Documento de gestión de la entrega y el despliegue

decide-part-chiquito

Grupo 2
Curso escolar: 2023/2024
Evolución y gestión de la configuración

Integrantes

Juan Jesús Campos Garrido	decide-part-chiquito-1
Alejandro Campano Galán	decide-part-chiquito-1
Antonio Carretero Díaz	decide-part-chiquito-1
Pablo Mera Gómez	decide-part-chiquito-1
Carlos Baquero Villena	decide-part-chiquito-1
David Cortabarra Romero	decide-part-chiquito-1
Alejandro Pérez Vázquez	decide-part-chiquito-2
Juan Carlos Ramírez López	decide-part-chiquito-2
Carmen Ruiz Porcel	decide-part-chiquito-2
Alejandro Santiago Félix	decide-part-chiquito-2
Sergio Santiago Sánchez	decide-part-chiquito-2
María Vico Martín	decide-part-chiquito-2

El despliegue de nuestra aplicación lo tenemos definido desde tres métodos distintos que son: **despliegue del sistema de forma local, despliegue del sistema mediante contenedores y despliegue del sistema mediante máquinas virtuales.**

DESPLIEGUE DEL SISTEMA DE MANERA LOCAL

Para el despliegue de forma local nos bastará con tener el proyecto clonado, una base de datos postgres y lanzarlo, ejecutando de la siguiente forma dentro de la carpeta del proyecto y dentro del entorno virtual:

- ./decide/manage.py runserver

Para la primera vez que se desea lanzar el proyecto se debe seguir los siguientes pasos:

1. Tener python 3.10,postgres y git instalado.
2. Crear un entorno virtual con python 3.10 y, a partir de ahora, trabajar con el entorno activo.
3. Clonar el proyecto.
4. Realizar los siguientes comandos dentro del proyecto:
    a. *pip install -r requirements.txt*
    b. *sudo su -postgres*
    c. *psql -c "create user decideuser with password 'decidepass123'"*
    d. *psql -c "create database decidedb owner decideuser"*
    e. *./decide/manage.py migrate*
    f. *./decide/manage.py createsuperuser*
5. *Configurar el local_settings.py correctamente con las urls y los datos de la base de dato de postgres*

DESPLIEGUE DEL SISTEMA MEDIANTE CONTENEDORES

Para el despliegue mediante contenedores hemos decidido usar docker y configurar todos sus correspondientes archivos haciendo pequeños cambios como usar una imagen “Slim” que es más pequeña que alpine y distintas modificaciones para los static. Para su despliegue nos situaremos dentro del directorio “docker” y ejecutaremos:

- *docker compose build –no-chache && docker compose up*

Ya podremos acceder a http://10.5.0.1:8000 y encontraremos el proyecto. Para crear un usuario admin (de momento, en una futura actualización se creará automáticamente), lanzar en otra terminal:

- *docker compose run web python manage.py createsuperuser*

Para la primera vez que iniciemos el proyecto tendremos que:

- Instalar docker:
    - sudo apt update
    - sudo apt install apt-transport-https ca-certificates curl software-properties-common
    - curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
    - echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg]

https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
    - sudo apt update
    - sudo apt install docker-ce
    - sudo usermod -aG docker ${USER}
    - su - ${USER}
- Instalar Docker-Compose:
    - mkdir -p ~/.docker/cli-plugins/
    - curl -SL https://github.com/docker/compose/releases/download/v2.3.3/docker-compose-linux-x86_64 -o ~/.docker/cli-plugins/docker-compose
    - chmod +x ~/.docker/cli-plugins/docker-compose

DESPLIEGUE DEL SISTEMA MEDIANTE MÁQUINAS VIRTUALES

Para el despliegue mediante máquinas virtuales hemos decidido usar vagrant y configurar todos sus correspondientes archivos. Para su despliegue nos situaremos dentro del directorio “vagrant” y ejecutaremos:

- *vagrant up*

Ya podremos acceder a http://localhost:8080/ y encontraremos el proyecto. Las credenciales de administrador son: admin admin

Para la primera vez que iniciemos el proyecto tendremos que:

- Instalar vagrant, ansible y virtual box:
    - sudo apt update
    - sudo apt install vagrant ansible virtualbox

DESPLIEGUE DEL SISTEMA MEDIANTE HOSTING

Para el despliegue en remoto mediante un proveedor de Hosting, hemos decidido emplear Render como Paas. Con el fin de conseguir una gestión óptima, hacemos uso de un archivo local_setting.deploy.py con el fin de facilitar el servidor la configuración oportuna. Debido a que nuestra suscripción a Render es gratuita, requerimos de un script que realice todas las acciones previas al lanzamiento. Finalmente el sistema es ejecutado mediante Gunicorn para garantizar el aprovisionamiento de los recursos del sistema. Según nuestra estrategias de despliegue, debemos de volcar el contenido de la rama **develop** en una nueva llamada **release/n** donde “n” hace referencia a la versión lanzada, de esta última en **main**, luego gracias a un workflow, este cambio será detectado por GitHub y se realizará un despliegue automático en Render.

Debemos de tener en cuenta que todas aquellas variables de entorno nuevas deberán de ser añadidas en Render.

Para acceder al sistema solo debemos de navegar hacia la dirección Decide-part-chiquito, haciendo click en el enlace.