Aquí tienes una guía paso a paso con todos los detalles y códigos que hemos usado para crear y ejecutar tu aplicación Python en un contenedor Docker, junto con los contenidos de los archivos involucrados:

---

## \*\*Guía para Crear y Ejecutar una Aplicación Python en Docker\*\*

### \*\*1. Crear el Proyecto Python\*\*

1. \*\*Estructura del Proyecto\*\*

Crea la estructura de directorios y archivos de tu proyecto:

```

my-python-app/

├── app/

│ └── \_\_init\_\_.py

├── Dockerfile

├── LICENSE.txt

├── README.md

├── requirements.txt

├── setup.cfg

├── setup.py

└── app.py

```

2. \*\*Contenido de los Archivos\*\*

\*\*`app/\_\_init\_\_.py`\*\*

Este archivo puede estar vacío o contener la inicialización del paquete.

```python

# app/\_\_init\_\_.py

```

\*\*`app.py`\*\*

```python

# app.py

print("Hello, Docker!")

```

\*\*`Dockerfile`\*\*

```Dockerfile

# Dockerfile

FROM python:3.9

# Copiar archivos al contenedor

COPY \*.py setup.cfg LICENSE.txt README.md requirements.txt setup.py /app/

COPY app /app/app/

# Establecer el directorio de trabajo

WORKDIR /app

# Instalar dependencias

RUN pip install -r requirements.txt

RUN pip install .

# Exponer el puerto

EXPOSE 8000

# Comando por defecto

CMD ["python", "app/app.py"]

```

\*\*`LICENSE.txt`\*\*

Contiene la licencia de tu proyecto.

```text

MIT License

```

\*\*`README.md`\*\*

Un archivo de texto que describe tu proyecto.

```markdown

# My Python App

This is a sample Python application.

```

\*\*`requirements.txt`\*\*

Lista las dependencias de tu proyecto.

```text

# requirements.txt

```

\*\*`setup.cfg`\*\*

Configuración del paquete.

```ini

[metadata]

name = my-python-app

version = 0.1

description = A sample Python application

author = Macazella

author\_email = magalicazellam@gmail.com

[options]

packages = find:

```

\*\*`setup.py`\*\*

Script para la instalación del paquete.

```python

# setup.py

from setuptools import setup, find\_packages

setup(

name='my-python-app',

version='0.1',

description='A sample Python application',

author='Macazella',

author\_email='magalicazellam@gmail.com',

packages=find\_packages(),

install\_requires=[

# Agrega las dependencias aquí si es necesario

],

)

```

### \*\*2. Construir la Imagen Docker\*\*

En la terminal, navega al directorio donde está tu Dockerfile y ejecuta:

```powershell

docker build -t my-python-app .

```

### \*\*3. Ejecutar el Contenedor\*\*

Ejecuta el contenedor basado en la imagen que acabas de construir:

```powershell

docker run -p 8000:8000 my-python-app

```

### \*\*4. Verificar en Docker Desktop\*\*

1. \*\*Abrir Docker Desktop\*\*

- \*\*Windows\*\*: Busca "Docker Desktop" en el menú de inicio y haz clic en él.

- \*\*Mac\*\*: Encuentra Docker Desktop en el Launchpad o en la carpeta de Aplicaciones.

2. \*\*Navegar a la Vista de Contenedores e Imágenes\*\*

- \*\*Contenedores\*\*: En el panel izquierdo, selecciona la pestaña "Containers" (Contenedores).

- \*\*Imágenes\*\*: Selecciona la pestaña "Images" (Imágenes).

3. \*\*Ver Detalles del Contenedor\*\*

- Haz clic en el contenedor para ver más detalles, como los registros y estadísticas de uso de recursos.

4. \*\*Interacción con Contenedores\*\*

- \*\*Logs\*\*: Ver los registros generados por tu aplicación.

- \*\*Exec\*\*: Abrir una terminal dentro del contenedor.

- \*\*Stop/Restart/Delete\*\*: Detener, reiniciar o eliminar contenedores.

5. \*\*Ver Información de la Imagen\*\*

- En la pestaña "Images", ver detalles de la imagen construida.

---