A dark blue vertical bar runs down the left side of the page. A blue arrow points to the right from this bar, containing the date.

4-5-2020

PRÁCTICA 2

Integración continua, Heroku y Vaadin
(Desarrollo e Integración de Software)

Several thin, curved lines in dark blue and light grey originate from the bottom left and curve upwards and to the right.

Daniel Álvarez, Miguel Ángel Fernández,
Juan Manuel Ortega y Rocío Sosa
UFV

ÍNDICE

| | |
|-----------------------------------|----|
| INTRODUCCIÓN..... | 2 |
| DESARROLLO DE LA APLICACIÓN | 3 |
| Github..... | 3 |
| Maven..... | 4 |
| Vaadin..... | 5 |
| TDD | 9 |
| DESPLIEGUE DE LA APLICACIÓN | 10 |
| Heroku | 10 |
| Docker y Jenkins..... | 11 |
| CONCLUSIONES | 15 |
| BIBLIOGRAFÍA..... | 16 |

INTRODUCCIÓN

El objetivo de la presente práctica es demostrar el conocimiento adquirido a lo largo de todo el cuatrimestre en cuanto a la asignatura de Desarrollo e Integración de Software.

Todos los temas que han sido expuestos en esta asignatura serán necesarios para la práctica: control de versiones con la herramienta Github, gestión de dependencias con maven, Test Driven Development (TDD) con JUnit, pruebas unitarias que irán unidas a la metodología TDD, Docker, PaaS, CI /CD, Jenkins...

La práctica está dividida en dos partes:

- Desarrollo de la aplicación Java.
- Configuración de un proceso de CI / CD y despliegue de la aplicación.

Además, todo este proceso estará alojado en un repositorio privado en Github para un mejor desarrollo en grupo y un eficiente control de versiones y cambios.

La práctica consiste en desarrollar una correcta gestión de una agenda de contactos con diferentes funciones y posteriormente desplegarla en una plataforma (Heroku) que esté unida a un proceso de integración continua (Jenkins).

DESARROLLO DE LA APLICACIÓN

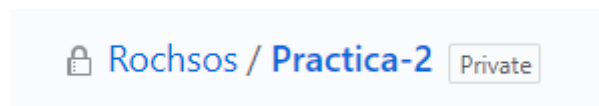
El desarrollo de esta aplicación se ha basado en poder desplegar correctamente una aplicación web basada en Java que permita gestionar una agenda de contactos con la utilización del framework Vaadin 8.6.2 gestionando todas las dependencias por Maven y el control de versiones con GitHub.

Github

Git es el sistema de control de versiones distribuidos que permite almacenar los cambios realizados en diferentes ficheros, además del desarrollo colaborativo.

GitHub es el proveedor de alojamientos de repositorios Git más conocido hoy en día.

Para el buen desarrollo en grupo de esta práctica usaremos este sistema Git alojando nuestro proyecto en un repositorio privado de la herramienta GitHub.

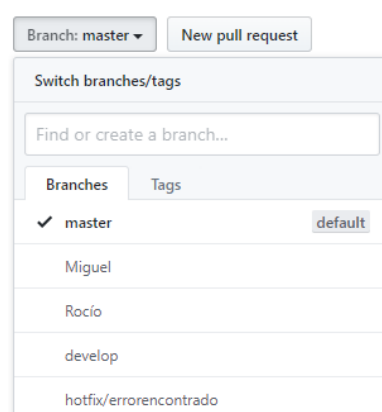


<https://github.com/Rochsos/Practica-2>

Al ser un repositorio privado se ha tenido que conceder el acceso a los diferentes desarrolladores de la práctica, y además a los usuarios: @nachoserranoUFV @pidal para la corrección de esta.

Se ha usado la metodología GitFlow para trabajar con Git de forma ordenada y escalable para la facilitación de trabajar en paralelo con el equipo de desarrollo. Esta metodología incluye las siguientes ramas:

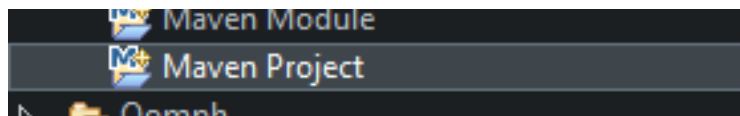
- Master: donde se han incluido las diferentes versiones del proyecto.
- Develop: donde se han subido las pequeñas versiones de funcionalidades del código, y que unía a las ramas Feature donde trabajaba cada desarrollador (se incorpora a master).
- Ramas Feature: se han creado dos diferentes ramas donde ha trabajado cada desarrollador una pequeña funcionalidad del código (se incorporan a develop).
- Hotfix: se encontró un error en la interfaz que se comentará posteriormente y se creó esta rama para corregirlo (se incorpora a master).



Maven

Apache Maven es una herramienta de software para la gestión y compresión de proyectos Java, que está basado en el concepto de un modelo de objeto de proyecto (POM) que es un fichero XML que se usa para describir el proyecto de software a construir y que contiene toda la información necesaria para compilar y ejecutar un proyecto de software con Maven (dependencias, plugins...).

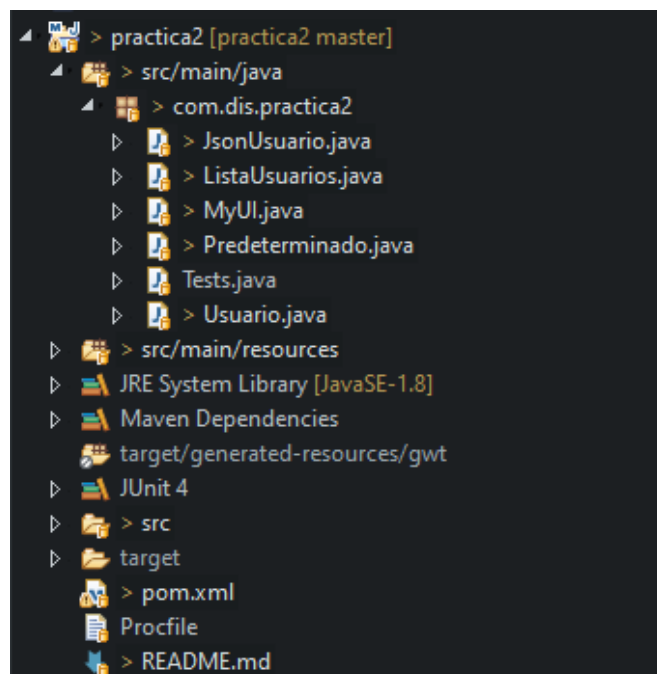
Para crear el trabajo se ha creado un proyecto de Maven Project:



Y se ha dado uso del archetype de vaadin para empezar la aplicación:

| Group Id | Artifact Id | Version | |
|------------|------------------------------|---------|--|
| com.vaadin | vaadin-archetype-application | 8.6.2 | |

Esta es la estructura final de archivos Maven de nuestro proyecto que se ha generado:



Vaadin

El framework Vaadin es una plataforma de código abierto para el desarrollo de aplicaciones web que incluye un conjunto de componentes web con el que se puede construir software.

El diseño de la web usando el framework Vaadin de nuestro proyecto es el siguiente:

AGENDA DE CONTACTOS PERSONAL

DATOS DE LA AGENDA:

Nombre: *
Ej: Luis

Apellidos:
Ej: Pérez

Empresa:
Ej: BABEL

Teléfono: *
Ej: +34 685928591

Email:
Ej: luisperez@gmail.co

Dirección:
Ej: Calle Gran vía, 13

OPCIONES:

Añadir contacto

Eliminar contacto

Modificar contacto

Detalle del contacto

CONTACTOS DE TU AGENDA:

| Nombre | Apellidos | Empresa | Teléfono | Email | Dirección |
|---------|-----------|-----------|-----------|-----------------------|--------------------|
| Luis | Perez | BABEL | 681629037 | luisperez@gmail.com | Calle Gran Vía, 13 |
| Antonio | Rios | Santander | 601883672 | antoniorios@gmail.com | Calle Goya, 34 |

* Si quieres eliminar un contacto, tendrás que introducir el número de teléfono del contacto que quieres eliminar.

* Si quieres modificar un contacto, tendrás que introducir el número de teléfono del contacto que quieres modificar, además de los nuevos datos del contacto a modificar.

Al ser una interfaz de gran tamaño se irán explicando las siguientes funcionalidades.

Datos de la agenda:

La aplicación consta de un formulario donde se deben rellenar unos datos cada vez que se quiera guardar un nuevo contacto en la agenda. Se indican unos datos de ejemplo en el campo de respuesta para guiar al usuario

- Nombre: es un campo obligatorio como indica el asterisco.
- Apellidos: es opcional.
- Empresa: es opcional.
- Teléfono: es un campo obligatorio, además de importante ya que se utilizará como ID único cuando se quiera eliminar un contacto o modificar.
- Email: es opcional.
- Dirección: es opcional.

DATOS DE LA AGENDA:

Nombre: *
Ej: Luis

Apellidos:
Ej: Pérez

Empresa:
Ej: BABEL

Teléfono: *
Ej: +34 685928591

Email:
Ej: luisperez@gmail.co

Dirección:
Ej: Calle Gran vía, 13

Opciones:

Las diferentes opciones que permite la agenda son las CRUD indicadas en el enunciado:

- Create: referida al botón de “añadir contacto” que te permitirá poder crear un nuevo contacto en tu agenda con los datos indicados en el anterior apartado.
- Read: se mostrarán los datos actualizados en la tabla del siguiente apartado.
- Update: referida al botón de “modificar contacto” que te permitirá con el mismo teléfono móvil, modificar los datos del contacto del apartado anterior. Si se quiere modificar el teléfono del contacto, elimínalo y vuélvalo a crear.
- Delete: referida al botón de “eliminar contacto” que te permite eliminar un contacto de tu agenda con el ID del teléfono móvil que escribas en el formulario.

OPCIONES:

Añadir contacto

Eliminar contacto

Modificar contacto

Detalle del contacto

Además, se ha generado un botón de “detalle del contacto” para poder visualizar los datos de un contacto en una página modal creada. Para poder ver los datos del contacto, tendrás que escribir el teléfono del contacto a visualizar y pulsar en el botón indicado.

Detalle del contacto: + x

Nombre: Antonio
Apellidos: Rios
Empresa: Santander
Telefono: 601883672
Email: antoniorios@gmail.com
Direccion: Calle Goya, 34

Contactos de tu agenda (tabla):

Se ha creado una tabla o grid para la visualización de los contactos de la agenda en cada momento.

La tabla contiene los diferentes campos descritos en el apartado del formulario. Además, se han creado unos usuarios predeterminados como ejemplo que siempre aparecerán al iniciar la aplicación.

CONTACTOS DE TU AGENDA:

| Nombre | Apellidos | Empresa | Teléfono | Email | Direccion |
|---------|-----------|-----------|-----------|-----------------------|--------------------|
| Luis | Perez | BABEL | 681629037 | luisperez@gmail.com | Calle Gran Vía, 13 |
| Antonio | Rios | Santander | 601883672 | antoniorios@gmail.com | Calle Goya, 34 |
| | | | | | |

Guía para el usuario:

Para evitar confusiones y que el usuario se sienta cómodo y confiado en la siguiente aplicación, se han aclarado algunos temas para la funcionalidad de la misma.

Para eliminar un contacto, se tendrá que introducir el teléfono móvil de dicho contacto en el formulario (sin necesidad de introducir otros datos) y posteriormente darle al botón de eliminar contacto.

Para modificar un contacto, se tendrán que introducir todos los datos de nuevo para guardar la última modificación del contacto.

Para ver el detalle un contacto, se tendrá que introducir el teléfono móvil de dicho contacto en el formulario (sin necesidad de introducir otros datos) y posteriormente darle al botón de detalle del contacto y se mostrarán los datos en una nueva ventana que podrás cerrarla o ampliarla cuando se desee.

* Si quieres eliminar un contacto, tendrás que introducir el número de teléfono del contacto que quieres eliminar.

* Si quieres modificar un contacto, tendrás que introducir el número de teléfono del contacto que quieres modificar, además de los nuevos datos del contacto a modificar.

Ventanas emergentes de errores:

Si el usuario ha cometido un error al introducir los datos del formulario, se le mostrará un mensaje de error indicando que por favor introduzca bien los campos obligatorios para poder realizar la acción indicada.

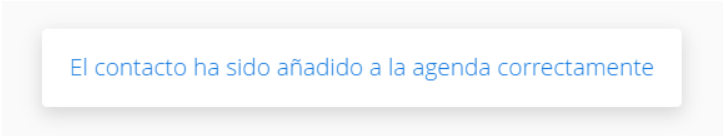
No se puede añadir el contacto, ya que faltan por completar datos obligatorios

Si el usuario no se ha leído correctamente la guía del usuario o ha puesto un teléfono no válido (que no se encuentra en la lista de contactos de tu agenda) se le mostrará un error notificándole.

Introduce un telefono válido.

Ventanas emergentes de avisos:

Cuando el nuevo contacto se añade a la agenda:



El contacto ha sido añadido a la agenda correctamente

Cuando el contacto se modifica en la agenda:



Usuario modificado de la agenda correctamente.

Cuando el contacto se elimina de la agenda.



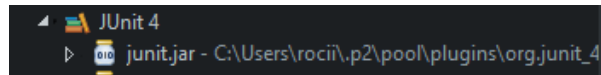
Usuario eliminado de la agenda correctamente.

TDD

Test Driven Development (TDD) es una práctica de ingeniería del software que involucra otras dos prácticas: escribir las pruebas primero y refactorización.

El desarrollo del patrón TDD se evaluará mediante los commits al repositorio de GitHub comprobando si se han enviado primero las pruebas antes de desarrollar la implementación de cada funcionalidad.

Se han desarrollado pruebas unitarias que comprobaban parte de la funcionalidad del proyecto usando la librería JUnit4 importada en las dependencias del POM.xml.



JUnit es un conjunto de bibliotecas que son utilizadas para hacer pruebas unitarias de aplicaciones Java. Se han usado métodos como: assertEquals(), assertTrue()...

Las anotaciones en Java es una forma de añadir metadatos al código que están disponibles cuando se ejecutan los tests. Algunas anotaciones usadas en este proyecto han sido: @Test, @Before...

DESPLIEGUE DE LA APLICACIÓN

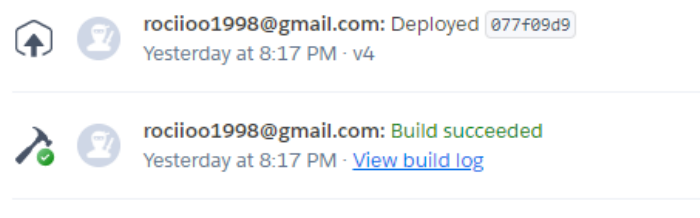
Una vez desarrollada la parte de BackEnd de la aplicación que engloba la lógica de esta, se procede a desplegar la aplicación en heroku para la correcta visualización de la interfaz gráfica del usuario que engloba el FrontEnd.

Heroku

Es una plataforma como servicio de computación en la nube que soporta distintos lenguajes de programación. Es uno de los PaaS (Platform as a Service) más utilizados en la actualidad en entornos empresariales por su fuerte enfoque en resolver el despliegue de una aplicación.

Esta práctica tenía que subirse a una web para que sea accesible a través de internet y para ello se ha hecho uso de la página web Heroku.

Para poder desplegar mi aplicación en Heroku se ha procedido a el login de la cuenta y la creación de una nueva aplicación. Además, para que todo funcionara correctamente se ha tenido que añadir a la estructura de mis ficheros del proyecto, un fichero llamado Procfile, recogido en Moodle, que especifica los comandos que ejecuta la aplicación al inicio y así crear el archivo war del proyecto. También la utilización de un plugin jetty (servidor HTTP) añadido al pom.xml de nuestro proyecto para que Heroku pudiera ejecutar la aplicación.



Una vez hecho esto asociamos el repositorio de la aplicación y ya nos indicaría el enlace donde se encuentra la aplicación.

El enlace de la aplicación desplegada en heroku es:

<https://practica2-dis-2020.herokuapp.com/>

Docker y Jenkins

Docker es una plataforma de software (Container as a Service) que permite crear, probar e implementar aplicaciones rápidamente. Empaqueta software en contenedores donde se incluye todo lo necesario para que el software se ejecute sin ningún problema desde cualquier entorno.

Jenkins es un servidor de automatización de parte del proceso de desarrollo de software mediante integración continua y facilita ciertos aspectos de la entrega continua.

En el enunciado se pedía montar una imagen de Jenkins en Docker con Maven instalado, en un contenedor de Docker para ejecutar las pruebas del proyecto de forma automática. Para ello se ha creado un Dockerfile, que es un archivo de configuración personalizado que define la creación de una imagen.

El Dockerfile creado contiene tres órdenes:

- FROM Jenkins/Jenkins:lts: donde se indica la imagen de la que se parte y se basa mi imagen personalizada, en este caso de la imagen de Jenkins que queremos virtualizar.
- USER root: indicamos el usuario root ya que será un sistema Linux.
- RUN /bin/bash -c ... : donde se indica que ejecute en la línea de comandos la orden de actualizar y instalar Maven para la correcta ejecución del proyecto.

Se visualiza el archivo Dockerfile con la orden cat, y se construye la imagen con el tag: jenkinspractica.

```
C:\Users\roci\OneDrive\Escritorio\jenkinspractica>cat Dockerfile
FROM jenkins/jenkins:lts
USER root
RUN /bin/bash -c "apt-get -y update && apt-get -y install maven"
C:\Users\roci\OneDrive\Escritorio\jenkinspractica>docker build -t jenkinspractica .
Sending build context to Docker daemon  2.048kB
Step 1/3 : FROM jenkins/jenkins:lts
--> 7e250da768ed
Step 2/3 : USER root
--> Using cache
--> dc4bacddab55
Step 3/3 : RUN /bin/bash -c "apt-get -y update && apt-get -y install maven"
--> Using cache
--> 4dc5d1a9e41d
Successfully built 4dc5d1a9e41d
Successfully tagged jenkinspractica:latest
SECURITY WARNING: You are building a Docker image from Windows against a non-Windows Docker host. All files and directories for sensitive files and directories.
```

Como podemos observar la imagen está creada

```
C:\Users\roci\OneDrive\Escritorio\jenkinspractica>docker images
```

| REPOSITORY | TAG | IMAGE ID | CREATED | SIZE |
|-----------------|--------|--------------|-------------|-------|
| jenkinsmaven | latest | 4dc5d1a9e41d | 11 days ago | 808MB |
| jenkinspractica | latest | 4dc5d1a9e41d | 11 days ago | 808MB |
| ubuntu | vim 1 | 236b57621dd9 | 12 days ago | 153MB |

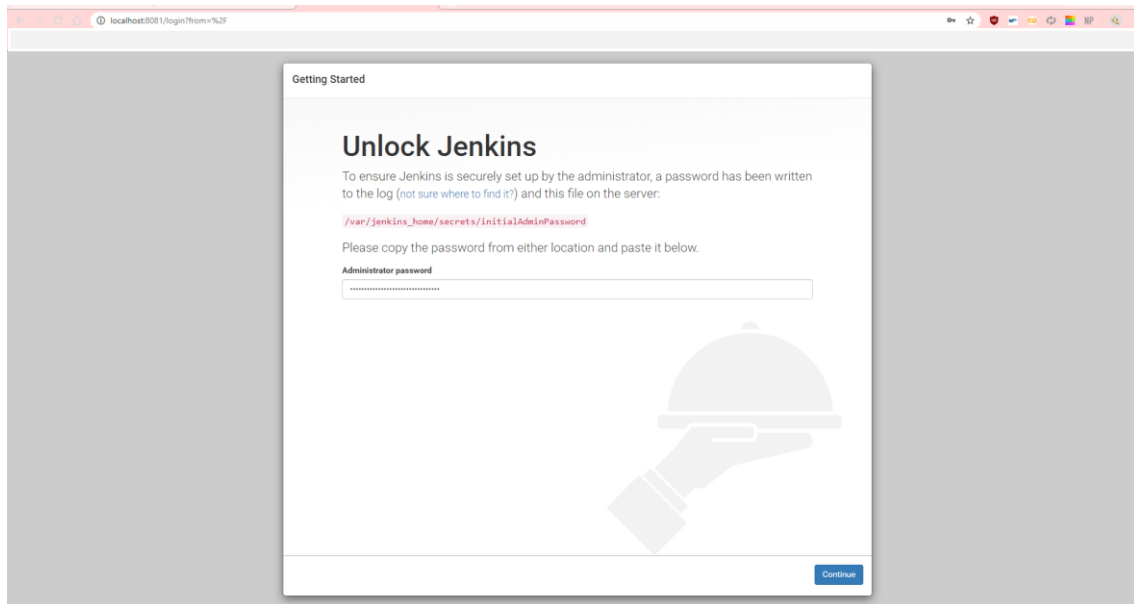
Una vez creada la imagen, se crea un contenedor con Docker run indicando el puerto para poder ejecutar ahí la imagen personalizada.

```
C:\Users\roci\OneDrive\Escritorio\jenkinspractica>docker run -p 8081:8080 -p 50000:50000 jenkinspractica
```

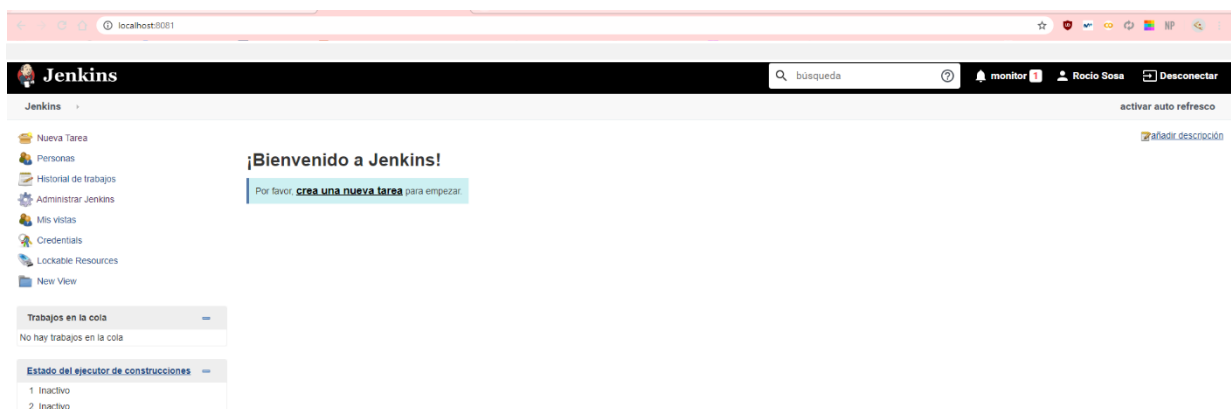
Se puede comprobar con Docker ps que el contenedor está creado y está ejecutándose.

```
C:\Users\roci\OneDrive\Escritorio\PRACTICA2-DIS-KRAKEN\Practica-2\jenkinspractica>docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS                               NAMES
be9b918f7934   jenkinspractica   "/sbin/tini -- /usr/..."   3 hours ago    Up 3 hours    0.0.0.0:50000->50000/tcp, 0.0.0.0:8081->8080/tcp   recursing_booth
```

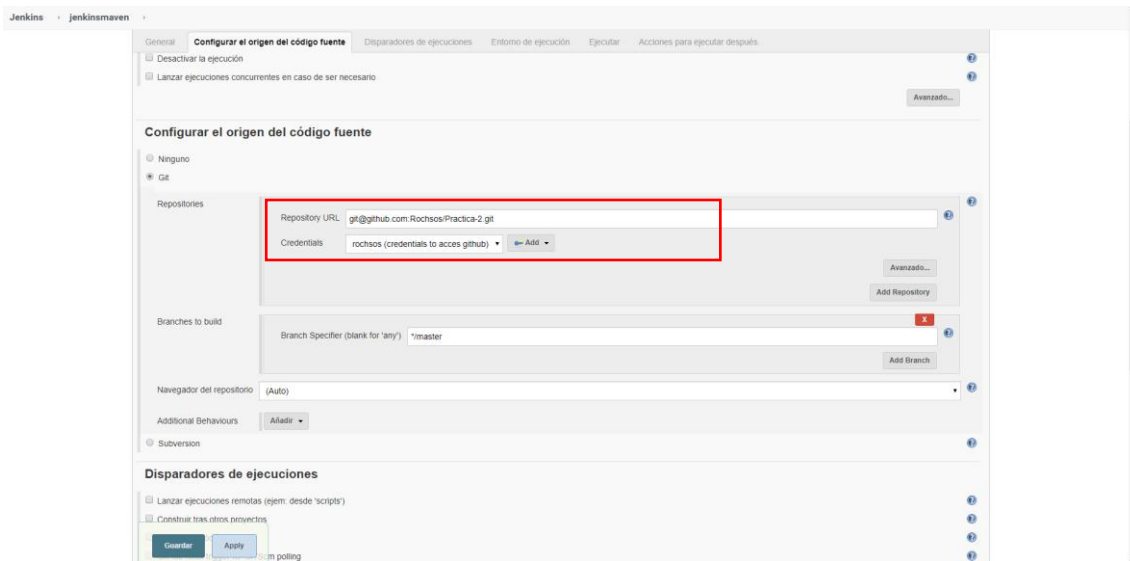
Una vez que el contenedor está corriendo, te proporciona una contraseña que se debe copiar para poder entrar correctamente al servidor de Jenkins. Primero entramos al puerto especificado: localhost:8081 y pegamos la contraseña copiada.



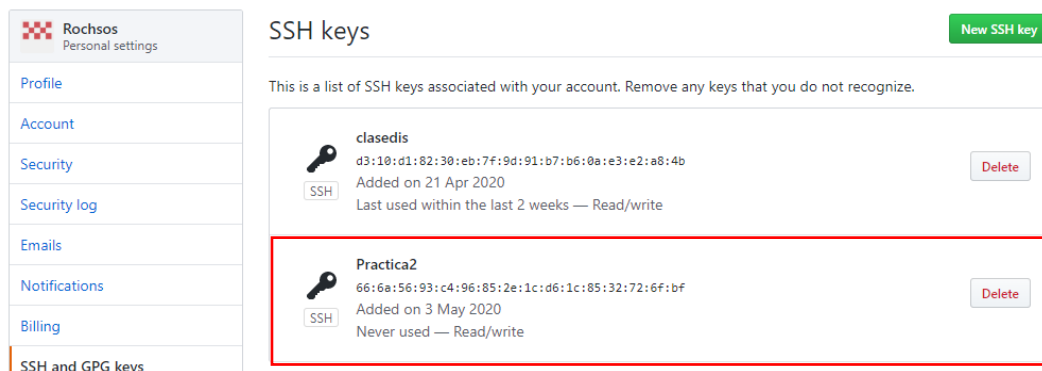
Ahora ya estamos dentro de Jenkins a través de un contenedor virtual:



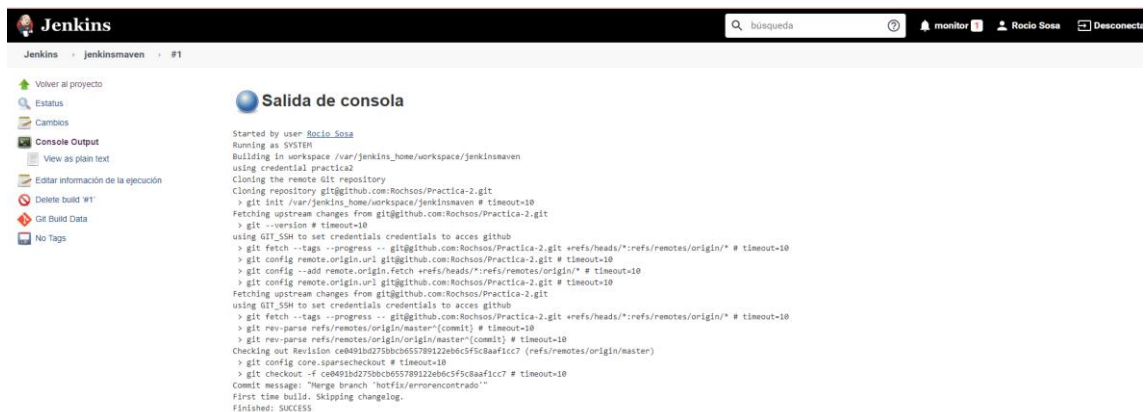
Creamos una nueva tarea llamada jenkinsmaven y lo configuramos para conectarlo con el repositorio de GitHub:



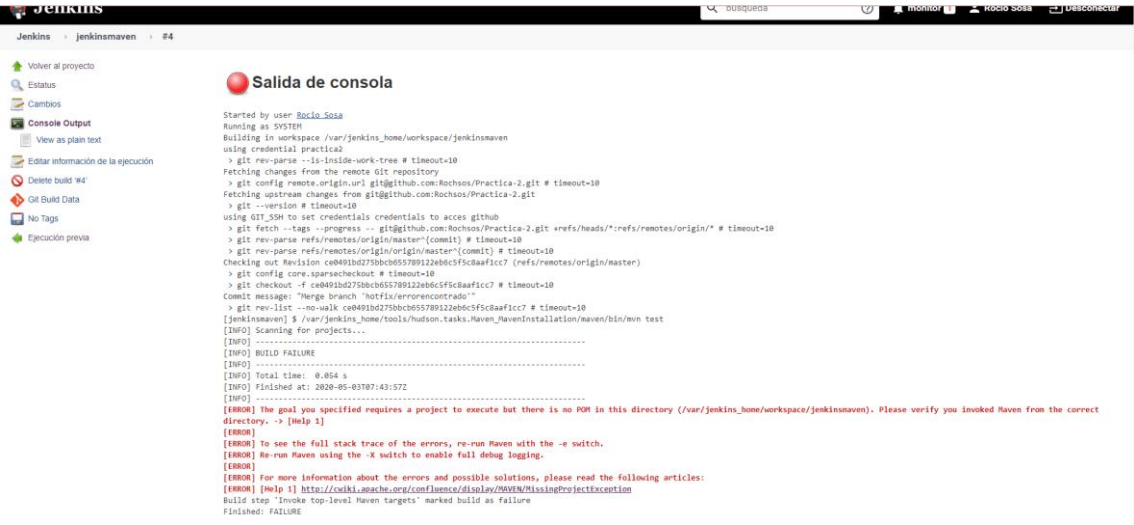
Para eso hemos necesitado crear una nueva clave ssh y añadirla en nuestro github:



Lo compilamos y podemos ver que se ha completado satisfactoriamente.



Después de esto creamos una nueva tarea para ejecutar el proyecto con Maven, pero nos da un error en consola que desconocemos:



The screenshot shows the Jenkins web interface for a build named 'jenkinsmaven' with ID '#4'. The console output is titled 'Salida de consola' and shows the following sequence of events:

- Started by user **Rocio Sosa**.
- Running as **SYSTEM**.
- Building in workspace `/var/jenkins_home/workspace/jenkinsmaven`.
- Using credential `practica2`.
- Git rev-parse --is-inside-work-tree # timeout=10
- Fetching changes from the remote Git repository
- git config remote.origin.url git@github.com:Rochas/Practica-2.git # timeout=10
- Fetching upstream changes from git@github.com:Rochas/Practica-2.git
- git --version # timeout=10
- using GIT_SSH to set credentials credentials to acces github
- git fetch --tags --progress -- git@github.com:Rochas/Practica-2.git +refs/heads/*:refs/remotes/origin/* # timeout=10
- git rev-parse refs/remotes/origin/master^{commit} # timeout=10
- git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
- Checking out Revision `ce0401bd2750bc6655789122eb6c5f5c8aaf1cc7` (refs/remotes/origin/master)
- git config core.sparsecheckout # timeout=10
- git checkout -f `ce0401bd2750bc6655789122eb6c5f5c8aaf1cc7` # timeout=10
- Commit message: "Merge branch 'hotfix/errorrecontrado'"
- git rev-list --no-walk `ce0401bd2750bc6655789122eb6c5f5c8aaf1cc7` # timeout=10
- [jenkinsmaven] \$ `/var/jenkins_home/tools/hudson.tasks.Maven_HavenInstallation/maven/bin/mvn test`
- [INFO] Scanning for projects...
- [INFO] BUILD FAILURE
- [INFO] Total time: 0.854 s
- [INFO] Finished at: 2020-05-03T07:43:57Z
- [ERROR] The goal you specified requires a project to execute but there is no POM in this directory (`/var/jenkins_home/workspace/jenkinsmaven`). Please verify you invoked Maven from the correct directory. -> [help 1]
- [ERROR]
- [ERROR] To see the full stack trace of the errors, re-run Maven with the `-e` switch.
- [ERROR] Re-run Maven using the `-X` switch to enable full debug logging.
- [ERROR]
- [ERROR] For more information about the errors and possible solutions, please read the following articles:
- [ERROR] [help 1] <http://wiki.apache.org/confluence/display/MYVN/MissingProjectException>
- Build step 'Invoke top-level Maven targets' marked build as failure
- Finished: FAILURE

CONCLUSIONES

Tal y como vimos en la anterior práctica, la herramienta GitHub nos ha facilitado mucho a la hora de trabajar en equipo y no tener ningún tipo de problema mientras desarrollábamos nuestro trabajo individualmente para luego implementarlo en una unidad conjunta.

Tras haber realizado esta práctica, podemos afirmar que Vaadin es un framework muy útil ya que nos permite trabajar con una gran variedad de funcionalidades fáciles de utilizar y muy visibles. El hecho de poder crear una interfaz gráfica con además una base del proyecto al inicio hace que tenga una gran capacidad de personalización.

También hemos podido comprobar la importancia de realizar pruebas en un proyecto, debido a que disminuye considerablemente la posibilidad de tener un fallo en el proyecto a futuro. En concreto, Junit nos ha parecido una herramienta muy útil y visual para poder probar estos tests, de manera que te avisaba de los tests que faltaban por implementar.

Hemos aprendido los conceptos de Docker y su gran utilidad en el mercado, para poder virtualizar aplicaciones rápidamente con un mejor rendimiento y una mayor compatibilidad entre diferentes entornos. En nuestro caso se ha virtualizado el servidor Jenkins.

Además, aunque no se ha podido automatizar correctamente las pruebas mediante Jenkins, conocemos realmente los beneficios de poder automatizar todos estos procesos de cara a un proyecto más grande.

Por último, al desplegar nuestra aplicación en Heroku, hemos podido visualizar de una manera más rápida nuestra aplicación, sin tener que ejecutar el proyecto, simplemente con un enlace. Esto nos ha parecido realmente útil y sencillo.

En general, el desarrollo del proyecto se ha podido llevar a cabo correctamente y hemos aprendido muchos conceptos que antes solo los llevábamos a cabo en la teoría. Además, hemos sabido enfrentarnos a diferentes errores que nos iban apareciendo en cada fase del proyecto de manera autónoma.

BIBLIOGRAFÍA

<https://vaadin.com/>

<https://es.wikipedia.org/wiki/Vaadin>

<http://maven.apache.org/>

<https://moodleufv.ufv.es>

<https://www.heroku.com/>

https://hub.docker.com/_/jenkins

<https://github.com>

<https://es.stackoverflow.com/>

<https://mvnrepository.com/>