

# **MANUAL DE INSTALACIÓN TABLERO**

**“COMPRAR O VENDER UNA VIVIENDA EN COLOMBIA”**

**PROYECTO FINAL DESPLIEGUE DE SOLUCIONES ANALÍTICAS**

**MAESTRÍA EN INTELIGENCIA ANALÍTICA DE DATOS MIAD**

**UNIVERSIDAD DE LOS ANDES.**

**Elaborado por: Daniela Castillo Tellez**

**Andres Santiago Serna Tangarife**

**William Morales**

**Katherine Russi Parra**

## INSTALACIÓN DE TABLERO “COMPRAR O VENDER UNA VIVIENDA EN COLOMBIA”

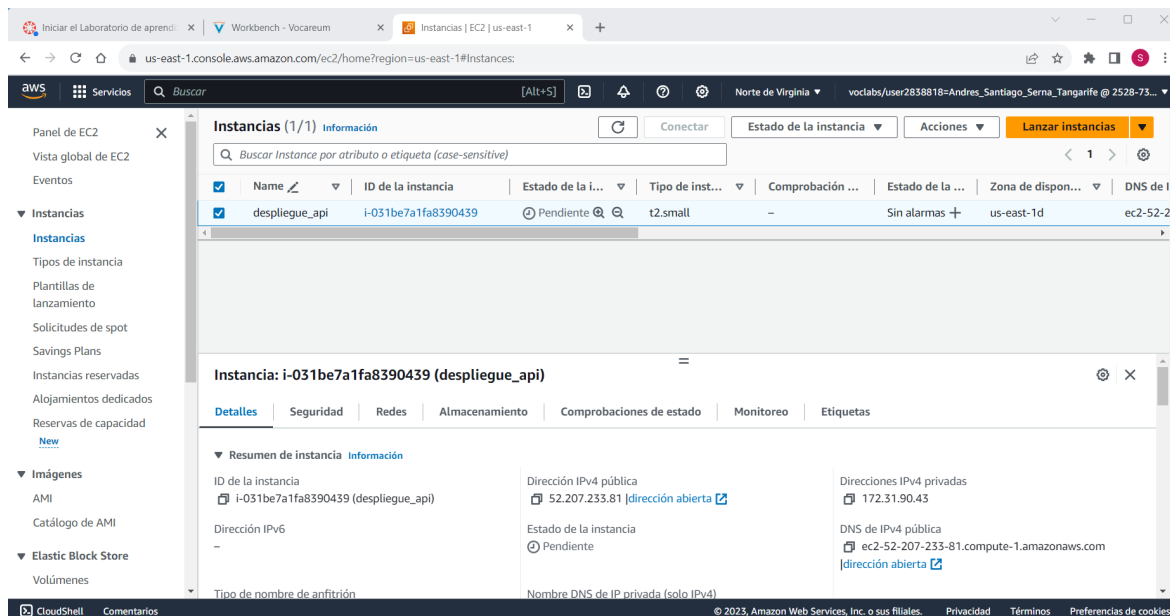
A continuación se definen las siguientes herramientas necesarias para la configuración de despliegue analítico usando contenedores Docker, conectando el tablero “COMPRAR O VENDER UNA VIVIENDA EN COLOMBIA” y la API de predicción.

### 1. Disposición tecnológica de la API

A continuación se definen las siguientes herramientas necesarias para la configuración de despliegue del tablero y la API.

1. Se crea en la consola de EC2 la instancia de t2.small, Ubuntu server con 20 GB de disco. Con el nombre despliegue\_api

Máquina API



2. abrir la instancia en la terminal donde se encuentra la llave.pem del proyecto y la IP de la dirección pública de la máquina virtual

```
ssh -i llave.pem ubuntu@IP_pública
```

Ejemplo

```
ssh -i llave.pem ubuntu@52.207.233.81
```

3. En la máquina virtual, se eliminan las versiones anteriores de Docker (esto genera un error si no hay versiones anteriores, en cuyo caso puede continuar sin problema)

```
sudo apt-get remove docker docker-engine docker.io containerd runc
```

4. Actualice el índice de paquetes

```
sudo apt-get update
```

5. Instale dependencias para verificar certificados (ca-certificates), obtener objetos con su URL (curl) y administrar llaves PGP (gnupg)

```
sudo apt-get install ca-certificates curl gnupg
```

6. Agregue la llave de Docker

```
sudo install -m 0755 -d /etc/apt/keyrings
```

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
```

7. Agregue el repositorio de Docker a su sistema para la instalación

```
echo \
"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \
"$(. /etc/os-release && echo "$VERSION_CODENAME")" stable " | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

8. Actualizar nuevamente el índice de paquetes con este nuevo repositorio incluido

```
sudo apt-get update
```

9. Instalar Docker Engine, containerd, y Docker Compose

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

10. En la terminal de la máquina virtual en EC2, clonar el repositorio de GitHub donde se encuentra en la carpeta del proyecto.

```
git clone https://github.com/asernat/miad_proyecto_despliegue.git
```

11. En la terminal abra el repositorio clonado

```
cd miad_proyecto_despliegue
```

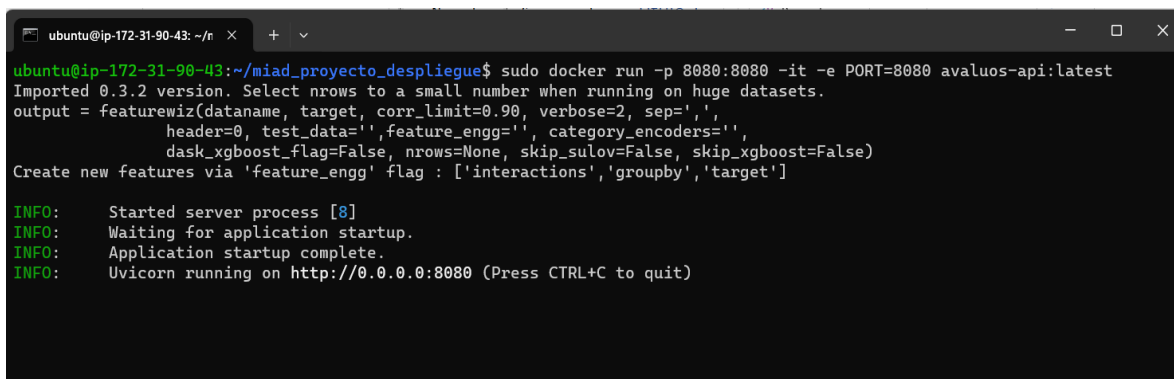
12. A partir del archivo Dockerfile construya la imagen que usará más adelante para lanzar contenedores

```
sudo docker build -f Dockerfile.api -t avaluos-api:latest .
```

13. Ahora ejecute un contenedor usando la imagen creada con el comando

```
sudo docker run -p 8080:8080 -it -e PORT=8080 avaluos-api:latest
```

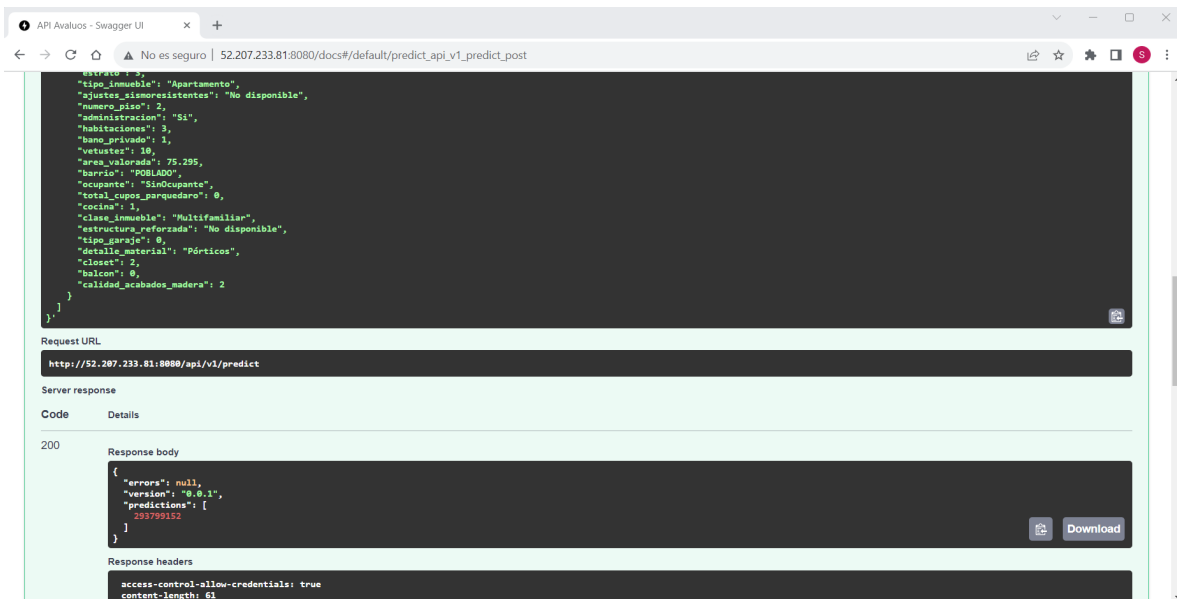
14. Ahora a la consola de EC2, seleccione su máquina virtual ([despliegue\\_api](#)), modificar el grupo de seguridad para permitir tráfico por el puerto 8080. En la pestaña de grupo de seguridad de la máquina y editar las reglas de entrada y agregue una que permita tráfico por el puerto TCP 8080 desde cualquier IP (anywhere IPv4).
15. Ejecución contenedor de api en máquina virtual, como se muestra a continuación



```
ubuntu@ip-172-31-90-43: ~/miad_proyecto_despliegue$ sudo docker run -p 8080:8080 -it -e PORT=8080 avaluos-api:latest
Imported 0.3.2 version. Select nrows to a small number when running on huge datasets.
output = featurewiz(dataname, target, corr_limit=0.90, verbose=2, sep=',',
                    header=0, test_data='', feature_engg='', category_encoders='',
                    dask_xgboost_flag=False, nrows=None, skip_sulov=False, skip_xgboost=False)
Create new features via 'feature_engg' flag : ['interactions', 'groupby', 'target']

INFO:      Started server process [8]
INFO:      Waiting for application startup.
INFO:      Application startup complete.
INFO:      Uvicorn running on http://0.0.0.0:8080 (Press CTRL+C to quit)
```

16. Se Copia la IP pública de su máquina del a API ejemplo [52.207.233.81](#) y en un navegador local navega en la página IP:8080 (ejemplo [52.207.233.81:8080](#)) Allí debe aparecer el tablero predict\_api\_v1-predict\_post en ejecución.

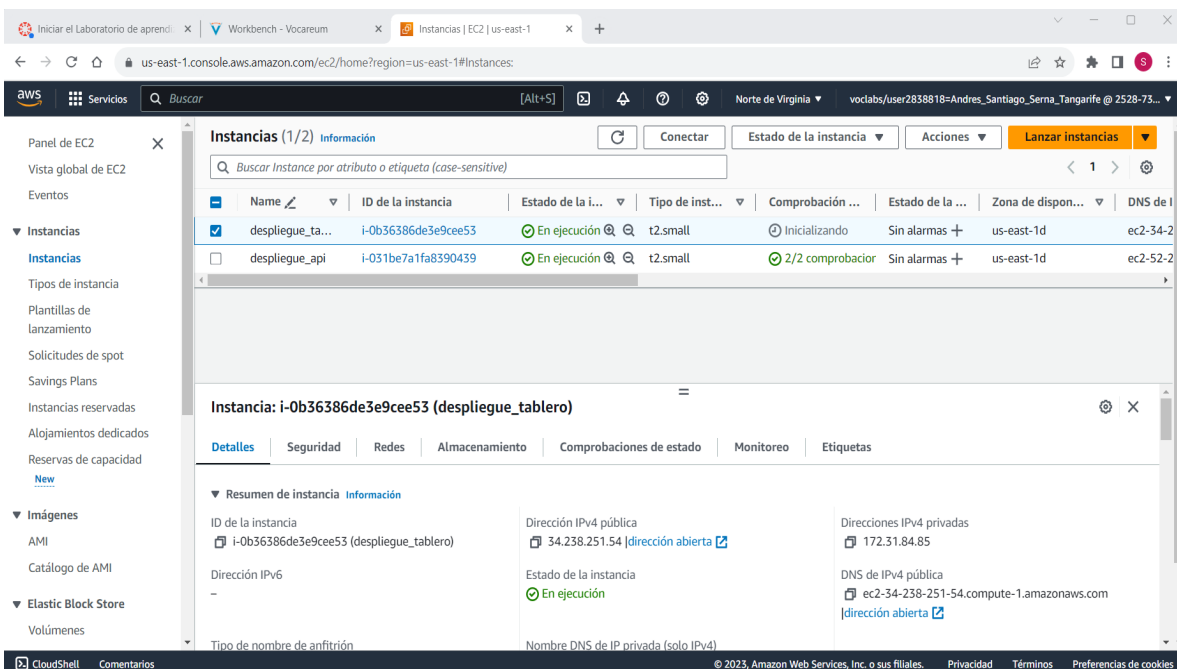


## 2. Disposición tecnológica del tablero

A continuación de define las siguientes herramientas necesarias para la configuración de despliegue del tablero.

1. Se crea en la consola de EC2 la instancia de t2.small, Ubuntu server con 20 GB de disco. Con el nombre despliegue\_tablero

### Máquina tablero



2. abrir la instancia en la terminal donde se encuentra la llave.pem del proyecto y la IP de la dirección pública de la máquina virtual del tablero.

```
ssh -i llave.pem ubuntu@IP_pública
```

Ejemplo `ssh -i llave.pem ubuntu@34.238.251.54`

3. En la máquina virtual, se elimina las versiones anteriores de Docker (esto genera un error si no hay versiones anteriores, en cuyo caso puede continuar sin problema)

```
sudo apt-get remove docker docker-engine docker .io containerd runc
```

4. Actualice el índice de paquetes

```
sudo apt-get update
```

5. Instale dependencias para verificar certificados (ca-certificates), obtener objetos con su URL (curl) y administrar llaves PGP (gnupg)

```
sudo apt-get install ca-certificates curl gnupg
```

6. Agregue la llave de Docker

```
sudo install -m 0755 -d /etc/apt/keyrings
```

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
```

7. Agregue el repositorio de Docker a su sistema para la instalación

```
echo \
"deb [arch=$(dpkg --print-architecture)] signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \
"$(. /etc/os-release && echo "$VERSION_CODENAME)" stable " | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

8. Actualizar nuevamente el índice de paquetes con este nuevo repositorio incluido

```
sudo apt-get update
```

9. Instalar Docker Engine, containers, y Docker Compose

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

10. En la terminal de la máquina virtual en EC2, clonar el repositorio de GitHub donde se encuentra en la carpeta del proyecto.

```
git clone https://github.com/asernat/miad_proyecto_despliegue.git
```

11. En la terminal abra el repositorio clonado

```
cd miad_proyecto_despliegue
```

12. A partir del archivo Dockerfile construya la imagen que usará más adelante para lanzar contenedores

```
sudo docker build -f Dockerfile.app -t avaluos-app:latest .
```

13. Ahora ejecute un contenedor usando la imagen creada con el comando; reemplazar la IP\_API por la IP pública de la API.

```
sudo docker run -p 8080:8080 -it -e PORT=8080 -e API_HOST=IP_API:8080 avaluos-app:latest
```

Ejemplo ejecución

```
sudo docker run -p 8080:8080 -it -e PORT=8080 -e API_HOST=35.175.246.254:8080 avaluos-app:latest
```

14. Ahora a la consola de EC2, seleccione su máquina virtual ([despliegue\\_tablero](#)), modificar el grupo de seguridad para permitir tráfico por el puerto 8080. En la pestaña de grupo de seguridad de la máquina y editar las reglas de entrada y agregue una que permita tráfico por el puerto TCP 8080 desde cualquier IP (anywhere IPv4).
15. Ejecución contenedor del tablero en máquina virtual, como se muestra a continuación

```
ubuntu@ip-172-31-84-85: ~/n x + v
[+] Building 46.7s (13/13) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile.app            0.1s
=> => transferring dockerfile: 609B                                0.1s
=> [internal] load .dockerignore                                   0.0s
=> => transferring context: 2B                                       0.0s
=> [internal] load metadata for docker.io/library/python:3.9      0.3s
=> [1/8] FROM docker.io/library/python:3.9@sha256:b6cc878074fdc6aff4486f742bf4cc6d18f4e71ed44027649856355b8e23db 0.0s
=> [internal] load build context                                   0.0s
=> => transferring context: 5.87kB                                    0.0s
=> CACHED [2/8] RUN adduser --disabled-password --gecos '' app-user 0.0s
=> CACHED [3/8] WORKDIR /opt/app                                    0.0s
=> [4/8] ADD ./app /opt/app/                                       0.1s
=> [5/8] RUN pip install --upgrade pip                             5.2s
=> [6/8] RUN pip install -r /opt/app/requirements.txt             32.2s
=> [7/8] RUN chmod +x /opt/app/run.sh                             0.5s
=> [8/8] RUN chown -R app-user:app-user ./                        0.5s
=> exporting to image                                              7.7s
=> => exporting layers                                              7.7s
=> => writing image sha256:4287aa41d3245e7df55d949dcca409ee60d4873575d97c32206bdf59be1e673b 0.0s
=> => naming to docker.io/library/avaluos-app:latest              0.0s
ubuntu@ip-172-31-84-85:~/miad_proyecto_despliegue$ sudo docker run -p 8080:8080 -it -e PORT=8080 -e API_HOST=52.207.233.81:8085 avaluos-app:latest
Dash is running on http://0.0.0.0:8080/

* Serving Flask app 'app'
* Debug mode: on
162.142.125.215 - - [18/Nov/2023 21:07:49] code 505, message Invalid HTTP version (2.0)
2023-11-18 21:07:53.091 | INFO | pages.predict:update_valor:211 - Response: {'errors': None, 'version': '0.0.1', 'predictions': [147723751.04]}
```

16. Se Copia la IP pública de su máquina ejemplo. [34.238.251.54](#) y en un navegador local navega en la página IP:8080 ([34.238.251.54:8080](#)) Allí debe aparecer el tablero

Acceso a tablero máquina virtual:

