**MANUAL DE DESPLIEGUE**

**API – DASHBOARD: READMISIÓN**

**Primera parte: despliegue de la API**

En esta primera parte, se usó una máquina virtual para desplegar allí la API empleando un contenedor Docker.

Paso 1: Lanzamiento de una Instancia EC2 en AWS

1. Se inicio sesión en la Consola de AWS y en el Panel de Control para lanzar una nueva instancia llamada empaq, seleccionando la AMI de Ubuntu Server y eligiendo la instancia t2.small.configurando el almacenamiento a 20 GB y creando una clave para conectarse a esta.

A screenshot of a computer

Description automatically generated

Paso 2: Conexión a la Instancia

1. Se abrió una terminal y se usó el siguiente comando para conectarte a la instancia:

ssh -i /path/to/llave.pem ubuntu@172-31-80-116

Paso 3: Instalación de Docker

1. Se eliminó posibles versiones anteriores de Docker con el siguiente comando:

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

1. Se actualizó el índice de los paquetes :

sudo apt-get update

1. Se Instaló las dependencias necesarias:

sudo apt -get install ca - certificates curl gnupg

1. Se agregó la llave GPG oficial 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

sudo chmod a+r / etc/apt/ keyrings / docker . gpg

1. Posteriormente, se agregó el repositorio Docker:

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

1. Se actualizó el índice de paquetes con el nuevo repositorio incluido y se instaló Engine, containerd, y Docker Compose:

sudo apt-get update

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

plugin docker - compose - plugin

1. Se verifico la instalación, descargue yejecución con la imagen hello-world

sudo docker run hello-world

Paso 4: Clonación de archivos , construcción y ejecución de la imagen Docker

1. Localmente se descomprimió el archivo llamado *despliegue-prediccion-readmision*.zip, en este, los archivos contenidos llamados *readmisión-api* y el archivo Dockerfile se movieron a una carpeta la cual se llamo *apis*, en esta carpeta se inició un repositorio Git y se conectó con un nuevo repositorio en GitHub utilizando los siguientes comandos (esto se realizó para conectar el repositorio local con el remoto)

cd apis

git init

git add .

git commit -m "Primer commit"

Al terminar se debe contener el archivo *readmision-api* y *Dockerfiler* .

1. Se regresó a la Máquina Virtual en EC2 y se clonó el repositorio.

git clone https://github.com/Caanar/despliegue-prediccion-readmision/tree/main/apis

1. Luego, se ingresó a la carpeta del repositorio desde de la terminal para verificar que los archivos *readmision-api* y *Dockerfiler* se.encuentran en el repositorio

cd apis

1. Se construyó en esta, la Imagen de Docker

sudo docker build -t readmision-api:latest .

1. Se listaron las imágenes de Docker:

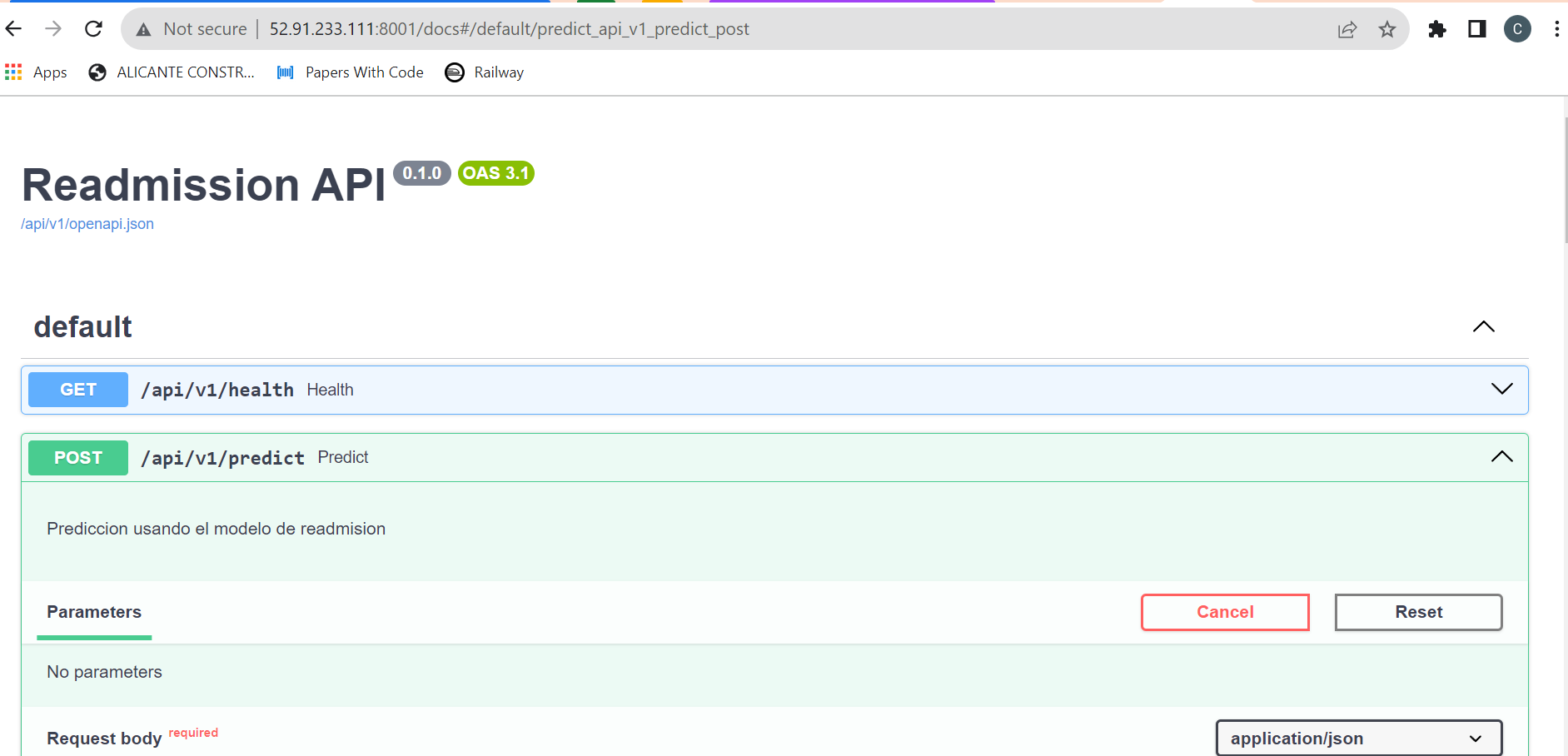
sudo docker images

1. Se ejecutó un contenedor utilizando la imagen creada:

sudo docker run -p 8001:8001 -it -e PORT=8001 readmision-api

1. Se Modificó el Grupo de Seguridad en la Consola de EC2. En la pestaña "Security", y se editó las reglas de entrada del grupo de seguridad asociado a la instancia agregando una regla que permitiera tráfico por el puerto TCP 8001 desde cualquier IP (anywhere IPv4).
2. Se Accedió a la API desde un Navegador, la IP pública de la máquina EC2. dirección 52.91.233.111:8001. Viendo el siguiente archivo en ejecución





A screenshot of a computer program

Description automatically generated

A black and white stripes

Description automatically generated

1. .Ahora ya instalado Docker en la instancia EC2. A continuación, se pasó a proceder a desplegar el API en el contenedor Docker en la máquina virtual.

**Segunda parte: despliegue del tablero**

Ahora en esta parte se usó otra máquina virtual para desplegar allí el tablero empleando un contenedor Docker.

Paso 1: Lanzamiento de una Instancia EC2 en AWS

1. Se inicio sesión en la Consola de AWS y en el Panel de Control para lanzar una nueva instancia llamada tablero-desp, seleccionando la AMI de Ubuntu Server y eligiendo la instancia t2.small.configurando el almacenamiento a 20 GB y creando una clave para conectarse a esta.

Paso 2: Conexión a la Instancia

1. Se abrió una terminal y se usó el siguiente comando para conectarte a la instancia:

ssh -i /path/to/llave.pem ubuntu@172-131-82-135

Paso 3: Instalación de Docker

1. Se eliminó posibles versiones anteriores de Docker con el siguiente comando:

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

1. Se actualizó el índice de los paquetes :

sudo apt-get update

1. Se Instaló las dependencias necesarias:

sudo apt -get install ca - certificates curl gnupg

1. Se agregó la llave GPG oficial 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

sudo chmod a+r / etc/apt/ keyrings / docker . gpg

1. Posteriormente, se agregó el repositorio Docker:

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

1. Se actualizó el índice de paquetes con el nuevo repositorio incluido y se instaló Engine, containerd, y Docker Compose:

sudo apt-get update

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

plugin docker - compose - plugin

1. Se verifico la instalación, descargue ye jecución con la imagen hello-world

sudo docker run hello-world

Paso 4: Clonación de archivos , construcción y ejecución de la imagen Docker

1. Localmente se descomprimió el archivo llamado *docker-dash-starter*.zip, en este, los archivos contenidos en esta carpeta llamados *app* y el archivo Dockerfile se movieron a una carpeta la cual se llamo *apis*, en esta carpeta se inició un repositorio Git y se conectó con un nuevo repositorio en GitHub utilizando los siguientes comandos (esto se realizó para conectar el repositorio local con el remoto)

cd apis

git init

git add .

git commit -m "Primer commit"

Al terminar se debe contener el archivo *app* y *Dockerfiler* .

1. Se regresó a la Máquina Virtual en EC2 y se clonó el repositorio.

git clone https://github.com/Caanar/despliegue-prediccion-readmision/tree/main/apis

1. Luego, se ingresó a la carpeta del repositorio desde de la terminal para verificar que los archivos *app* y *Dockerfiler* se.encuentran en el repositorio

cd apis

1. Se construyó en esta, la Imagen de Docker

sudo docker build -t readmision-api:latest .

1. Se listaron las imágenes de Docker:

sudo docker images

1. Se ejecutó un contenedor utilizando la imagen creada y se modifico el enlace donde laAPI\_URL se modificó por la IP pública de la maquina donde esta corriendo la API

sudo docker run -p 8050:8050 -it -e PORT=8001 -e API\_URL =44.211.148.23 app

1. Se Modificó el Grupo de Seguridad en la Consola de EC2. En la pestaña "Security", y se agrego la regla de entrada del grupo de seguridad asociado a la instancia agregando una regla que permitiera tráfico por el puerto TCP 8050 desde cualquier IP (anywhere IPv4).
2. Se Accedió IP pública de la máquina EC2. dirección 44.211.148.23:8050 del dashboard. Viendo la siguiente imagen de despliegue:

