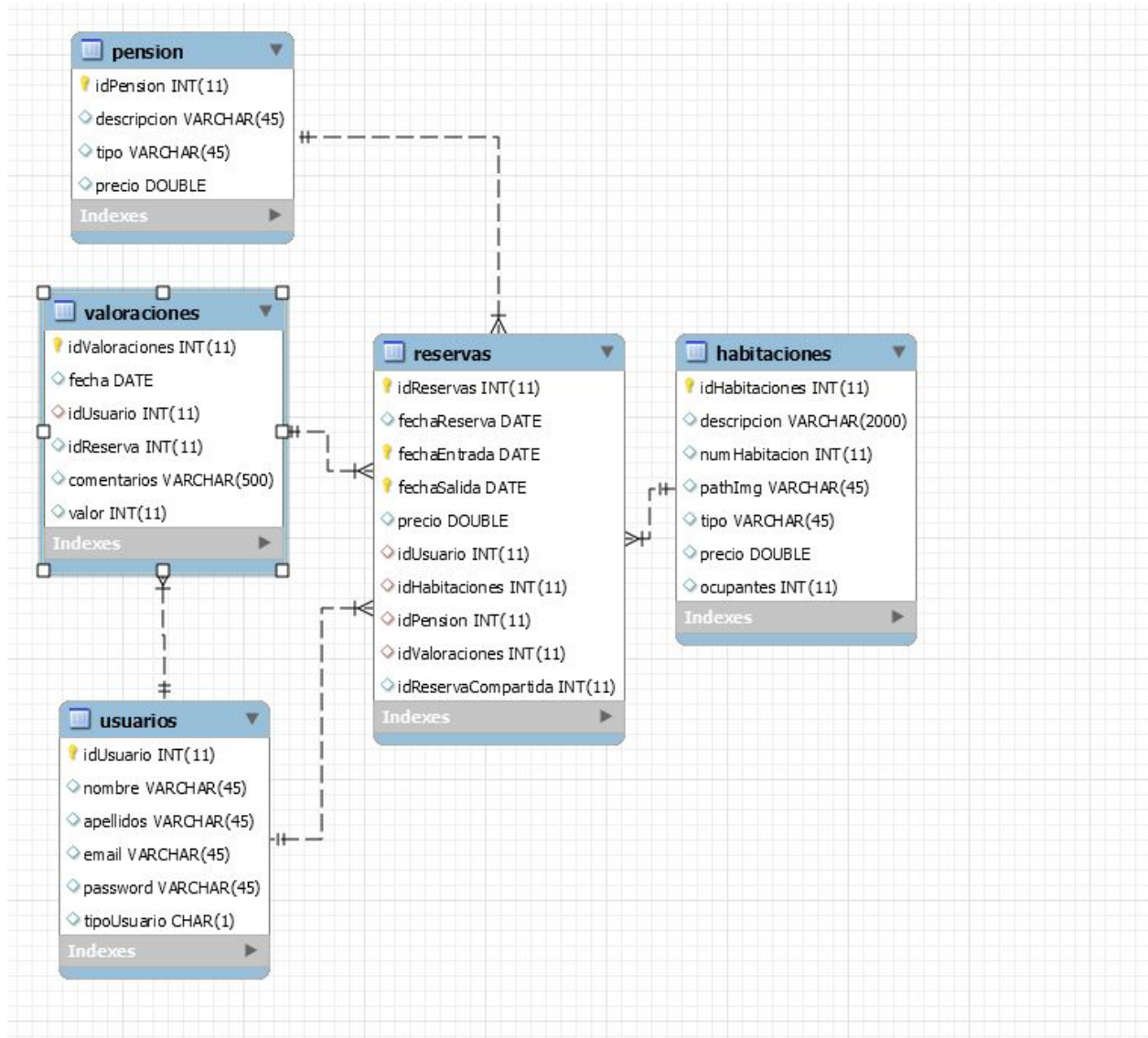
El siguiente manual explica las tecnologías y la estructura elegidas para la realización de este proyecto, así como su puesta en marcha tanto en entornos locales como en entornos remotos.

A modo informativo previa explicación, la base de datos elegida ha sido un SQL Server, el backend se ha realizado en Spring Boot junto con JPA e Hibernate, y el frontend se ha realizado siguiendo una plantilla en Bootstrap HTML5 con Javascript y JQuery. Para el servidor remoto se ha usado un Ubuntu Server 16.04 alojado en una instancia EC2 de Amazon.

Aunque el dominio se ha comprado y configurado, dado que no se ha conseguido hacer que funcione, no se incluye en este manual. No obstante, se referenciará en el apartado bibliográfico.

## Base de datos

La base de datos elegida ha sido SQL Server, y posee la siguiente estructura:



De esta forma podemos observar varias situaciones:

- Un usuario realiza 1 o más reservas, y puede realizar 1 o más valoraciones.

- Una valoración es aplicable a una reserva, haciendo una relación 1-1. Una reserva no tendrá más de una (1) valoración. Para que esto sea cierto, se toman medidas en el frontal que previenen a un usuario de valorar una reserva más de una vez.
- Desligada la pensión de la habitación, ya que provocaría que cada habitación tuviese que ser multiplicada por el número de pensiones. De esta forma, la pensión es independiente de la habitación y el cálculo se realiza conforme a la reserva.

Además, en caso de que no tuviésemos un tipo de pensión por habitación y tuviéramos que incluirlo a mano, estaríamos constantemente modificando el campo de pensión de las habitaciones, lo cual, bajo mi punto de vista, es poco práctico: tendría más sentido ligar la pensión a la reserva, ya que una persona alquila una habitación y como servicio adicional puede tener una pensión, es decir, por separado.

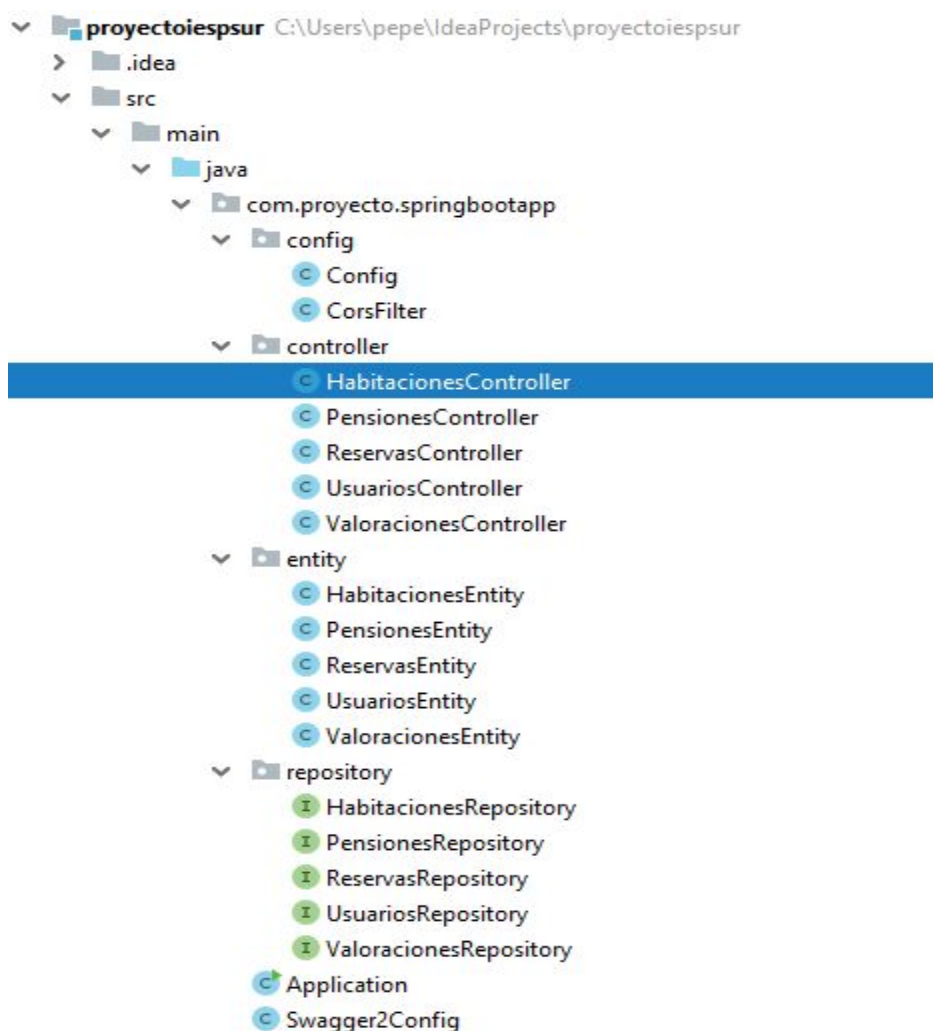
- Dentro de la reserva encontramos el campo `idReservaCompartida`, el cual nos permite agrupar las reservas de habitaciones en una sola reserva sin generar una tabla. De esta forma podemos obtener el total de habitaciones incluidos en una reserva de un usuario, así como todas las habitaciones que se han reservado.
- No tener una pensión está considerado como contratar un servicio, a coste 0 (según se recoge dentro de los datos de la tabla `pension`). No obstante, aunque es posible no tener una pensión en BBDD, se han tomado medidas para que en el aplicativo se deba tener un tipo de pensión.

## Backend - Tecnologías

El backend a realizar se ha decidido que sea Spring Boot con JPA e Hibernate. Si bien es cierto que estas tecnologías como tal no se han impartido en el módulo (a excepción de Spring, salvo que se ha visto Spring MVC), considero importante hacer uso de ellas ya que son las tecnologías que uso en la FCT. Si bien la curva de aprendizaje es alta y se tarda tiempo en dominar dichas tecnologías, se realizará un Backend no muy complejo.

Cabe destacar que a lo largo de las clases del Backend encontraremos anotaciones, estas anotaciones son una mejora que sustituyen los archivos .xml Si bien son propias de Spring, también podemos encontrar varias anotaciones (especialmente en las clases Entity) que pertenecen a Hibernate: éstas delimitan el tipo de dato a representar en el esquema de base de datos.

La estructura del proyecto Spring boot es la siguiente:



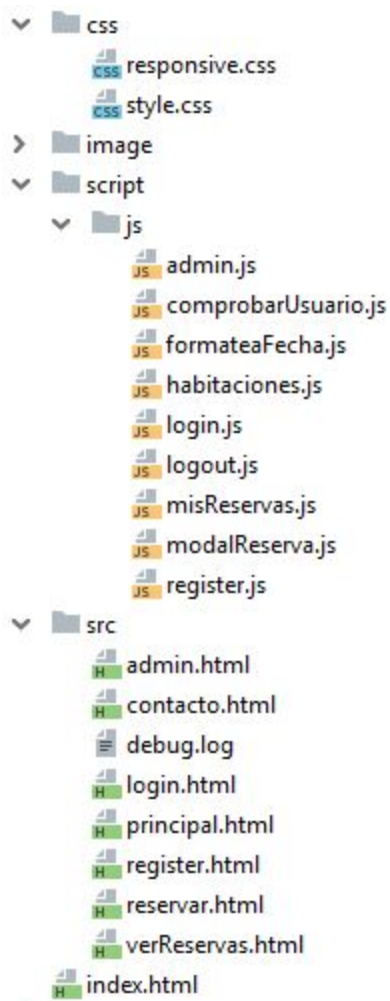
- El paquete config incluye las configuraciones necesarias para la puesta en marcha del proyecto, tanto para la conexión de BBDD como para la administración de las entidades. Adicionalmente se ha necesitado administrar las peticiones entrantes y salientes, dado que backend y frontend están ubicados en el mismo dispositivo.
- El paquete controller incluye los controladores del proyecto, routeando las salidas por las cuales se puede acceder a los servicios que presta el aplicativo.
- El paquete entity hace alusión a las entidades, que en este proyecto son la representación JAVA de las tablas y sus relaciones de la BBDD.
- El paquete repository hace alusión a todas las funcionalidades que podemos encontrar dentro de la aplicación, o dicho de otra manera, las consultas que nuestro aplicativo backend puede realizar contra la BBDD.
- La clase Application es el análogo a la clase Main en JAVA, incluye una clase Main que inicia el proyecto.
- La clase Swagger2Config añade la configuración de la herramienta de Testing "Swagger", que permite comprobar la salud de los Endpoints de nuestro proyecto backend. Dispone de un entorno gráfico.

## Frontend - Tecnologías

Para el frontal se ha decidido utilizar HTML5 con CSS y Javascript, con la librería JQuery.

Se ha dispuesto de una plantilla que se ha utilizado a modo de guía para la realización del apartado gráfico del frontal, mientras que con Javascript se han realizado todas las funcionalidades.

La estructura es la siguiente:



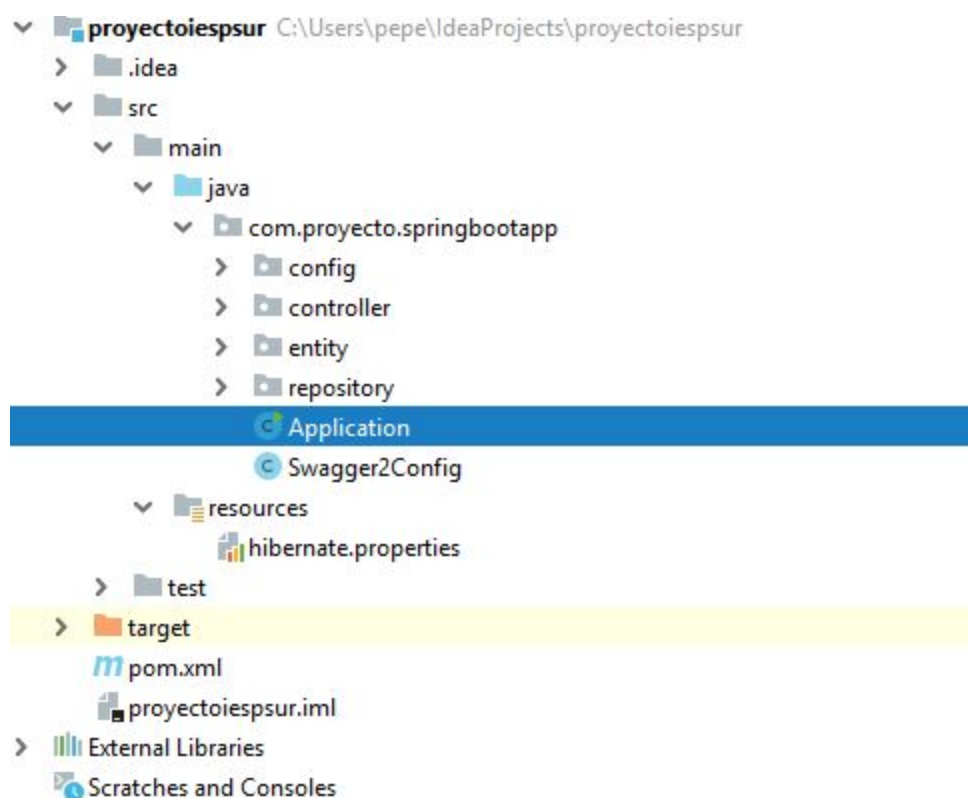
Para la creación de los archivos HTML y JS se ha utilizado el editor Sublime Text.



## Puesta en marcha del proyecto - Local

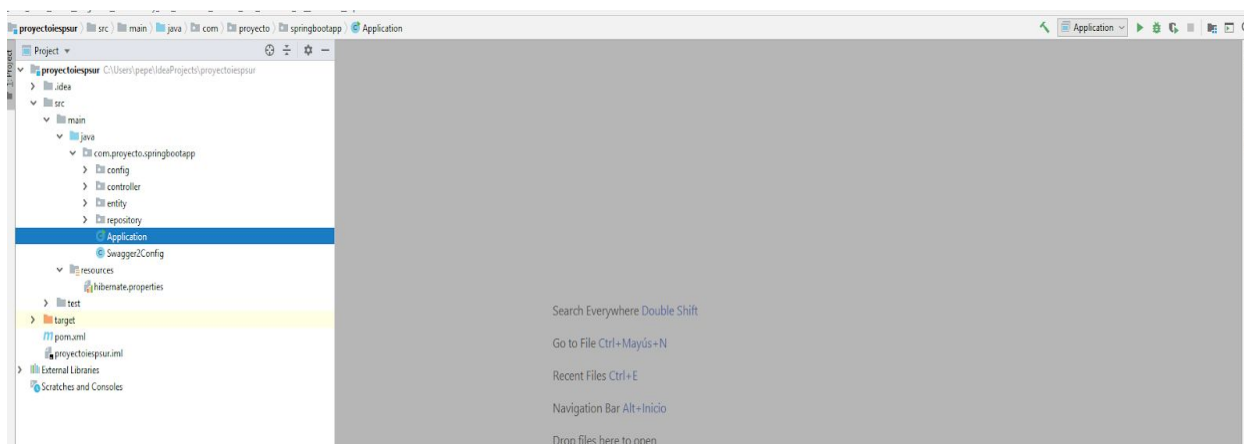
### Backend - IntelliJ

Dado que el IDE a usar es IntelliJ (versión Community, no tiene soporte con Spring Boot), para poder iniciar el Backend necesitaremos tener claro cuál es la clase que incluye el Main, en nuestro caso la clase Application.java. Una vez la localizamos, podemos darle botón derecho y a Run Application.java. No obstante, es recomendable crear una configuración.



Para ello, IntelliJ tiene esa opción a través del menú situado arriba a la derecha (véase que la configuración está creada previa redacción de este documento, no obstante se explicará como configurar la misma).

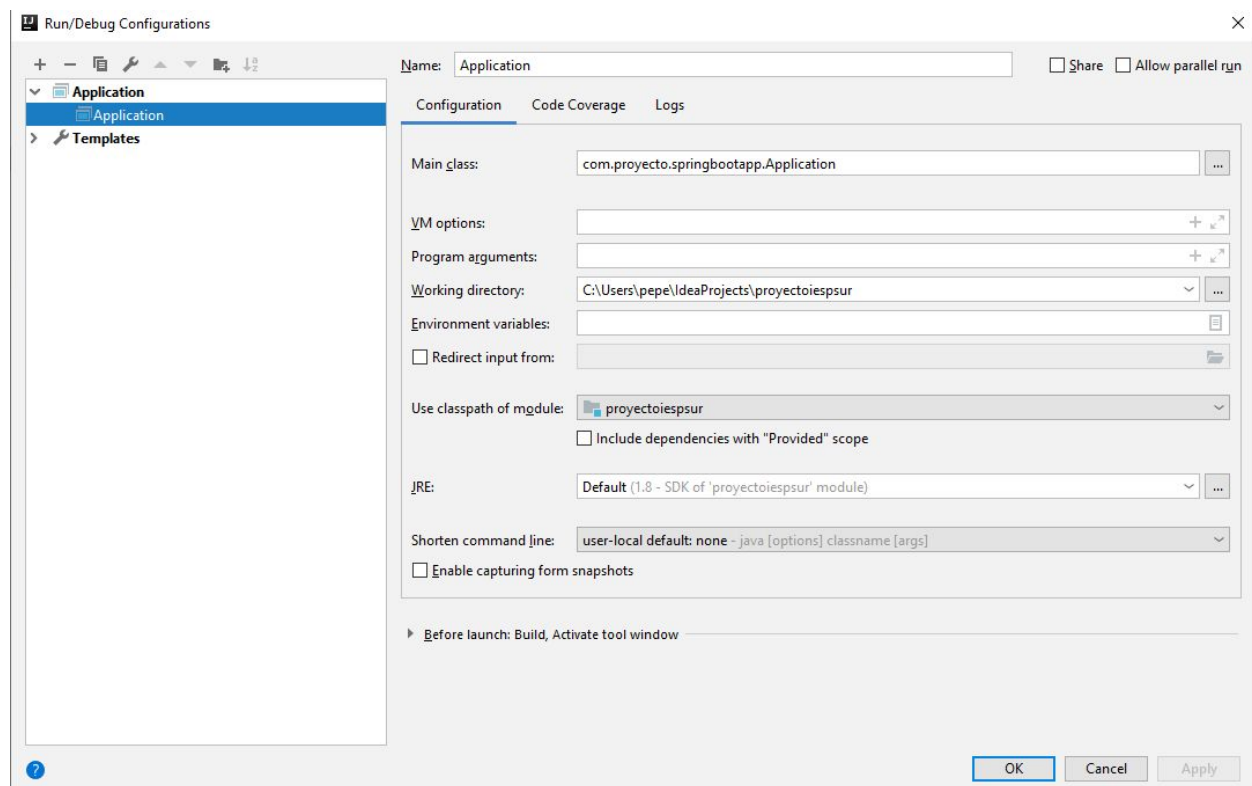
En el cuadro a la derecha del símbolo del martillo, aparecerá un desplegable con todas las configuraciones de inicio que tengamos en nuestro proyecto, y arriba del todo aparecerá Edit Configuration, donde podremos añadir una nueva o modificar una previa.



Una vez accedemos, deberemos tener claro que una de las ventajas de usar Spring Boot con este IDE es que el proyecto Maven ya se “autobuildea” por si solo, por lo que no tendremos previamente que realizar una configuración en Maven para poder “bildear” el proyecto (no obstante, para este proyecto no es indispensable, pero por norma general para proyectos mucho más amplios es altamente recomendable). No obstante, esto es una característica propia del IDE seleccionado, por lo que deberemos “bildear” el proyecto en el servidor remoto.

En este caso, hay 2 campos a tener en cuenta: Main class, aquella clase donde se encuentra el Main (en nuestro caso, Application.java) y VM options, donde podremos especificar, entre otros, el puerto donde queremos que se lance (por defecto, al tener un Tomcat embebido, será en el puerto 8080).

No obstante, abajo del todo, en el desplegable “before launch” podemos especificar los goals que queramos de Maven.



## Frontend y Servidor : WAMP

Para la puesta en marcha del servidor local, se ha usado un Windows 10. Descargando el paquete WAMP, se han copiado todos los elementos del frontal en la carpeta **C:\wamp64\www**.

## Puesta en marcha del proyecto - Servidor

### Crear y configurar el servidor - AWS

En primer lugar, tendremos que crearnos una cuenta en AWS. Este paso se omitirá dado que el equipo educativo ha provisto al alumnado de un portal que ya incluye una cuenta creada.

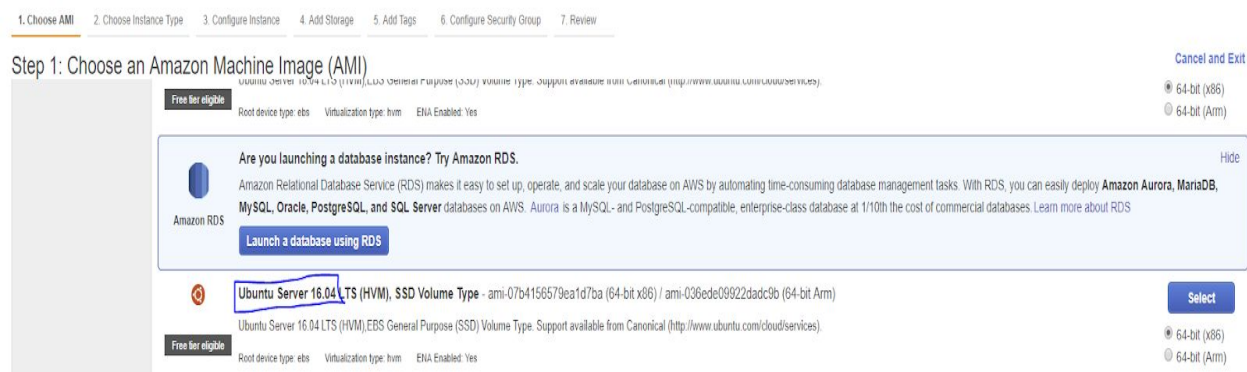
Una vez tenemos acceso al portal AWS de Amazon, tendremos que iniciar una estancia EC2.

The image shows the AWS Management Console interface. The top section is titled 'AWS-Managementkonsole'. Below it, the 'AWS-Services' section is visible, with a search bar and a list of services. The 'Datenverarbeitung' (Data Processing) category is highlighted with a blue box, showing services like EC2, Lightsail, ECR, ECS, EKS, Lambda, Batch, Elastic Beanstalk, and Serverless Application Repository. The 'Entwicklertools' (Developer Tools) category is also visible, showing services like CodeStar, CodeCommit, CodeBuild, CodeDeploy, CodePipeline, Cloud9, and X-Ray. The 'Machine Learning' category shows services like Amazon SageMaker, Amazon Comprehend, AWS DeepLens, Amazon Lex, Machine Learning, Amazon Polly, Rekognition, Amazon Transcribe, Amazon Translate, and Amazon Personalize. The 'Mobil' (Mobile) category shows services like AWS Amplify, Mobile Hub, AWS AppSync, and Device Farm. The 'AR & VR' category shows Amazon Sumerian. The 'Anwendungsintegration' (Application Integration) category is also visible.

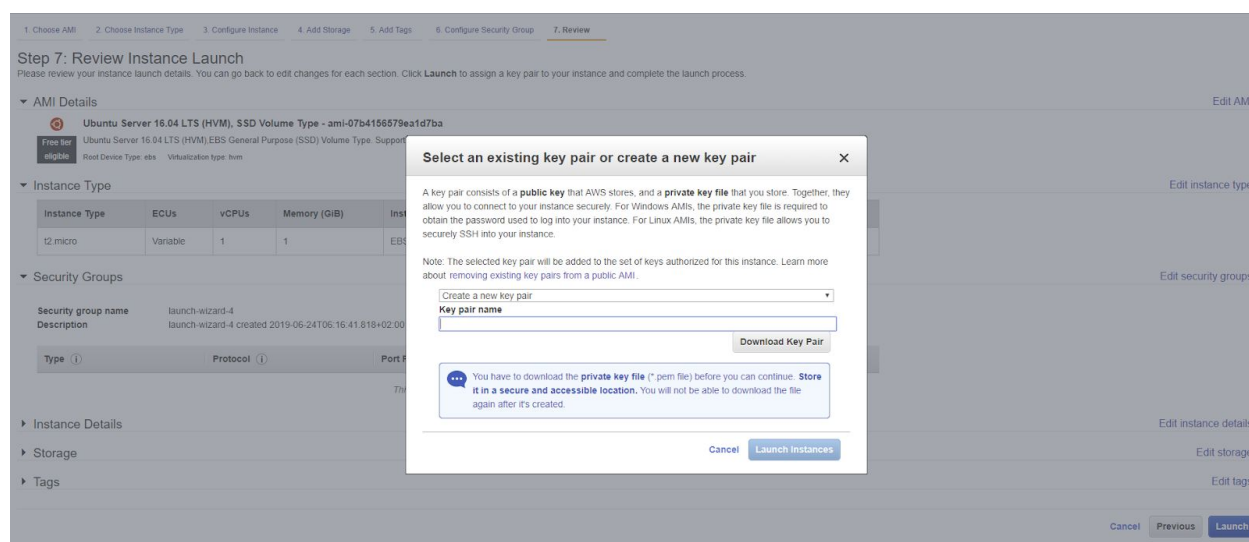
The bottom section of the image shows the 'EC2 Dashboard' with a sidebar on the left containing links to Events, Tags, Reports, Limits, INSTANCES, Launch Templates, Spot Requests, Reserved Instances, Dedicated Hosts, Capacity Reservations, IMAGES, AMIs, Bundle Tasks, ELASTIC BLOCK STORE, Volumes, Snapshots, Lifecycle Manager, NETWORK & SECURITY, Security Groups, and Elastic IPs. The main content area is titled 'Resources' and shows a summary of EC2 resources in the US East (N. Virginia) region: 1 Running Instances, 0 Elastic IPs, 0 Dedicated Hosts, 0 Snapshots, 1 Volumes, 0 Load Balancers, 3 Key Pairs, and 4 Security Groups. Below this, there is a 'Create Instance' button highlighted with a blue box. The 'Create Instance' button is located under the 'Create Instance' section, which includes a 'Launch Instance' button and a 'Note: Your instances will launch in the US East (N. Virginia) region'.

The right sidebar contains 'Account Attributes' (Supported Platforms, VPC, Default VPC, Resource ID length management, Console experiments, Settings), 'Additional Information' (Getting Started Guide, Documentation, All EC2 Resources, Forums, Pricing, Contact Us), and 'AWS Marketplace' (Find free software trial products in the AWS Marketplace from the EC2 Launch Wizard. Or try these popular AMIs. Barracuda CloudGen Firewall for AWS - PAYG. By Barracuda Networks, Inc. Rating: 5 stars. Starting from \$0.60/hr or from \$4,599/yr (12% savings) for software + AWS usage fees).

Seleccionaremos el sistema Ubuntu Server 16.04 y lanzaremos la instancia.



Seleccionamos el servidor que se nos ofrece por defecto y continuamos hasta el paso 7, donde al hacer clic en Launch se nos notificará que necesitaremos una clave de acceso para entrar en el servidor.



Daremos nombre a ese archivo y lo descargaremos, acto seguido el servidor comenzará a instalarse y podremos acceder a él con la clave que hemos descargado. De no tener esta clave, no podremos acceder al servidor.

Para este proyecto, también será importante definir grupos de seguridad, o dicho de otra forma, puertos habilitados para el acceso remoto. Para ello, accederemos a la pestaña "Security Groups":

**EC2 Dashboard**

- Events
- Tags
- Reports
- Limits
- INSTANCES
  - Instances
  - Launch Templates
  - Spot Requests
  - Reserved Instances
  - Dedicated Hosts
  - Scheduled Instances
  - Capacity Reservations
- IMAGES
  - AMIs
  - Bundle Tasks
- ELASTIC BLOCK STORE
  - Volumes
  - Snapshots
  - Lifecycle Manager
- NETWORK & SECURITY
  - Security Groups**
  - Elastic IPs
  - Placement Groups
  - Key Pairs
  - Network Interfaces

**Resources**

You are using the following Amazon EC2 resources in the US East (N. Virginia) region:

- 1 Running Instances
- 0 Elastic IPs
- 0 Dedicated Hosts
- 0 Snapshots
- 1 Volumes
- 0 Load Balancers
- 3 Key Pairs
- 4 Security Groups
- 0 Placement Groups

Learn more about the latest in AWS Compute from AWS re:invent by viewing the EC2 Videos.

**Create Instance**

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

[Launch Instance](#)

Note: Your instances will launch in the US East (N. Virginia) region

**Service Health**

**Service Status:**

US East (N. Virginia):

Availability Zone Status:

- us-east-1a: Availability zone is operating normally
- us-east-1b: Availability zone is operating normally
- us-east-1c: Availability zone is operating normally

**Scheduled Events**

US East (N. Virginia):

No events

**Account Attributes**

Supported Platforms

VPC

Default VPC

vpc-f553338f

Resource ID length management

Console experiments

Settings

**Additional Information**

Getting Started Guide

Documentation

All EC2 Resources

Forums

Pricing

Contact Us

**AWS Marketplace**

Find free software trial products or EC2 Launch Wizard. Or try these

Barracuda CloudGen Firewall for

By Barracuda Networks, Inc.

Rating ★★★★★

Starting from \$0.60/hr or from \$4, + AWS usage fees

View all Infrastructure Software

Matillion ETL for Amazon Redshift

Una vez dentro, hacemos clic en Actions -> Edit Inbound Rules, y las configuramos de la siguiente forma:

**Edit inbound rules**

Type	Protocol	Port Range	Source	Description
HTTP	TCP	80	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
HTTP	TCP	80	Custom ::/0	e.g. SSH for Admin Desktop
Custom TCP	TCP	8080	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
Custom TCP	TCP	8080	Custom ::/0	e.g. SSH for Admin Desktop
Custom TCP	TCP	8090	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
MySQL/Aurora	TCP	3306	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
MySQL/Aurora	TCP	3306	Custom ::/0	e.g. SSH for Admin Desktop

[Add Rule](#)

NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This will cause traffic that depends on that rule to be dropped for a very brief period of time until the new rule can be created.

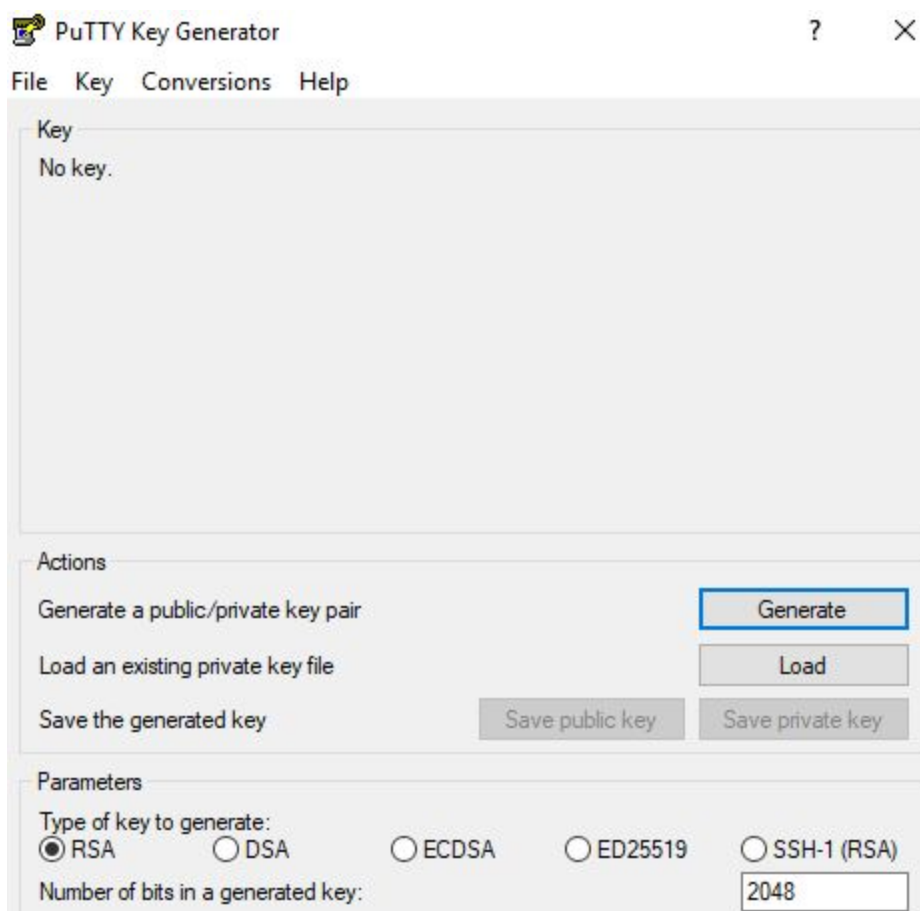
[Cancel](#) [Save](#)

**Importante:** aunque en esta imagen se puede observar el puerto 8080, no se requiere el acceso a este puerto, por lo que este acceso no es necesario configurarlo.

## Conectar con el servidor - PUTTY, Filezilla y SQL Workbench

Dado que el SO en el que estamos trabajando es un Windows 10, para poder acceder al servidor haremos uso de la herramienta PUTTY. Además, la clave de acceso al servidor es un archivo .pem, por lo que para realizar la conexión necesitaremos que el archivo tenga la extensión .ppk .

Por defecto, PUTTY trae una herramienta para poder realizar la conversión del archivo descargado a la extensión necesaria. Para ello, abriremos el ejecutable PUTTYgen.



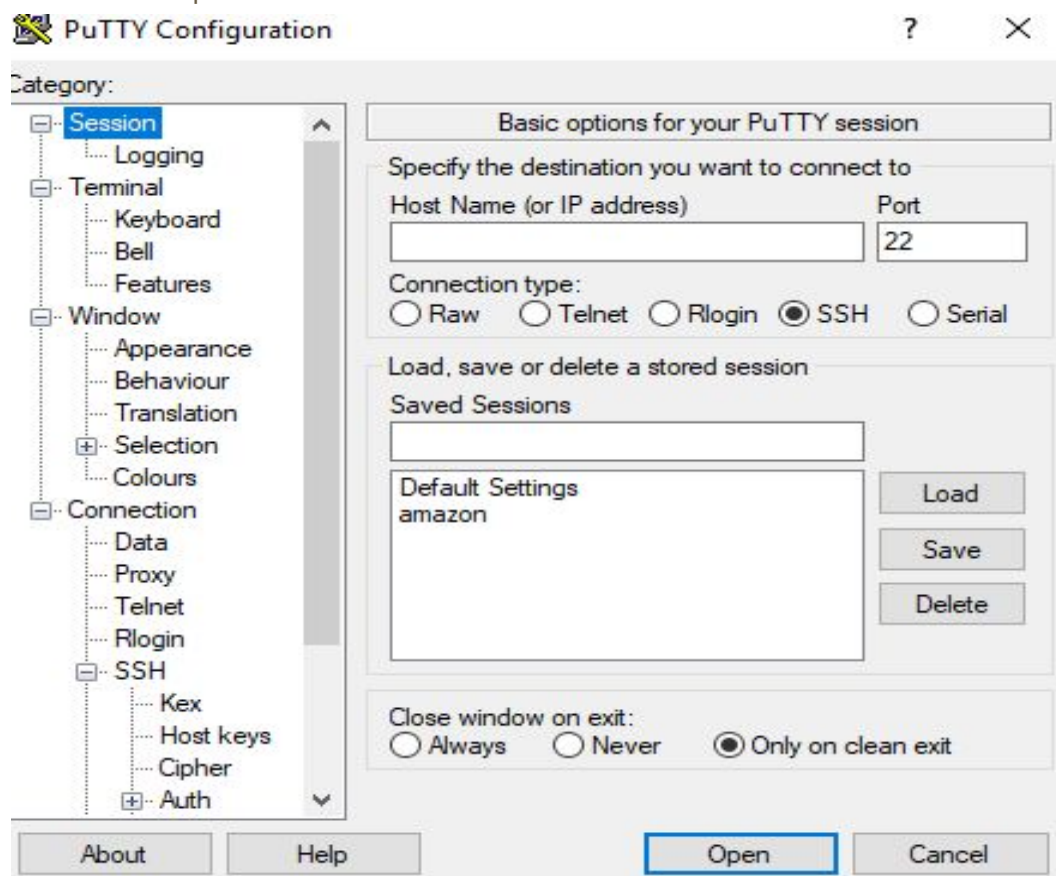
Una vez dentro, en el apartado Conversions -> Import Key importaremos el archivo .pem que hemos descargado. Una vez realizado, únicamente tendremos que hacer clic en "Save private key", y tendremos nuestro archivo .ppk con el que poder acceder al servidor. No obstante, seguiremos necesitando el archivo .pem .

Para acceder al servidor, abriremos PUTTY y realizaremos las siguientes configuraciones:

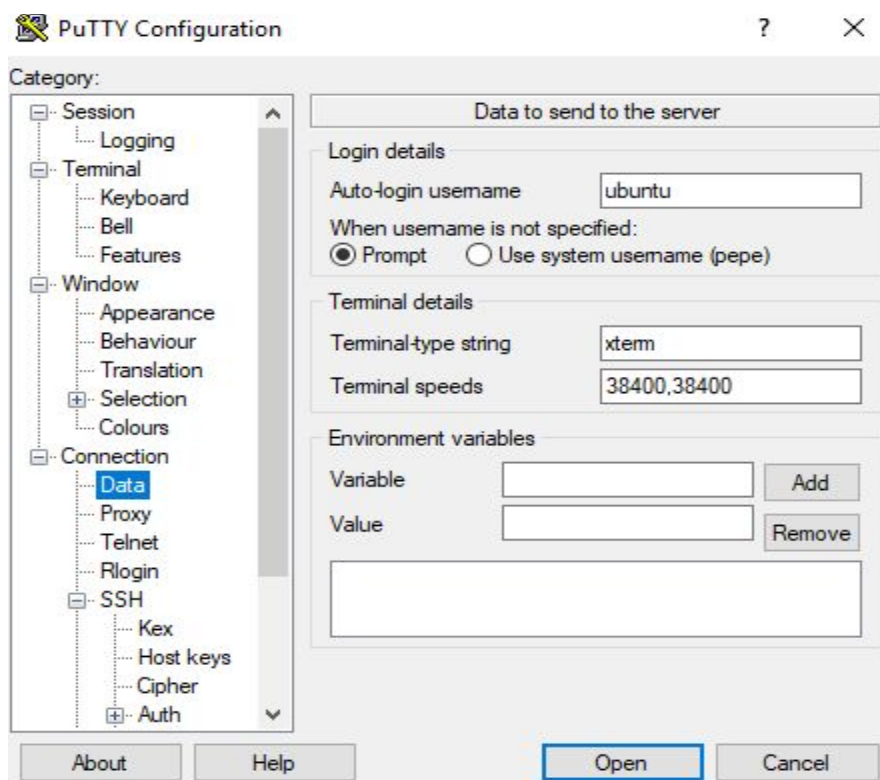
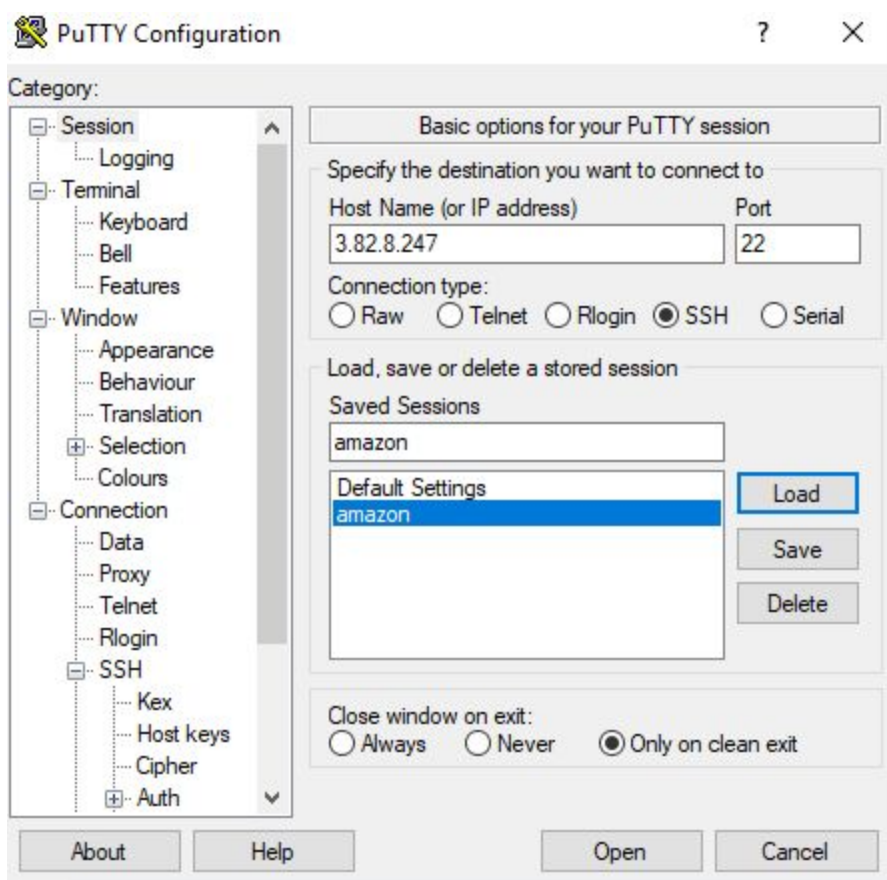
- En primer lugar, guardaremos una sesión y haremos clic en save.

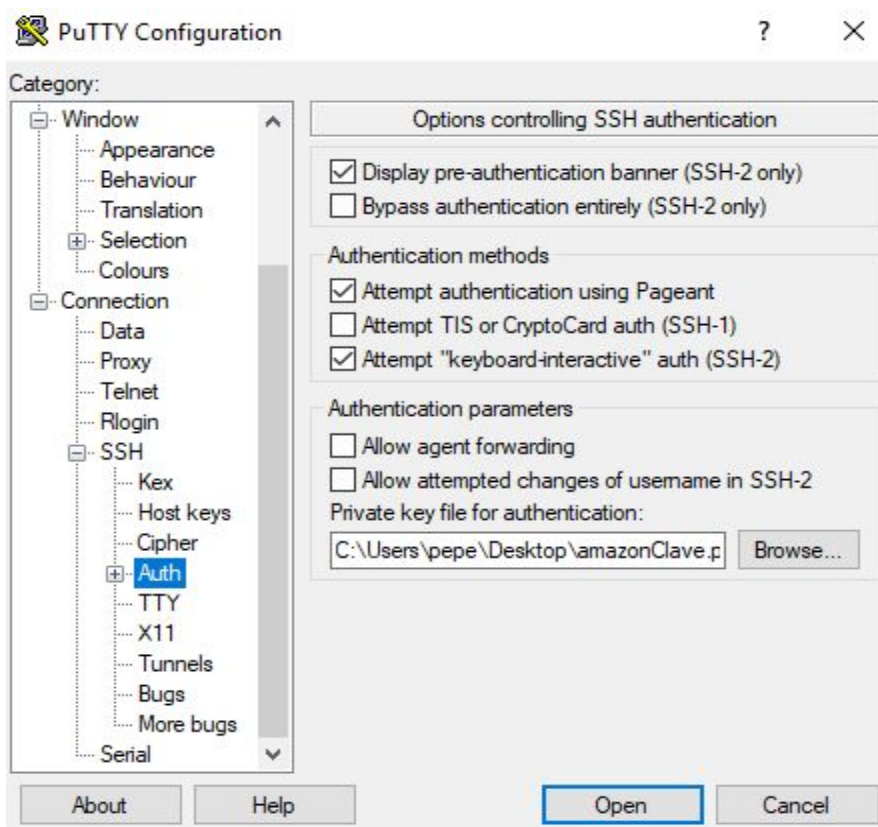


- Acto seguido, en la misma pestaña inicial, introduciremos el nombre del servidor o su IP pública.
- En la pestaña Data, escribiremos dentro del campo Auto-login "ubuntu". Si lo dejamos vacío no es perjudicial, pero de esta forma no preguntará constantemente el usuario con el que deseamos acceder.
- En la pestaña SSH, vamos al apartado Auth, donde introduciremos nuestro archivo .ppk .
- Volvemos al punto inicial, Session, y seleccionamos la sesión creada. Hacemos clic en save y tendremos toda la configuración guardada. Nos será útil para poder acceder múltiples veces.









Si se ha realizado correctamente el resultado es el siguiente:

```

ubuntu@ip-172-31-41-37: ~
Using username "ubuntu".
Authenticating with public key "imported-openssh-key"
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-1083-aws x86_64)

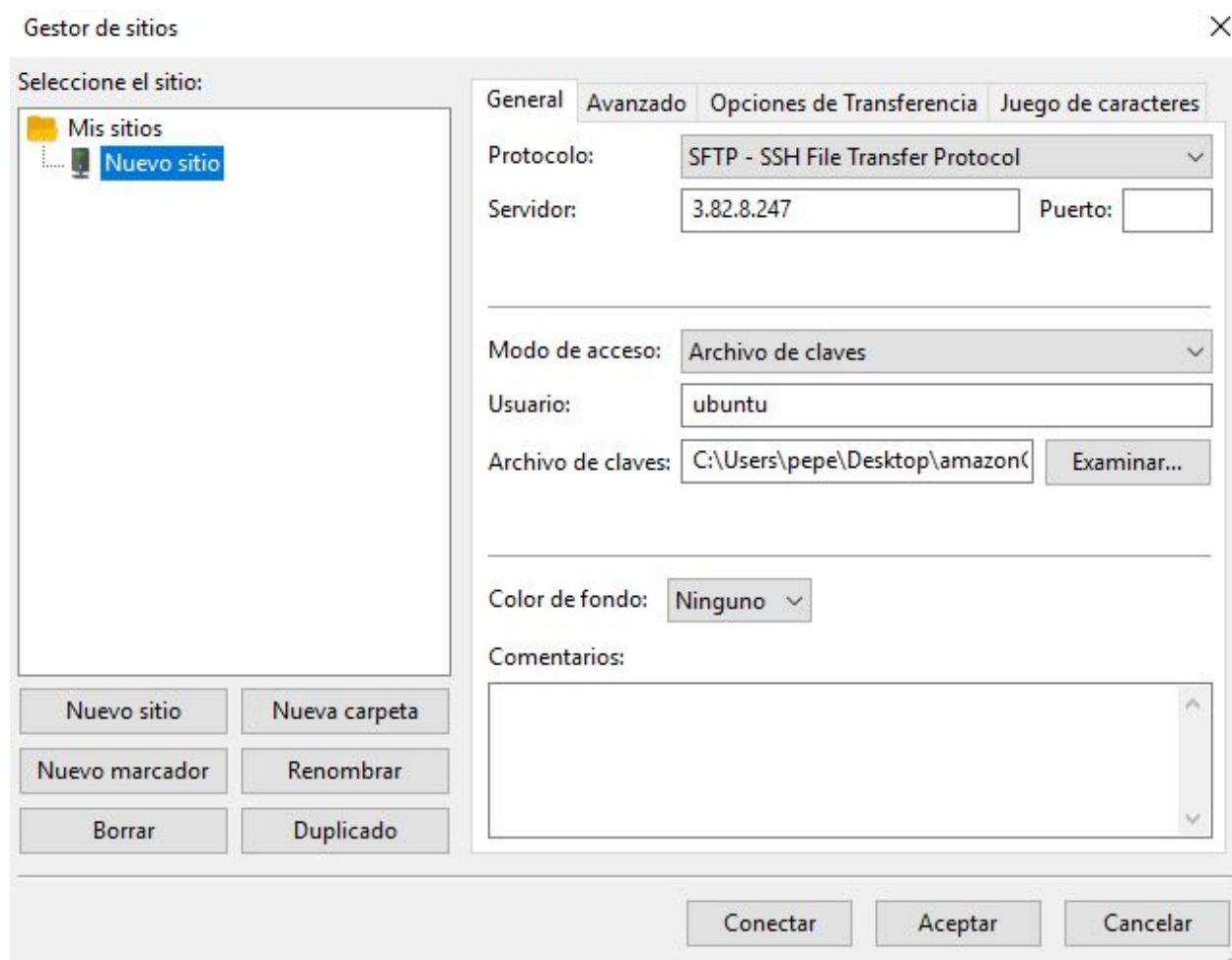
 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

23 packages can be updated.
0 updates are security updates.

*** System restart required ***
Last login: Sun Jun 23 14:47:15 2019 from 94.73.60.199
ubuntu@ip-172-31-41-37:~$

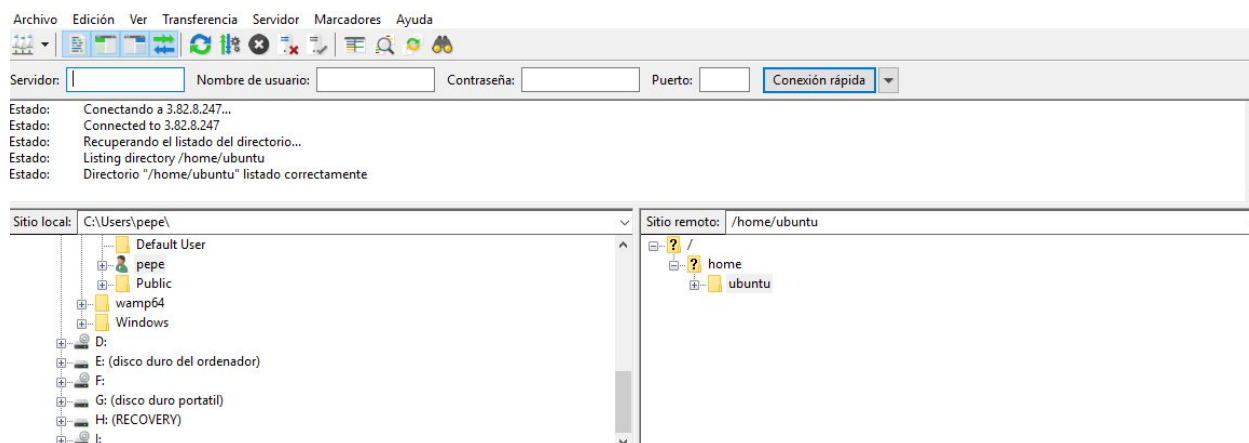
```

Para la transferencia de datos, descargaremos un cliente Filezilla, y realizaremos la configuración de acceso de la siguiente forma dentro de Archivo -> Gestor de Sitios:

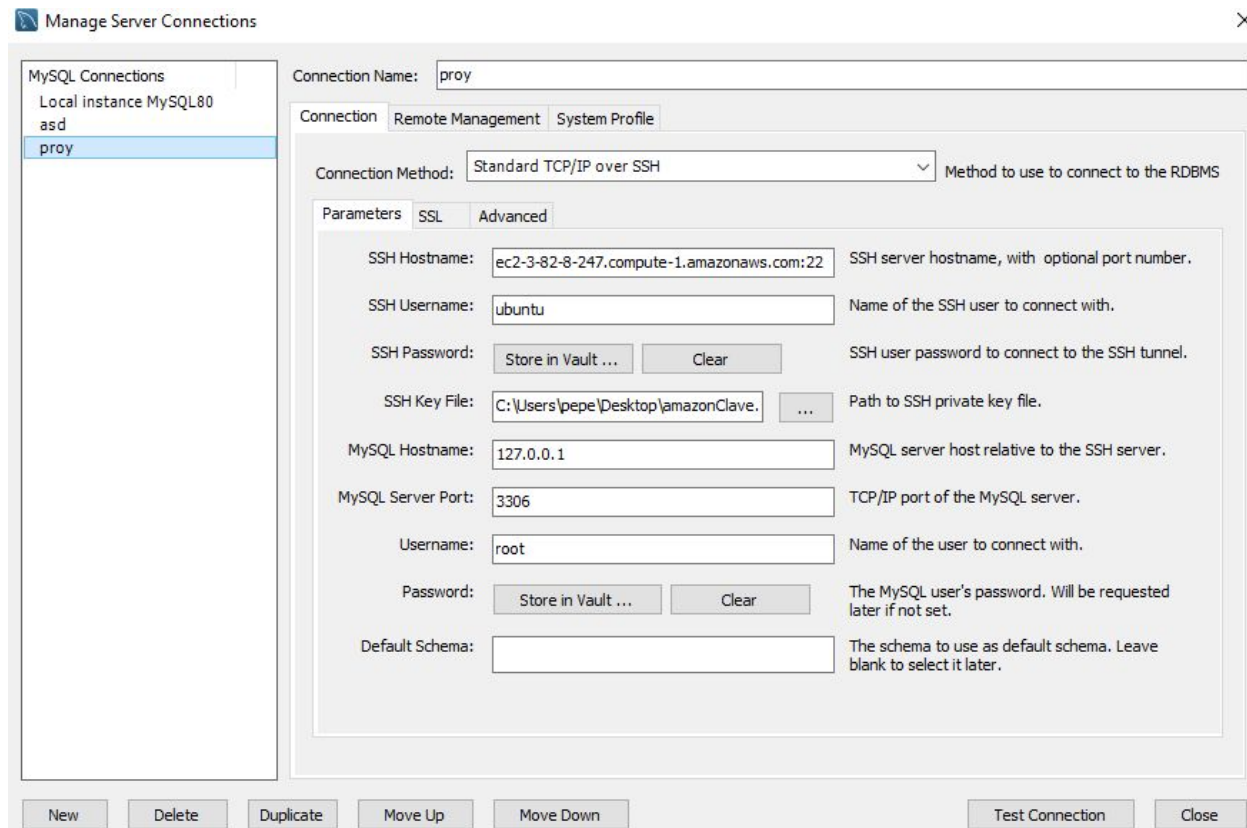


El archivo de claves a añadir en este caso es el mismo archivo de extensión .ppk que utilizamos con PUTTY.

Si se ha realizado correctamente la configuración, la salida debe ser la siguiente:



En el caso de SQL Workbench, tendremos que crear una nueva conexión, donde seleccionaremos “Standard TCP/IP over SSH”, y rellenaremos los datos de la siguiente forma:



The screenshot shows the 'Manage Server Connections' window in SQL Workbench. On the left, a list of MySQL connections includes 'Local instance MySQL80' and 'proy'. The 'proy' connection is selected. The main area displays the configuration for this connection. The 'Connection Name' is 'proy'. The 'Connection Method' is set to 'Standard TCP/IP over SSH'. The 'Parameters' tab is active, showing fields for SSH Hostname, SSH Username, SSH Password, SSH Key File, MySQL Hostname, MySQL Server Port, Username, Password, and Default Schema. The 'Test Connection' button is visible at the bottom right.

Field	Value	Description
Connection Name	proy	
Connection Method	Standard TCP/IP over SSH	Method to use to connect to the RDBMS
SSH Hostname	ec2-3-82-8-247.compute-1.amazonaws.com:22	SSH server hostname, with optional port number.
SSH Username	ubuntu	Name of the SSH user to connect with.
SSH Password	Store in Vault ... Clear	SSH user password to connect to the SSH tunnel.
SSH Key File	C:\Users\pepe\Desktop\amazonClave. ...	Path to SSH private key file.
MySQL Hostname	127.0.0.1	MySQL server host relative to the SSH server.
MySQL Server Port	3306	TCP/IP port of the MySQL server.
Username	root	Name of the user to connect with.
Password	Store in Vault ... Clear	The MySQL user's password. Will be requested later if not set.
Default Schema		The schema to use as default schema. Leave blank to select it later.

## Servidor - Instalaciones y despliegue

Una vez tenemos acceso al servidor mediante Putty, necesitaremos realizar varias descargas para poder poner en marcha nuestro proyecto. Dado que nuestro servidor no tiene contraseña para usuario root, podremos hacer **sudo su** sin introducir contraseña.

Los comandos a realizar para las descargas como usuario root son:

```
apt update
```

```
apt-get install maven
```

```
apt-get install install openjdk-8-jre
```

```
apt-get install mysql-server (en nuestro caso usuario y contraseña root)
```

```
apt-get install apache2
```

Para poder trasladar tanto nuestro frontal como nuestro Backend, haremos uso de Filezilla.

En el caso del frontal, copiaremos todo en la carpeta `/var/www/html`, y automáticamente podremos utilizar nuestro aplicativo en el puerto 80 (por defecto de Apache).

Para la BBDD, podemos hacer una exportación de la BBDD local e importarla en el servidor remoto gracias a la conexión creada.

No obstante, para nuestro Backend, lo recomendable sería tener un descriptor de despliegue que nos permita desplegar el proyecto. Aunque en un principio se pensó en Jenkins como descriptor de despliegue, se descartó esa idea. Al ser un proyecto Spring Boot lleva embebido un servidor Tomcat, por lo que no nos es necesario un descriptor de despliegue.

La única problemática que esto conlleva es el hecho de que el despliegue del backend solo estaría disponible mientras la conexión SSH esté activa. Para ello, hemos creado un script que incluye la directiva maven para la ejecución del proyecto y se ha hecho que se mantenga activo incluso con el cierre de terminal.

Para ello nos situamos dentro del proyecto backend y creamos nuestro script usando nano.

```
nano service.sh
```

Dentro de `service.sh` escribimos la siguiente directiva:



```
Mvn clean install spring-boot:run -Dserver.port=8090
```

Guardamos y ejecutamos el siguiente comando:

```
setsid service.sh
```

De esta forma podremos mantener la ejecución del Backend sin necesidad de usar ningún tipo de descriptor de despliegues.

## Bibliografía

Plantilla utilizada

<https://colorlib.com/wp/template/royal/>

Referencias para la realización del Backend y resolución de dudas

<https://stackoverflow.com>

<https://www.baeldung.com>

<https://docs.spring.io/spring-data>

<https://docs.spring.io/spring-boot/>

Referencias para el Frontend

<https://developer.mozilla.org/>

<https://api.jquery.com/>

<https://getbootstrap.com/docs/4.3/getting-started/introduction/>

Referencias para el servidor

<https://docs.aws.amazon.com/>

<https://help.ubuntu.com/16.04/serverguide/index.html>

<https://www.linkedin.com/pulse/using-mysql-workbench-connect-your-server-aws-ec2-ubuntu-william-ku>

<https://askubuntu.com/questions/88091/how-to-run-a-shell-script-in-background>

Compra de dominio y configuración a servidor remoto

<https://my.freenom.com>

Nombre de dominio comprado

[www.descansoroyal.tk](http://www.descansoroyal.tk)