
Trabajo Integrador: Virtualización

Datos Generales

- **Título del trabajo:** Virtualización
- **Alumnos:**
 - Albertini Hugo Agustín - agustin_alber@hotmail.com
 - Calcatelli Renzo - rcalcatelli@gmail.com
- **Materia:** Arquitectura y Sistemas Operativos
- **Profesor:** Ariel Enferrel
- **Fecha de Entrega:** 05/06/2025

Índice

1. Introducción
2. Marco Teórico
3. Caso Práctico
4. Metodología Utilizada
5. Resultados Obtenidos
6. Conclusiones
7. Bibliografía
8. Anexos

1. Introducción

Este trabajo aborda la **virtualización** utilizando **VirtualBox**, una herramienta fundamental para crear entornos aislados donde ejecutar sistemas operativos invitados. La elección del tema surge de la necesidad de probar sistemas operativos alternativos (como Ubuntu) y tecnologías emergentes (como Docker) en un entorno seguro y controlado, sin afectar el sistema anfitrión.

Para el técnico en programación, dominar la virtualización es crucial, ya que permite:

- Testear aplicaciones en múltiples entornos.
- Simular servidores.
- Trabajar con tecnologías de contenedores.
- Garantizar la compatibilidad entre sistemas.

Estas habilidades son esenciales en el desarrollo e implementación de software.

Objetivos específicos:

- Instalar y configurar VirtualBox.
- Implementar Ubuntu como sistema operativo invitado.
- Utilizar Docker dentro de Ubuntu para ejecutar un programa Python mediante línea de comandos.

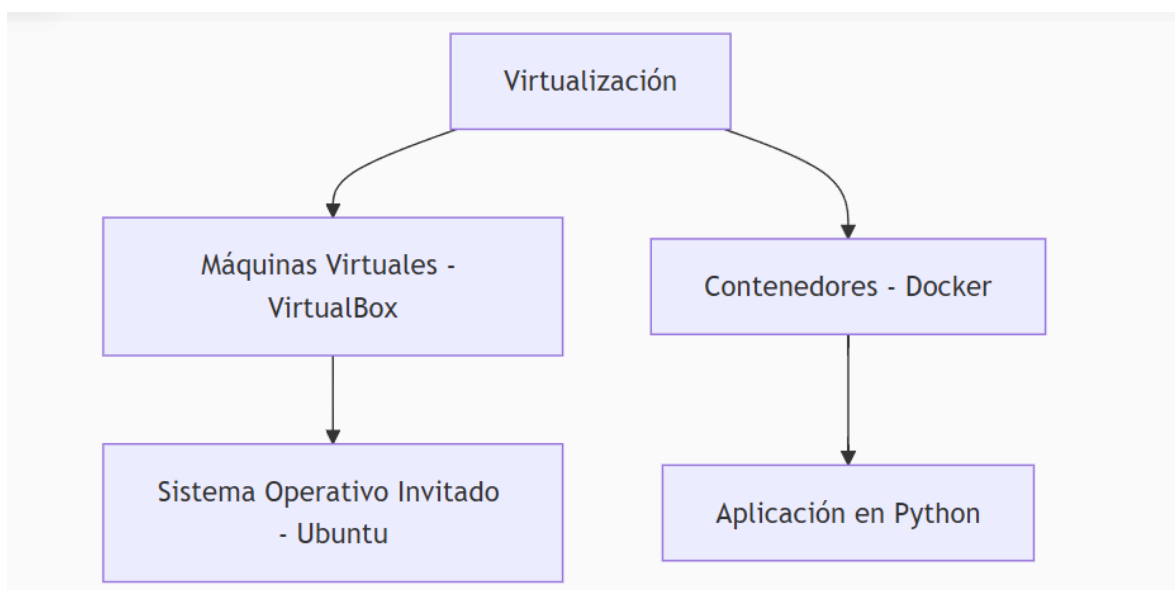
2. Marco Teórico

Definiciones Clave

- **VirtualBox:** Software de virtualización **Tipo 2 (Hosted)** desarrollado por Oracle. Permite crear y gestionar máquinas virtuales (VM) sobre sistemas operativos anfitriones (Windows, Linux, macOS).
- **Virtualización:** Técnica que abstrae recursos de hardware (CPU, RAM, disco) para crear entornos virtuales aislados. Existen distintos tipos:
 - **Virtualización Completa (Full Virtualization):** Simula hardware completo (ej. VirtualBox, VMware).
 - **Paravirtualización:** Requiere un sistema operativo invitado modificado para acceder directamente al hardware (ej. Xen).
 - **Contenedores:** Virtualización a nivel de sistema operativo, compartiendo el kernel del host (ej. Docker).

- **Sandbox:** Entorno aislado y restringido donde se pueden ejecutar aplicaciones potencialmente riesgosas sin afectar el sistema principal. Las máquinas virtuales y los contenedores actúan como sandboxes.
- **Ubuntu:** Distribución de Linux basada en Debian, enfocada en la facilidad de uso y el soporte comunitario. Es ideal para el desarrollo debido a su amplio soporte de herramientas y paquetes.
- **Docker:** Plataforma de contenedores que empaqueta aplicaciones y sus dependencias en unidades estandarizadas (**imágenes**). Permite una ejecución rápida y consistente en cualquier entorno.

Jerarquía de Tecnologías



Arquitectura de Docker



Tabla Comparativa: Virtualización vs. Contenedores

| Característica | Máquinas Virtuales (VirtualBox) | Contenedores (Docker) |
|------------------|----------------------------------|----------------------------------|
| Aislamiento | Hardware completo | Procesos del SO |
| Overhead | Alto (SO completo por VM) | Bajo (comparte kernel host) |
| Tiempo de inicio | Minutos | Segundos |
| Uso típico | Sistemas operativos heterogéneos | Microservicios/apps empaquetadas |

Fuentes Consultadas para el Marco Teórico

- **VirtualBox:** Documentación oficial de Oracle.
- **Docker:** Guías oficiales de Docker Inc.
- **Fuentes Adicionales:** Videos del profesor Ariel Enferrel (canal de YouTube) y material de estudio de la UTN.

3. Caso Práctico

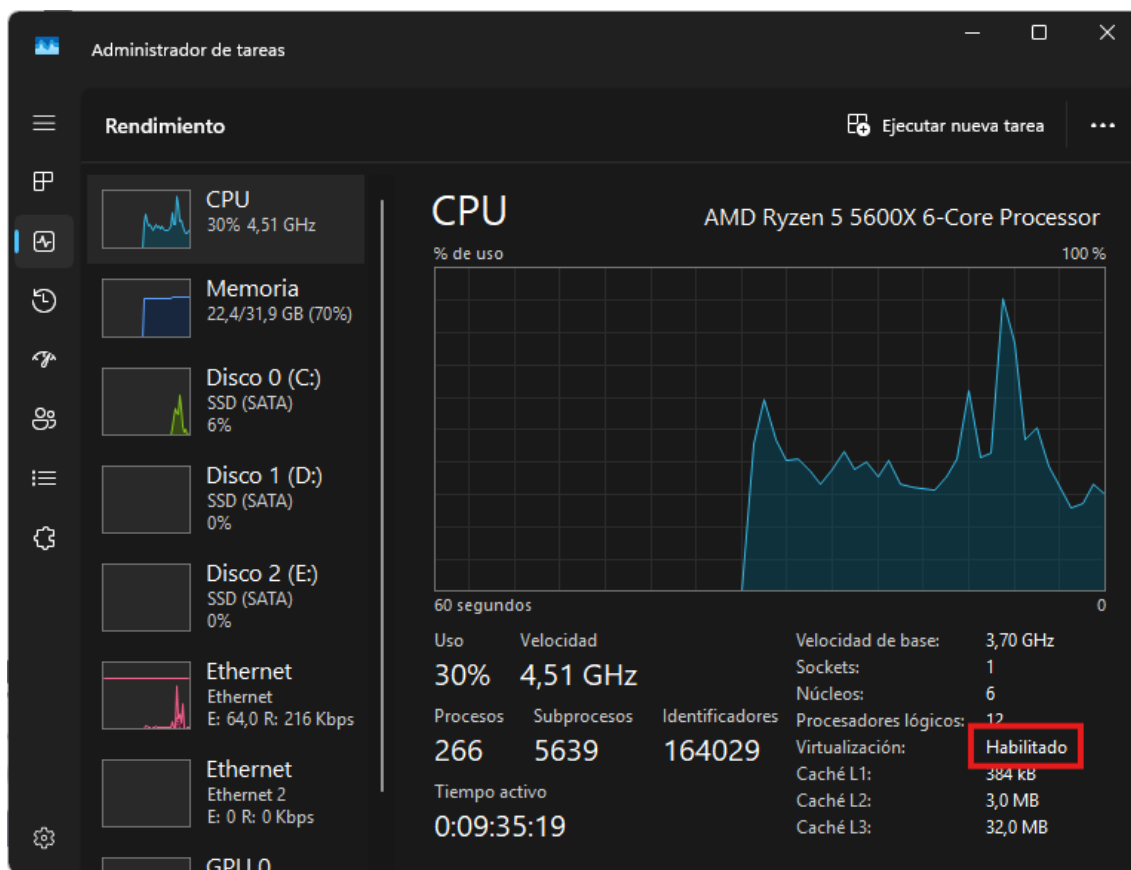
El objetivo de este proyecto fue instalar VirtualBox en una máquina con Windows para luego desplegar la última versión de Ubuntu como sistema operativo invitado. Una vez configurada la máquina virtual, se procedió a instalar Docker y Visual Studio Code. Finalmente, se desarrolló un programa para calcular promedios y se configuró el entorno para poder testearlo desde la terminal utilizando Docker.

Capturas del Paso a Paso en el Proyecto

Instalación de VirtualBox en Windows 10 Pro

1. Verificación de la Virtualización:

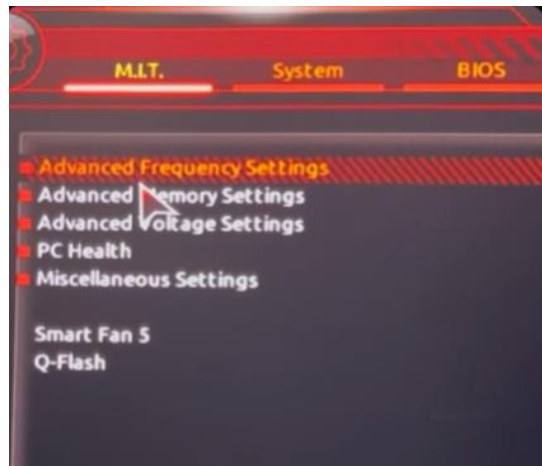
Ingresamos al Administrador de Tareas (Ctrl+Alt+Supr), seleccionamos la solapa "RENDIMIENTO" y verificamos si la virtualización está habilitada.



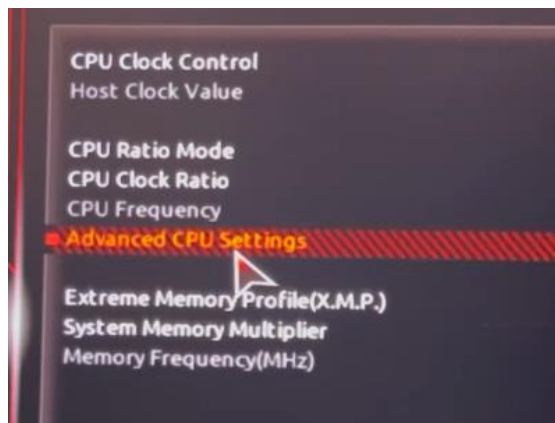
2. Habilitación de la Virtualización en BIOS (en caso de estar deshabilitada):

En caso de estar deshabilitada, se debe activar mediante la BIOS/UEFI. En este caso, fue necesario reiniciar la computadora e ingresar a la BIOS (presionando F2 en el arranque del sistema operativo).

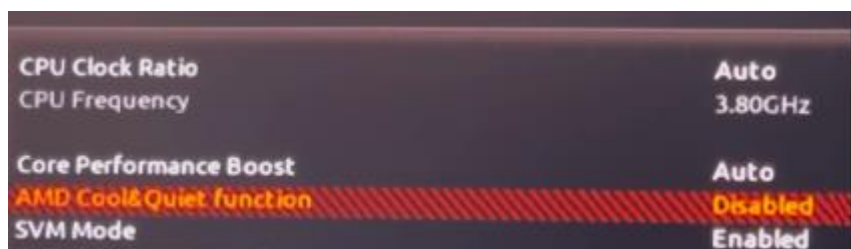
- **Primero:** Ingresamos en **Advanced Frequency Settings**



- **Segundo:** Ingresamos en **Advanced CPU Settings**.



- **Tercero:** Deberemos activar las opciones **AMD Cool Quiet Function** y **SVM Mode**, que por defecto estarán en "DISABLED". Deberemos cambiarlas a "ENABLED".



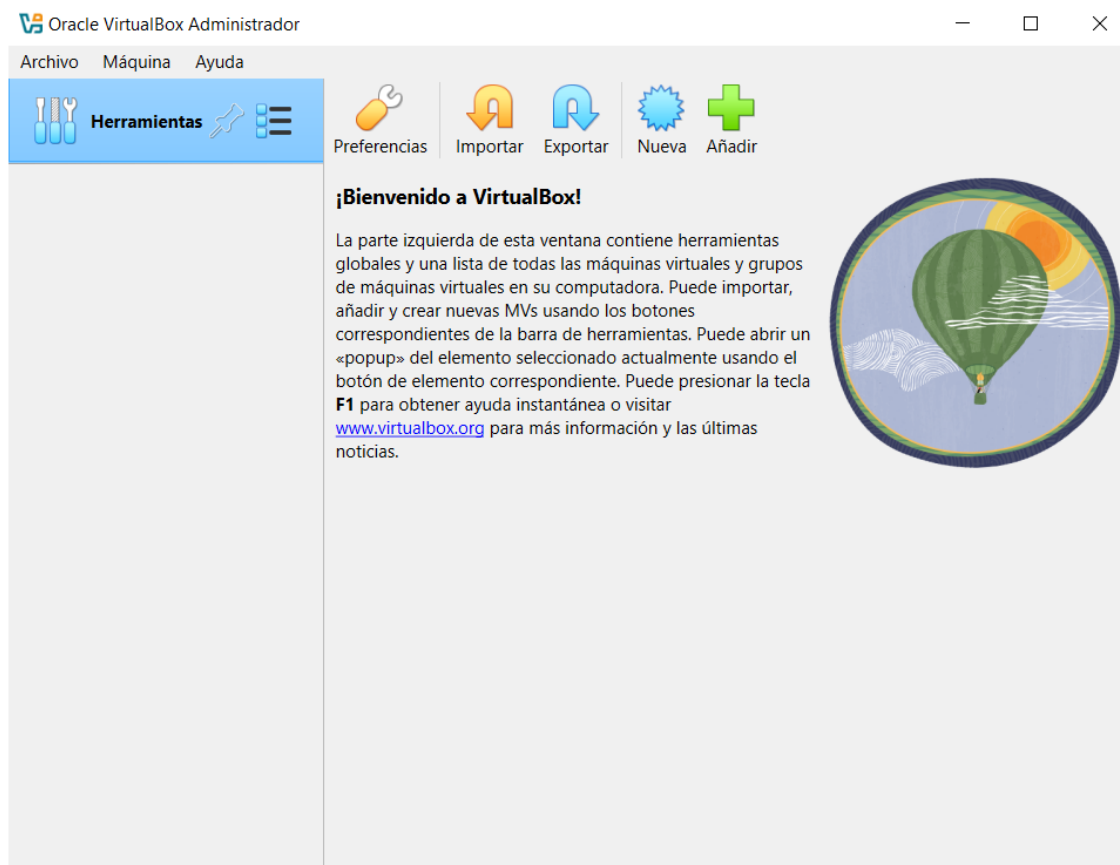
Al terminar, se debe guardar la configuración.

3. Descarga e Instalación de VirtualBox y Extension Pack:

Accedemos a la web oficial de VirtualBox (<https://www.virtualbox.org>), seleccionamos el sistema operativo anfitrión (en este caso, Windows) y también descargamos el VirtualBox Extension Pack.

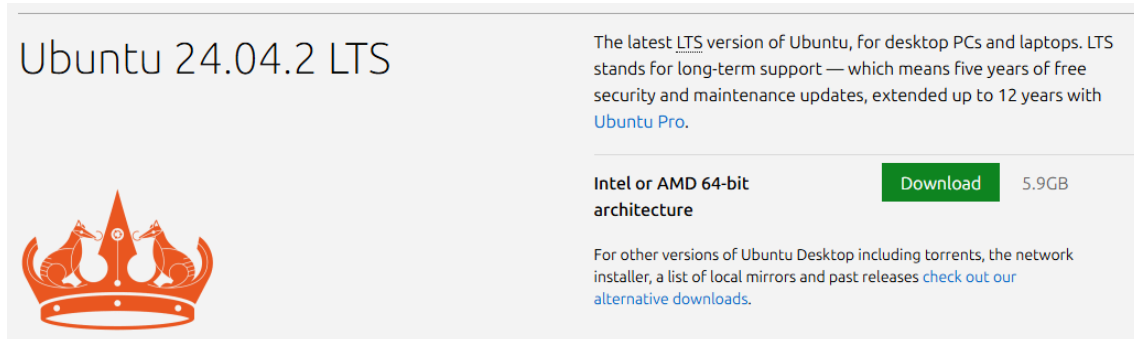


Una vez descargados, se ejecuta el instalador de VirtualBox, siguiendo los pasos indicados. Luego, se repite el proceso con la instalación del **Extension Pack**. Al finalizar la instalación, el programa se ejecutará.



4. Descarga de la Imagen ISO de Ubuntu:

Descargamos la imagen ISO del sistema operativo Ubuntu 24.04.2 LTS desde su página oficial (<https://ubuntu.com/download/desktop>).

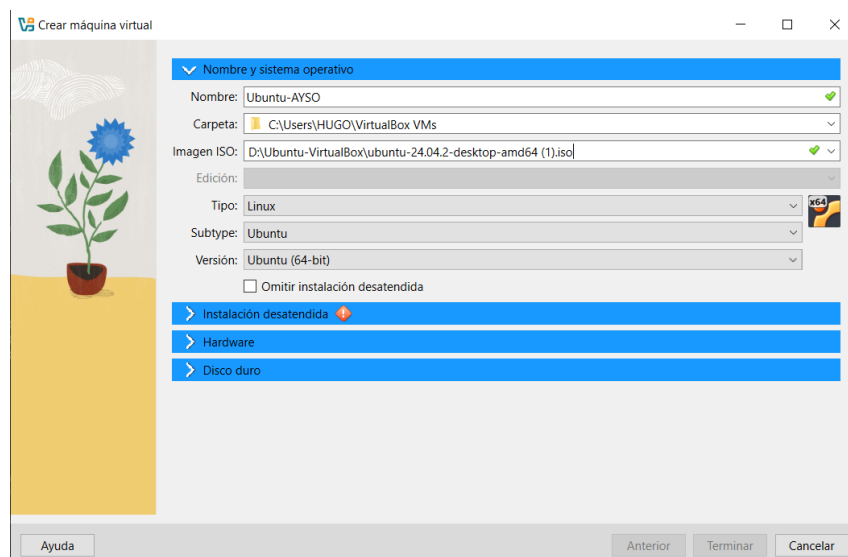


5. Instalación de Ubuntu dentro de VirtualBox:

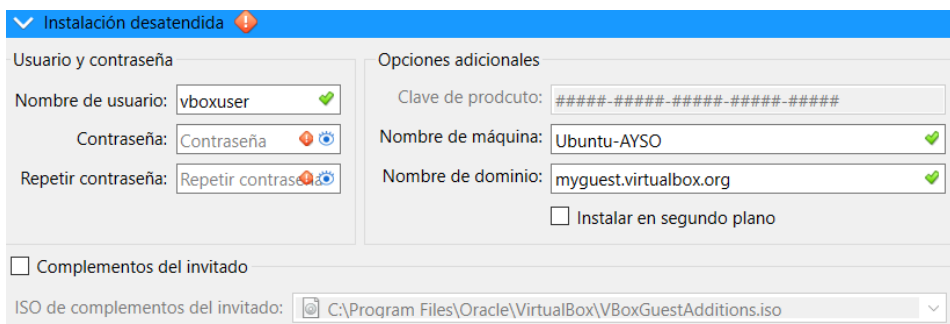
- **Primer paso:** Ingresamos en la solapa de "Nueva" en la parte superior del programa VirtualBox.



- **Segundo paso:** Configuramos algunos datos para la instalación de Ubuntu:
 - **Nombre de la máquina:** (en este caso, "Ubuntu-AySO").
 - **Carpeta donde guardar los archivos:** Seleccionamos la ubicación dentro de nuestra computadora principal.
 - **Imagen ISO:** Seleccionamos la imagen ISO de Ubuntu descargada.



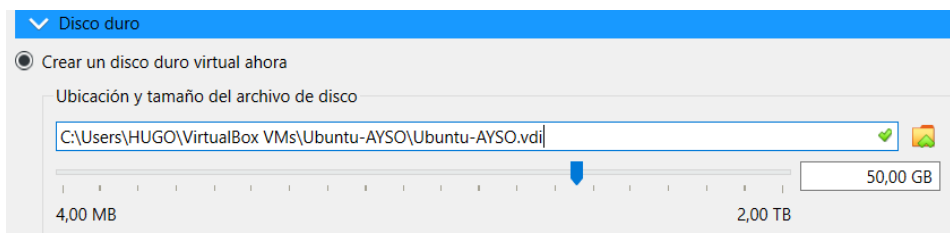
- **Tercer paso:** Por defecto, ya viene un nombre de usuario y contraseña. Tendremos que modificarlos a nuestro gusto.



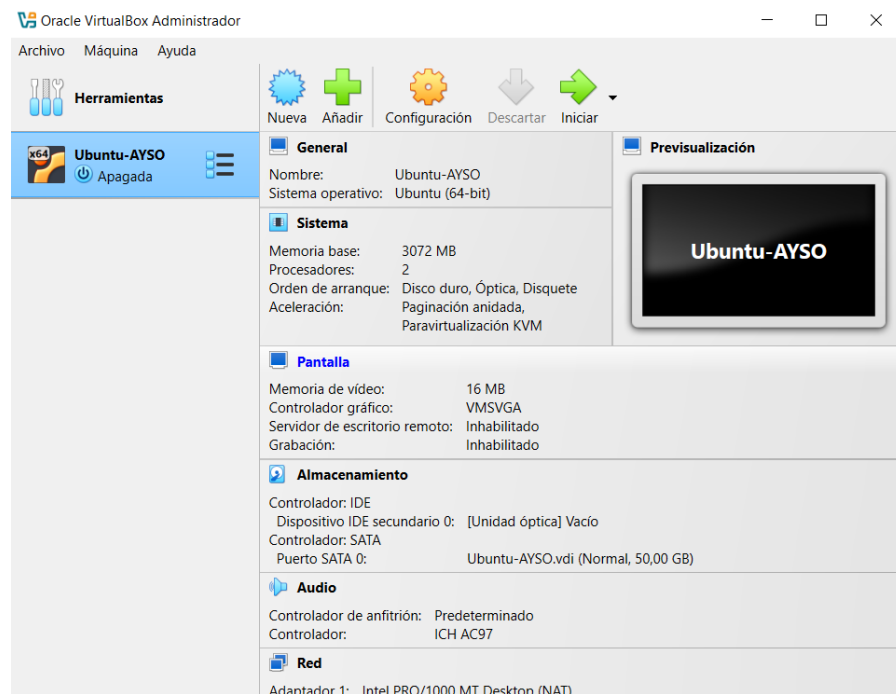
- **Cuarto paso:** Configuramos el **Hardware** (memoria RAM y núcleos de procesador) que queremos asignar desde la computadora principal a la máquina virtual. En este caso, se seleccionaron 3GB de memoria RAM y 2 núcleos de procesador (esto dependerá de los componentes de cada computadora).



- **Quinto paso:** Configuramos la cantidad de espacio en el disco local que vamos a otorgarle a la máquina virtual para las tareas realizadas dentro. En este caso, se prestaron 50 GB.

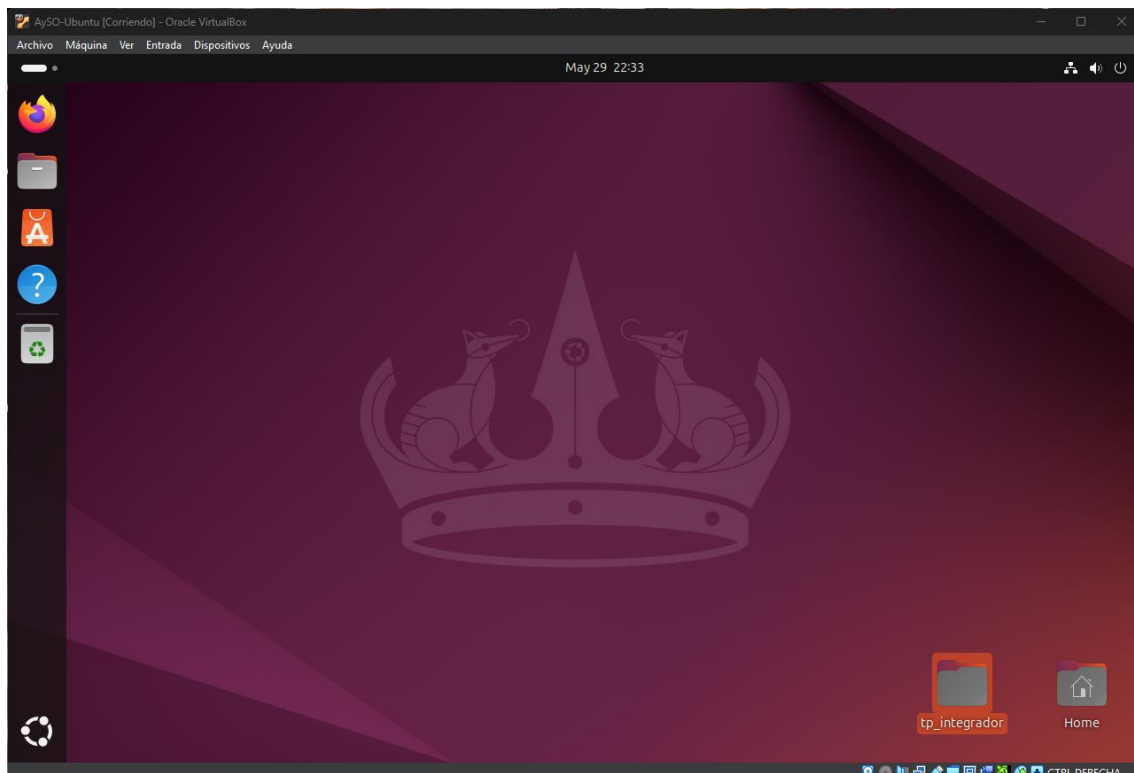


- **Por último:** Se abre una pestaña donde se ejecuta el sistema operativo Ubuntu, se procede a la instalación y, una vez terminada, ya podremos utilizarlo sin problemas. La nueva VM quedará guardada en VirtualBox.



6. Verificación del Funcionamiento de Ubuntu en la Máquina Virtual:

Dentro de la máquina virtual, podremos ver el sistema operativo Ubuntu funcionando correctamente.

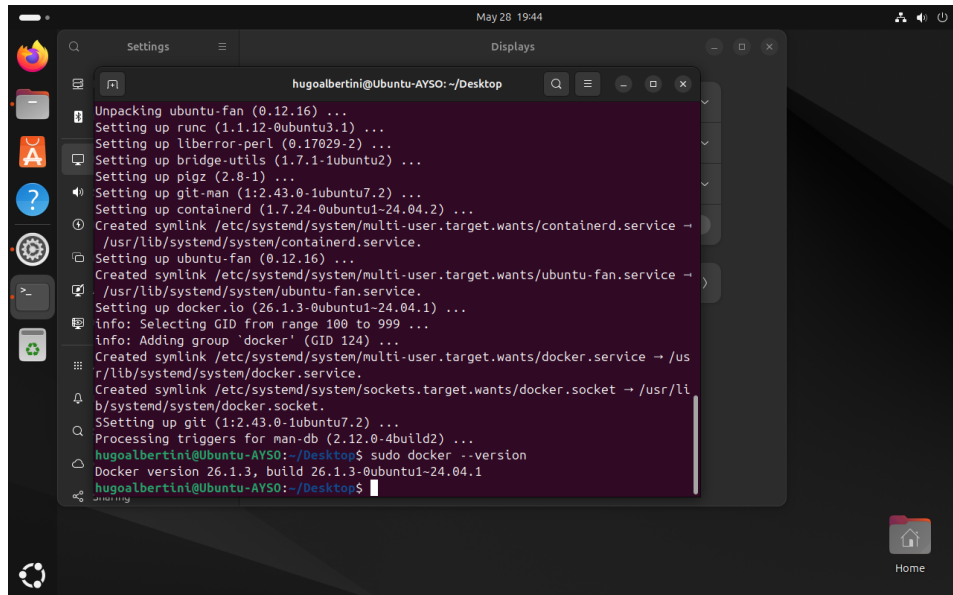


7. Instalación de Python, Docker y Visual Studio Code en Ubuntu:

- **Python:** Verificamos si la versión de Python (en este caso, 3.12.3) está instalada o la actualizamos según sea necesario.

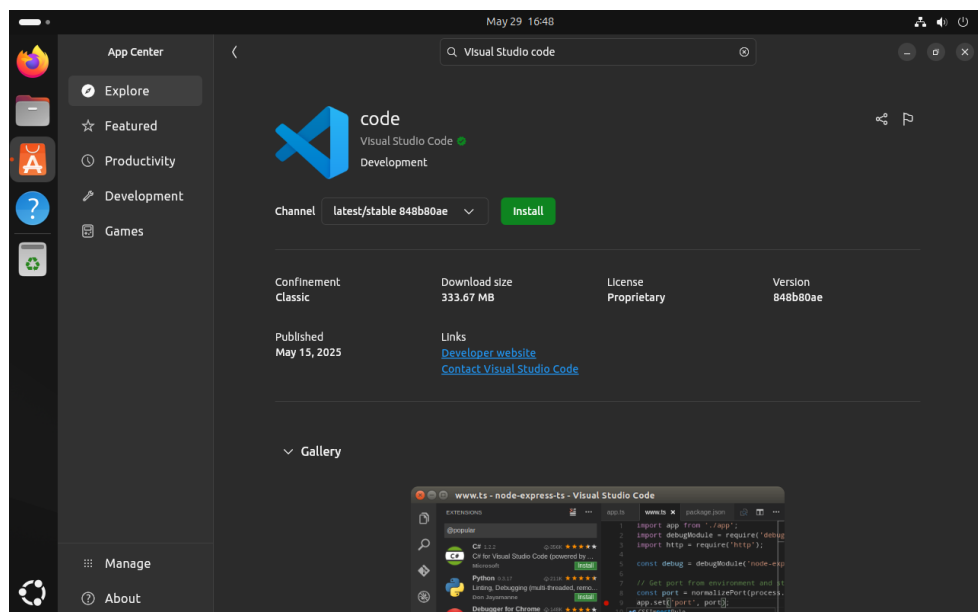
```
hugoalbertini@Ubuntu-AYSO:~/Desktop$ python3 --version
Python 3.12.3
hugoalbertini@Ubuntu-AYSO:~/Desktop$
```

- **Docker:** Utilizamos el comando `sudo apt install docker.io -y` para instalar Docker (versión 26.1.3).



```
hugoalbertini@Ubuntu-AYSO:~/Desktop$ sudo apt install docker.io -y
Unpacking ubuntu-fan (0.12.16) ...
Setting up runc (1.1.12-0ubuntu3.1) ...
Setting up liberror-perl (0.17029-2) ...
Setting up bridge-utils (1.7.1-1ubuntu2) ...
Setting up pigz (2.8-1) ...
Setting up git-man (1:2.43.0-1ubuntu7.2) ...
Setting up containerd (1.7.24-0ubuntu1-24.04.2) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /usr/lib/systemd/system/containerd.service.
Setting up ubuntu-fan (0.12.16) ...
Created symlink /etc/systemd/system/multi-user.target.wants/ubuntu-fan.service → /usr/lib/systemd/system/ubuntu-fan.service.
Setting up docker.io (26.1.3-0ubuntu1-24.04.1) ...
info: Selecting GID from range 100 to 999 ...
info: Adding group 'docker' (GID 124) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Setting up git (1:2.43.0-1ubuntu7.2) ...
Processing triggers for man-db (2.12.0-4build2) ...
hugoalbertini@Ubuntu-AYSO:~/Desktop$ sudo docker --version
Docker version 26.1.3, build 26.1.3-0ubuntu1-24.04.1
hugoalbertini@Ubuntu-AYSO:~/Desktop$
```

- **Visual Studio Code:** Ingresamos a la aplicación "APP CENTER", buscamos VS Code en su buscador y lo instalamos.



4. Metodología Utilizada

Pasos Seguidos en el Desarrollo del Proyecto

1. Investigación Previa y Preparación del Entorno

- **Fuentes Consultadas:**
 - Videos tutoriales del profesor Ariel Enferrel (guía principal para instalaciones y configuración).
 - **Documentación oficial:**
 - Guías de Docker Inc. (instalación y comandos de Docker).
 - Documentación de Oracle (uso de VirtualBox).
 - Soporte adicional con ChatGPT para resolver errores específicos en comandos de Docker y ejecución del programa.
- **Configuración del Entorno de Desarrollo:**
 - **Paso 1:** Habilitar la virtualización en el equipo físico.
 - **Paso 2:** Instalar Oracle VirtualBox.
 - **Paso 3:** Descargar la imagen ISO de Ubuntu.
 - **Paso 4:** Crear una máquina virtual (VM) en VirtualBox e instalar Ubuntu.
 - **Paso 5:** Verificar la instalación de Python 3.12 (preinstalado en Ubuntu).
 - **Paso 6:** Instalar Docker desde la terminal de Ubuntu.
 - **Paso 7:** Instalar Visual Studio Code (editor para el desarrollo del código).

2. Diseño y Prueba del Código

- **Desarrollo:**
 - Crear un programa en Python que solicite al usuario:
 - La cantidad de notas a calcular.
 - Cada nota.
 - El porcentaje correspondiente a cada nota.
 - Calcular el promedio ponderado final.

- **Pruebas:**
 - Ejecutar el programa dentro de Ubuntu para verificar su funcionalidad.
 - Validar que cumpliera con la consigna: manejo dinámico de notas y cálculos correctos.
 - Corregir errores detectados durante las pruebas.

3. Contenerización con Docker

- Crear un Dockerfile en la misma carpeta del código.
- Definir las instrucciones para construir la imagen:
 - Usar una imagen base de Python.
 - Copiar el código del programa.
 - Establecer el comando de ejecución.
- Solucionar errores de comandos con ayuda de ChatGPT (ej. permisos o sintaxis).
- Construir la imagen y ejecutar el contenedor para verificar el funcionamiento.

Herramientas y Recursos Utilizados

- **Virtualización:** Oracle VirtualBox + Ubuntu (SO invitado).
- **Lenguaje:** Python 3.12.
- **Entorno de Desarrollo:** Visual Studio Code (IDE).
- **Contenerización:** Docker.

Trabajo Colaborativo

- **Hugo Albertini:**
 - Documentación (preparación del PDF).
 - Capturas de pantalla de instalaciones y configuración.
 - Investigación teórica (cuadros comparativos y definiciones).
- **Renzo Calcatelli:**
 - Desarrollo y depuración del código en Python.
 - Pruebas del programa y Dockerización.
 - Solución de errores en contenedores.

5. Resultados Obtenidos

Logros del Caso Práctico

- ☒ Programa Funcional en Python:

Se desarrolló exitosamente un programa que:

- Solicita al usuario la cantidad de notas a calcular.
- Permite ingresar cada nota.
- Calcula el promedio de todas las notas.
- Muestra el resultado esperado.

- ☒ Contenerización con Docker:

El programa se empaquetó en un contenedor mediante un Dockerfile, permitiendo su ejecución portable y reproducible en cualquier entorno.

- ☒ Pruebas Exitosas:

El código y su versión dockerizada entregaron resultados numéricos correctos en todos los casos de prueba.

Aspectos que Funcionaron Correctamente

- Cálculo del Promedio:

El programa manejó correctamente diferentes combinaciones de notas.

- **Ejemplo:**
 - Notas a calcular: [3]
 - Notas: [8.1, 10, 6]
 - **Resultado:** 8.17

```
Materias a promediar?: 3

Ingresa las notas de cada materia:
Nota de la materia 1: 8.5
Nota de la materia 2: 10
Nota de la materia 3: 6

-- Resultado --
El promedio de tus 3 materias es: 8.17
rcalcatelli@AySO-Ubuntu:~/tp_integrador$
```

- **Dockerización:**

La imagen Docker se construyó sin errores. Para verificar las imágenes y contenedores creados, utilizamos el comando: `sudo docker image ls`

```
rccatelli@AySO-Ubuntu:~/Desktop/tp_integrador$ sudo docker image ls
REPOSITORY          TAG         IMAGE ID      CREATED        SIZE
calculadora-promedios latest      d71152fd8bfb  About a minute ago  1.03GB
python              3          07e1786f1f0a  2 weeks ago    1.02GB
python              latest     07e1786f1f0a  2 weeks ago    1.02GB
rccatelli@AySO-Ubuntu:~/Desktop/tp_integrador$
```

El contenedor ejecutó el programa interactivamente utilizando el comando: `sudo docker run -it --rm --name calculadora calculadora-promedios`.

```
rccatelli@AySO-Ubuntu:~/Desktop/tp_integrador$ sudo docker run -it --rm --name
calculadora calculadora-promedios
Materias a promediar?:
```

También se pudo ejecutar el programa sin crear el contenedor ni la imagen con el comando: `sudo docker run -it --rm -v "$(pwd)":/app python:3 python /app/promedio.py`

```
rccatelli@AySO-Ubuntu:~/tp_integrador$ sudo docker run -it --rm -v "$(pwd)":/a
pp python:3 python /app/promedio.py
Materias a promediar?: 3
```

Errores Detectados y Soluciones

- **Error:** Al utilizar el comando `sudo docker build -t promedio:1.0 .` arrojó un error.

```
rccatelli@AySO-Ubuntu:~/tp_integrador$ sudo docker build -t promedio:1.0 .
[sudo] password for rccatelli:
DEPRECATED: The legacy builder is deprecated and will be removed in a future rel
ease.

          Install the buildx component to build images with BuildKit:
          https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  4.096kB
Error response from daemon: the Dockerfile (dockerfile) cannot be empty
rccatelli@AySO-Ubuntu:~/tp_integrador$
```

- **Solución:** Se corrigió el nombre `promedio:1.0` por el nombre correcto del Dockerfile, que era `calculadora-promedios`.

6. Conclusiones

Aprendizajes Obtenidos

A lo largo de este trabajo práctico, adquirimos habilidades fundamentales para nuestro futuro como programadores. Aprendimos a utilizar **VirtualBox** como herramienta clave para crear máquinas virtuales (VM), lo que nos permitió realizar testeos seguros y explorar sistemas operativos distintos a Windows, en este caso, Ubuntu. Esta experiencia no solo amplió nuestro conocimiento sobre entornos Linux, sino que también nos familiarizó con operaciones avanzadas de terminal. Además, profundizamos en el uso de **Docker**, comprendiendo su utilidad para gestionar contenedores de forma eficiente y práctica.

Posibles Mejoras o Extensiones Futuras

Identificamos la necesidad de profundizar en Docker y en las instalaciones mediante terminal, ya que su dominio agiliza flujos de trabajo y optimiza recursos.

- Automatizar despliegues mediante scripts de terminal para reducir errores humanos.
- Experimentar con imágenes personalizadas de Docker para escenarios más complejos.

Dificultades que Surgieron y Cómo se Resolvieron

Enfrentamos dos desafíos principales:

- **Instalación de Docker:** Los comandos iniciales no funcionaron debido a diferencias entre versiones de software.
 - **Solución:** Investigamos guías actualizadas, comparamos documentación oficial y ajustamos los comandos según los requisitos de nuestra versión de Ubuntu.
- **Creación de imágenes en Docker:** Un error de sintaxis en los archivos de configuración (Dockerfile) impedía generar la imagen correctamente.
 - **Solución:** Revisamos la estructura del archivo, corregimos la sintaxis y validamos el proceso paso a paso hasta lograr el resultado esperado.

Como reflexión final, este trabajo no solo mejoró nuestras competencias técnicas, sino que también destacó la importancia de la adaptabilidad y el aprendizaje continuo en tecnologías emergentes.

7. Bibliografía

- Enferrel, A. (s.f.). *Virtualización - Conceptos y Beneficios.pdf*.
- UTN. (s.f.). *Hipervisor tipo 1 y tipo 2.pdf* (Material de estudio).
- Enferrel, A. (s.f.). *Virtualización de Recursos por un Hypervisor Tipo 2.pdf*.
- UTN. (s.f.). *Virtualización de recursos por un hipervisor tipo 2.pdf* (Material de estudio).
- Enferrel, A. (s.f.). Videos tutoriales.
<https://www.youtube.com/watch?v=tTU1n1ErWo0>
- Oracle. (s.f.). *VirtualBox Manual Oficial* <https://www.virtualbox.org/manual/>
- Docker Inc. (s.f.). *Guías Oficiales de Docker* <https://docs.docker.com/guides/>
- UTN. (s.f.). *Virtualización BIOS-UEFI.pdf* (Guía).