



TRABAJO INTEGRADOR
ARQUITECTURA Y SISTEMAS OPERATIVOS
TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN A DISTANCIA

Título del trabajo:

Virtualización con VirtualBox y desarrollo en Python

Alumnos:

Julio Cesar Roja - julioroja987@gmail.com

Cristian Emmanuel Rivero Corradi - cristianemmanuelrivero@gmail.com

Docente Titular:

Martin Aristiaran

Docente Tutor:

Oscar Londero

Fecha de entrega:

04/06/2025

1) INTRODUCCIÓN

La virtualización es una tecnología central en la informática moderna, ya que permite ejecutar múltiples sistemas operativos en una sola máquina física, optimizando recursos y favoreciendo la gestión y experimentación en entornos controlados.

El motivo de la elección de este tema radica en su impacto directo en la formación de técnicos en programación. El conocimiento sobre entornos virtuales es fundamental para quienes buscan desarrollarse profesionalmente en la administración de sistemas, el desarrollo y la prueba de software, así como en la implementación de soluciones informáticas seguras y eficientes.

Este trabajo integrador tiene como objetivo principal profundizar en los conceptos teóricos y prácticos de la virtualización. Se propone instalar y configurar una máquina virtual utilizando Oracle VirtualBox, instalar un sistema operativo Linux en ella y ejecutar un programa en Python desarrollado por el equipo. A través de estas actividades, se busca integrar los conocimientos adquiridos en la cursada, fortalecer competencias técnicas y comprender el valor de la virtualización como herramienta clave en el campo de la programación y la administración de sistemas.

2) MARCO TEÓRICO

¿Qué es la Virtualización?

La virtualización es una tecnología que crea versiones virtuales de recursos físicos como sistemas operativos, servidores, almacenamiento o redes. Permite ejecutar múltiples sistemas operativos en el mismo hardware, de manera aislada y eficiente.

Conceptos Básicos

Para lograrlo, se utiliza un hipervisor que abstrae el hardware y crea máquinas virtuales (VMs). Cada VM funciona como un equipo independiente.

Tipos de Virtualización

- **Virtualización de hardware:** Emula una máquina física completa.
- **Virtualización de software:** Ejecuta aplicaciones o escritorios en servidores remotos.

Tipos de Hipervisor

- **Tipo 1 (bare-metal):** Corre directamente sobre el hardware (ejemplo: VMware ESXi).
- **Tipo 2 (hosted):** Corre sobre un sistema operativo ya instalado (ejemplo: VirtualBox).

Estructura Jerárquica

- **Host:** Hardware físico.
- **SO:** Sistema operativo
- **Hipervisor tipo 2:** Administra las máquinas virtuales.
- **VM:** Máquina virtual creada.
- **Guest:** Sistema operativo que corre dentro de la VM.

Ventajas Clave

- **Aislamiento:** Mayor seguridad y posibilidad de pruebas sin riesgos.
- **Optimización de recursos:** Uso eficiente del hardware disponible.
- **Escalabilidad:** Facilidad para crear, duplicar o eliminar VMs.
- **Pruebas seguras:** Ideal para desarrollo y testeo.
- **Ahorro:** Reduce la necesidad de hardware físico adicional.

Sandbox

Una VM funciona como un entorno aislado y seguro donde se puede probar software sin afectar al sistema principal.

VirtualBox


VirtualBox es un hipervisor tipo 2, gratuito y multiplataforma, que permite la creación y administración de VMs. Entre sus funciones destacan el uso de snapshots, distintos modos de red y la posibilidad de compartir carpetas y portapapeles entre el host y el guest.

3) CASO PRÁCTICO

Descripción del Problema

En contextos donde no es posible modificar la instalación principal de un equipo (por restricciones de seguridad, políticas institucionales o para evitar conflictos de software), surge la necesidad de desarrollar y probar programas en entornos alternativos. En este caso, se requiere ejecutar y testear un script de Python en un sistema operativo Linux, garantizando un entorno aislado y seguro, sin afectar el sistema operativo Windows del equipo principal.

Descargamos Linux Mint <https://linuxmint.com/download.php>

linuxmint

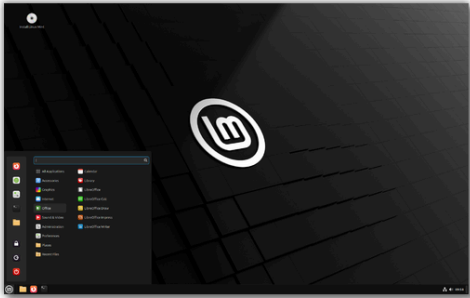
HomeDownloadProjectAboutLinksDonate

Linux Mint 22.1 'Xia'

Ready to download?

Linux Mint is available in different flavors. Choose an edition below.
For more information read the installation instructions.

[Installation Instructions](#)



Sleek, modern, innovative

Cinnamon Edition

The most popular version of Linux Mint is the Cinnamon edition. Cinnamon is primarily developed for and by Linux Mint. It is slick, beautiful, and full of new features.

[Download](#)[New Features](#)[Release Notes](#)

Descargamos VirtualBox para nuestro sistema operativo [Downloads – Oracle](#)

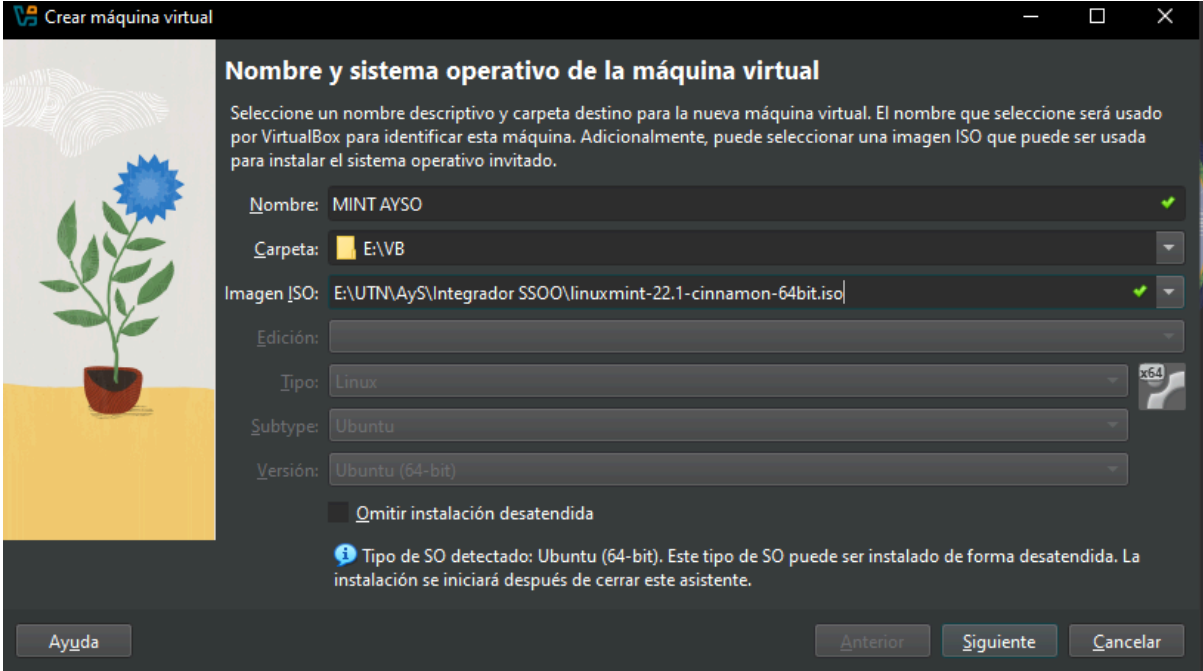
[VirtualBox](#)

The screenshot shows the Oracle VirtualBox website's download page. At the top, there's a navigation bar with links for Home, Download, Documentation, and Community, along with a search bar. The main heading is "Download VirtualBox". Below this, a paragraph states that the VirtualBox Extension Pack is available for personal and educational use under the PUEL license, and also available under commercial or enterprise terms. The page is divided into two main sections. The left section, titled "VirtualBox Platform Packages", lists VirtualBox 7.1.10 platform packages for various operating systems: Windows hosts, macOS / Intel hosts, macOS / Apple Silicon hosts, Linux distributions, Solaris hosts, and Solaris 11 IPS hosts. It also mentions that platform packages are released under the terms of the GPL version 3. The right section, titled "VirtualBox Extension Pack", shows the VirtualBox 7.1.10 Extension Pack. It includes a scrollable text area with the PUEL license details, a link to the FAQ, and three buttons: "PUEL License FAQ", "PUEL License Text", and "Accept and download".

Una vez instalado el software vamos a crear una nueva máquina virtual presionando el botón “Nueva”



Colocamos el nombre del sistema operativo, en este caso **MINT AYSO**



Crear máquina virtual

Nombre y sistema operativo de la máquina virtual

Seleccione un nombre descriptivo y carpeta destino para la nueva máquina virtual. El nombre que seleccione será usado por VirtualBox para identificar esta máquina. Adicionalmente, puede seleccionar una imagen ISO que puede ser usada para instalar el sistema operativo invitado.

Nombre: MINT AYSO ✓

Carpeta: E:\VB

Imagen ISO: E:\UTN\AyS\Integrador SSOO\linuxmint-22.1-cinnamon-64bit.iso ✓

Edición:

Tipo: Linux x64

Subtype: Ubuntu

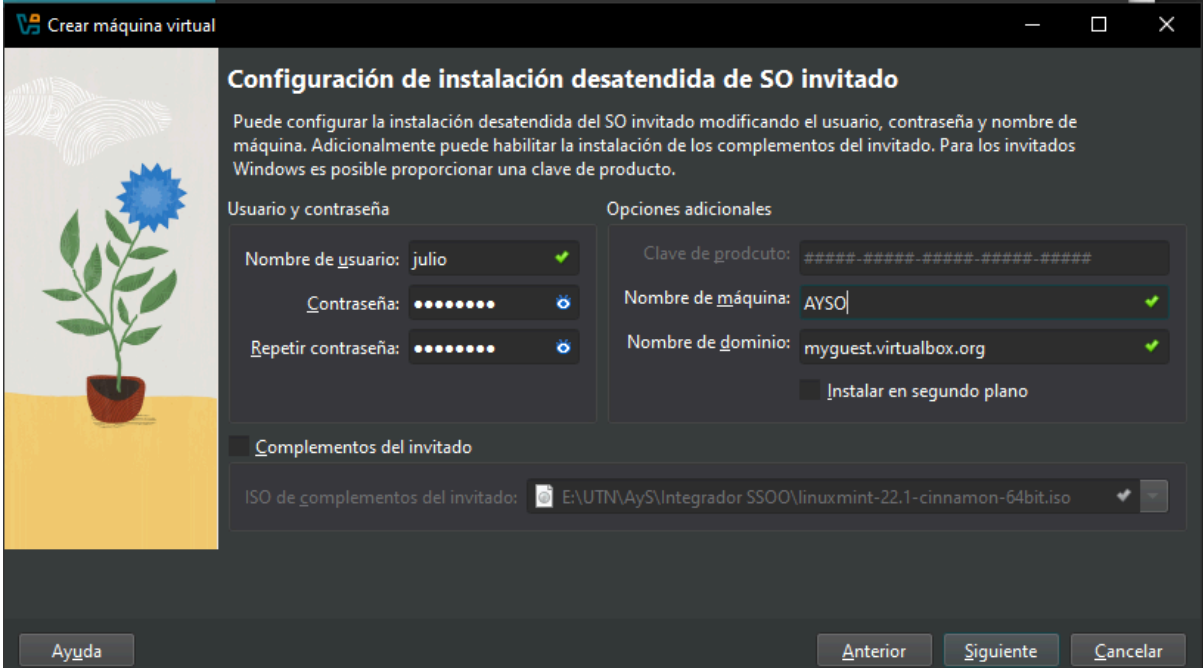
Versión: Ubuntu (64-bit)

☐ Omitir instalación desatendida

! Tipo de SO detectado: Ubuntu (64-bit). Este tipo de SO puede ser instalado de forma desatendida. La instalación se iniciará después de cerrar este asistente.

Ayuda Anterior Siguiente Cancelar

Colocamos un usuario y contraseña



Crear máquina virtual

Configuración de instalación desatendida de SO invitado

Puede configurar la instalación desatendida del SO invitado modificando el usuario, contraseña y nombre de máquina. Adicionalmente puede habilitar la instalación de los complementos del invitado. Para los invitados Windows es posible proporcionar una clave de producto.

Usuario y contraseña

Nombre de usuario: julio ✓

Contraseña: ●●●●●●●●

Repetir contraseña: ●●●●●●●●

Opciones adicionales

Clave de producto: #####-####-####-####-####

Nombre de máquina: AYSO ✓

Nombre de dominio: myguest.virtualbox.org ✓

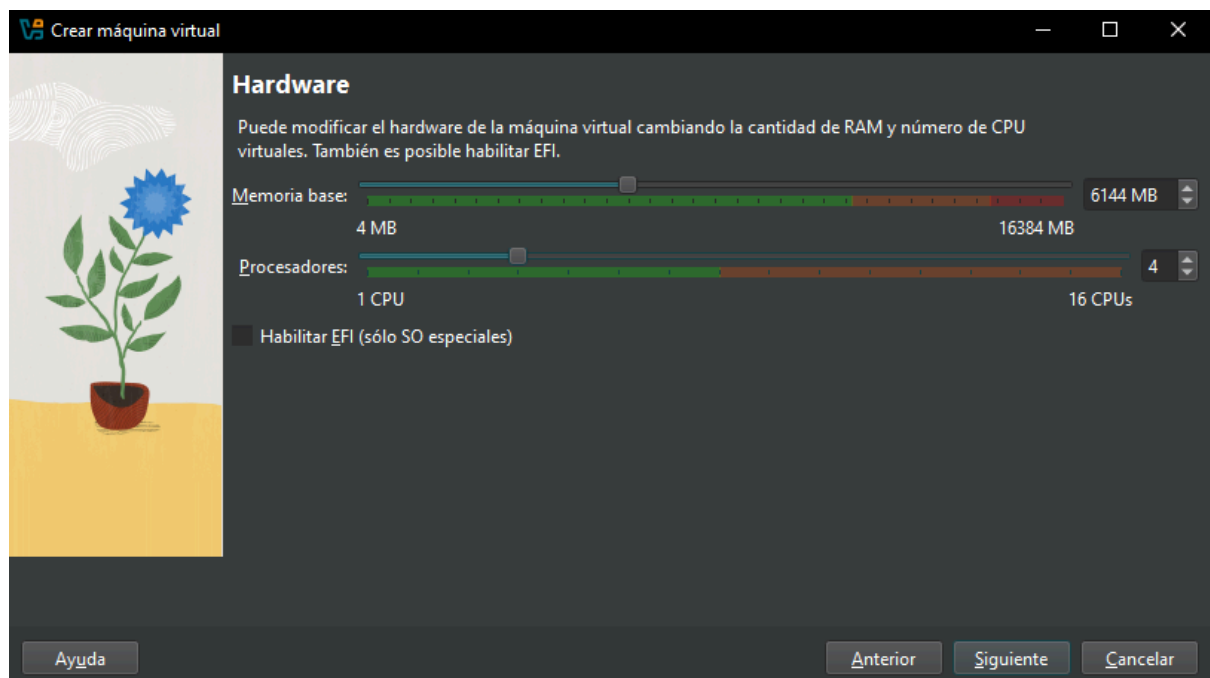
☐ Instalar en segundo plano

Complementos del invitado

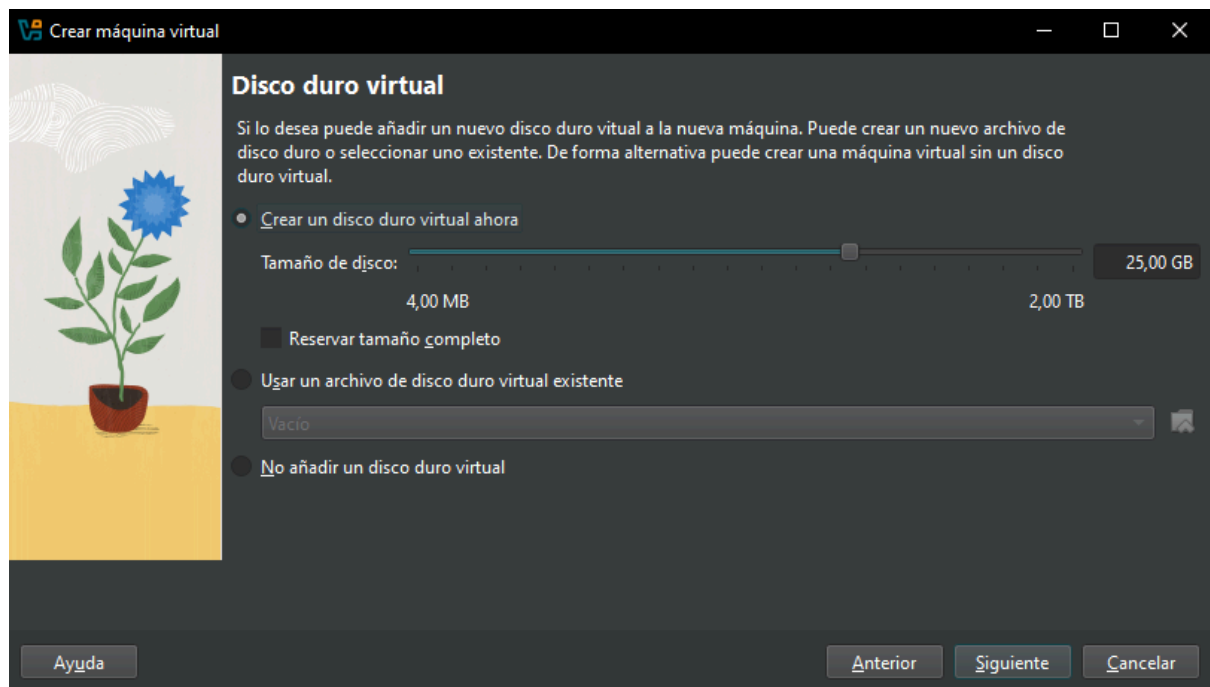
ISO de complementos del invitado: E:\UTN\AyS\Integrador SSOO\linuxmint-22.1-cinnamon-64bit.iso ✓

Ayuda Anterior Siguiente Cancelar

Elegimos cuanto memoria ram vamos a designar a la máquina, 6Gb



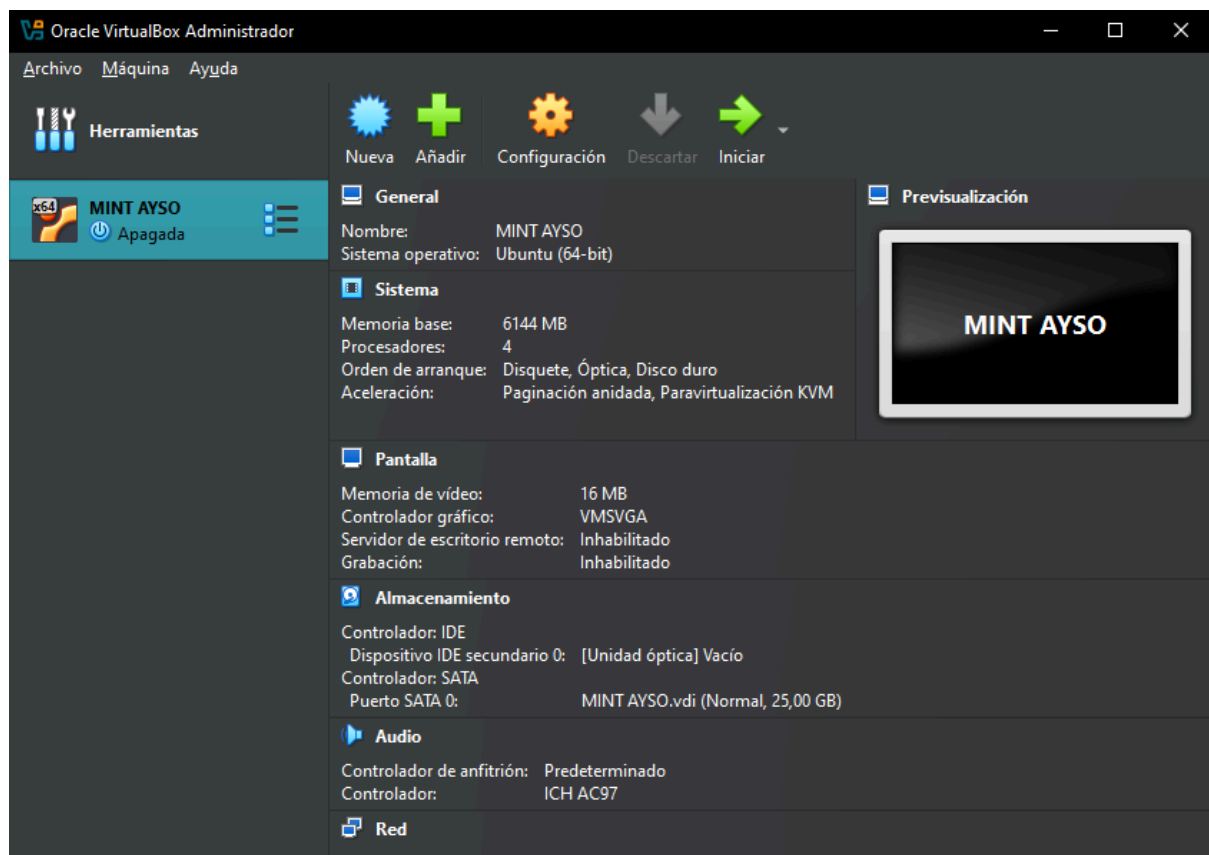
Determinamos la capacidad del disco rígido para la máquina virtual, 25GB



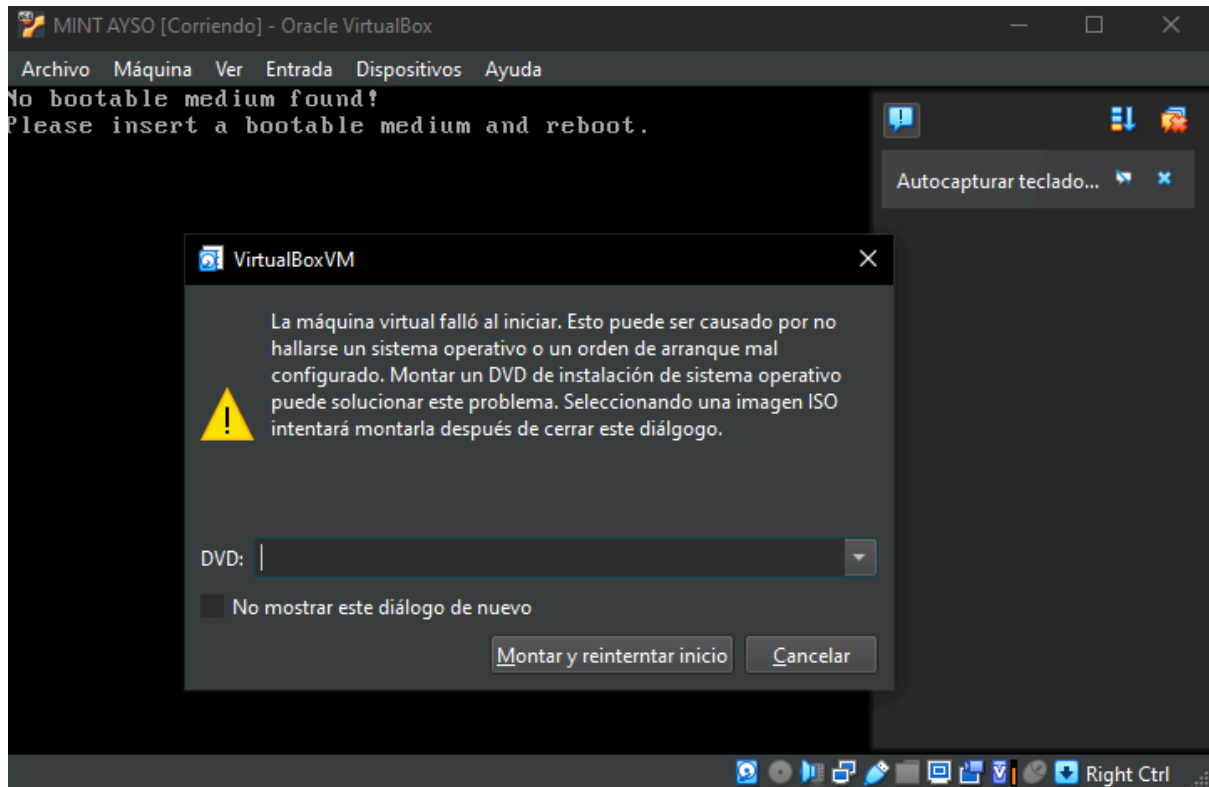
Vemos el resumen de lo seleccionado y presionamos **terminar**



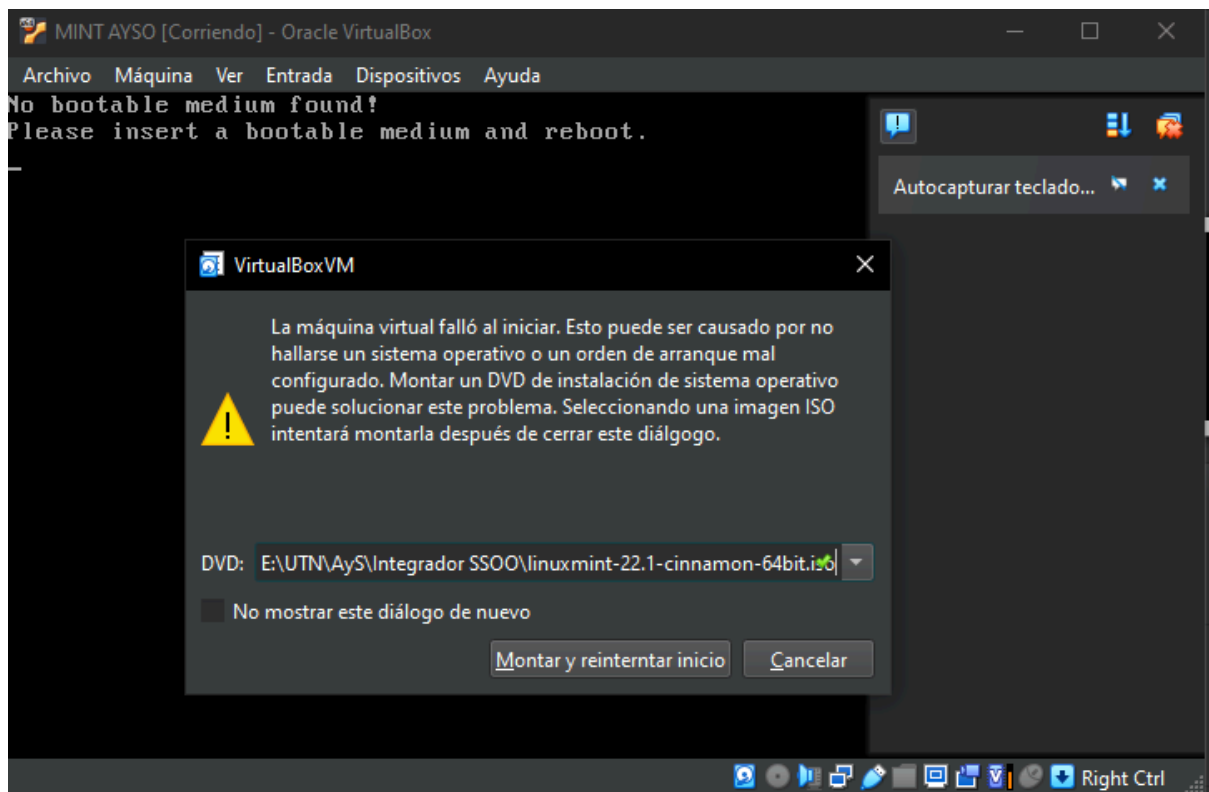
Ya con la máquina creada vamos a presionar **Iniciar**



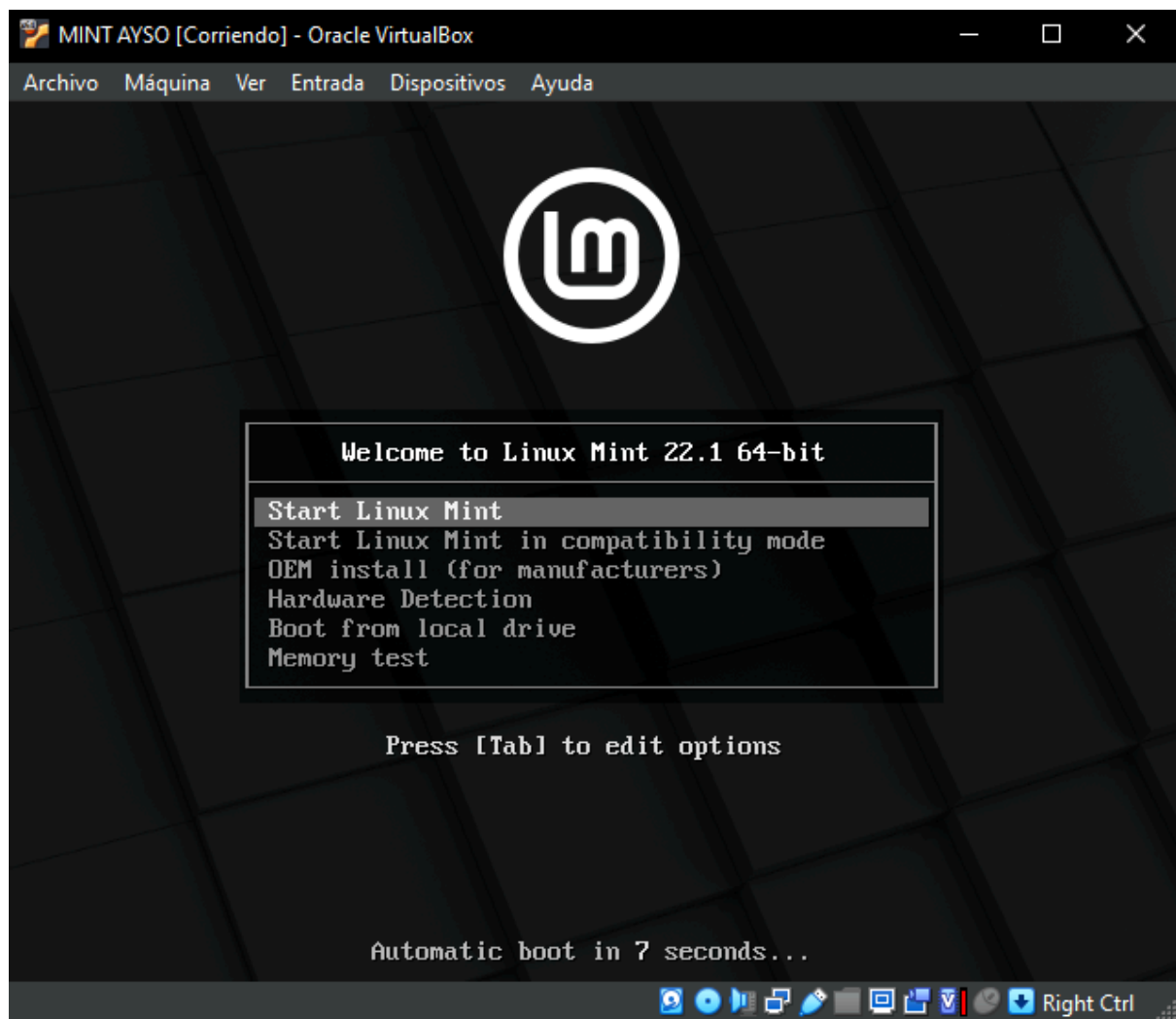
Nos va a pedir que coloquemos el ISO descargado para poder bootear el sistema operativo



Lo seleccionamos



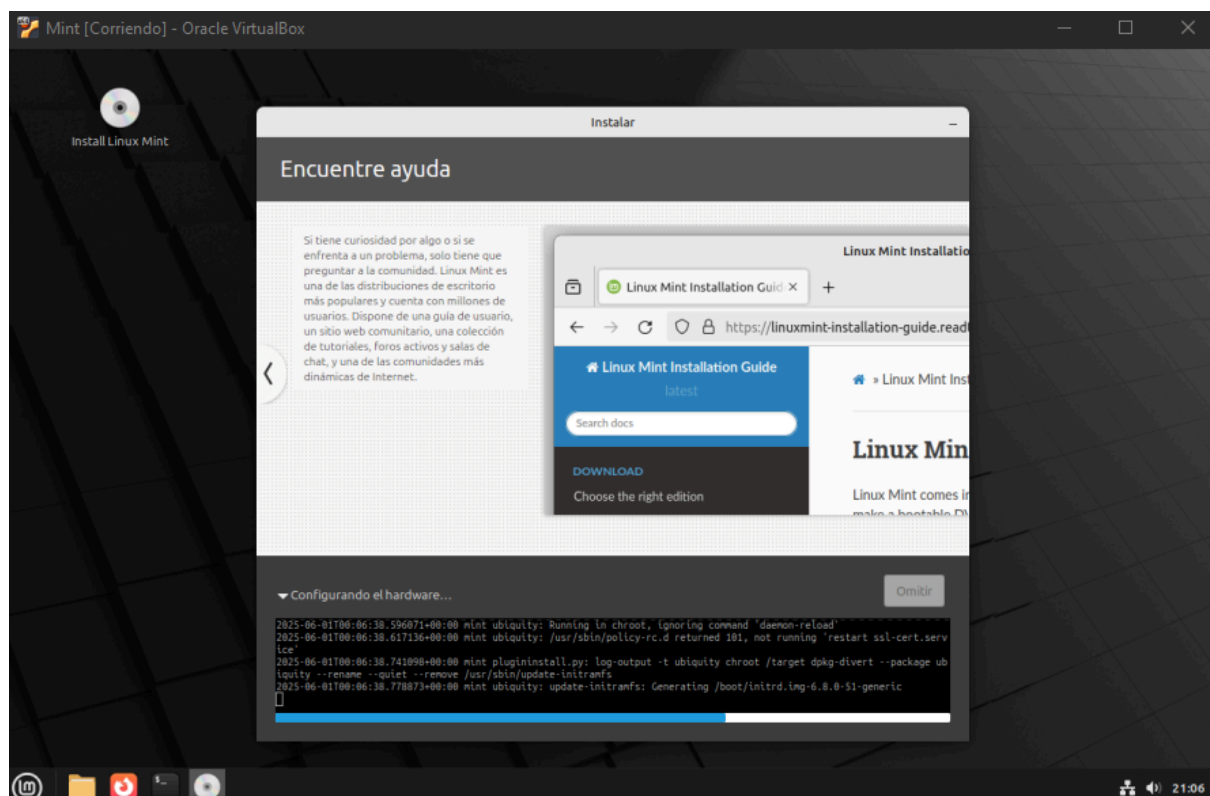
Comenzamos con la ejecución



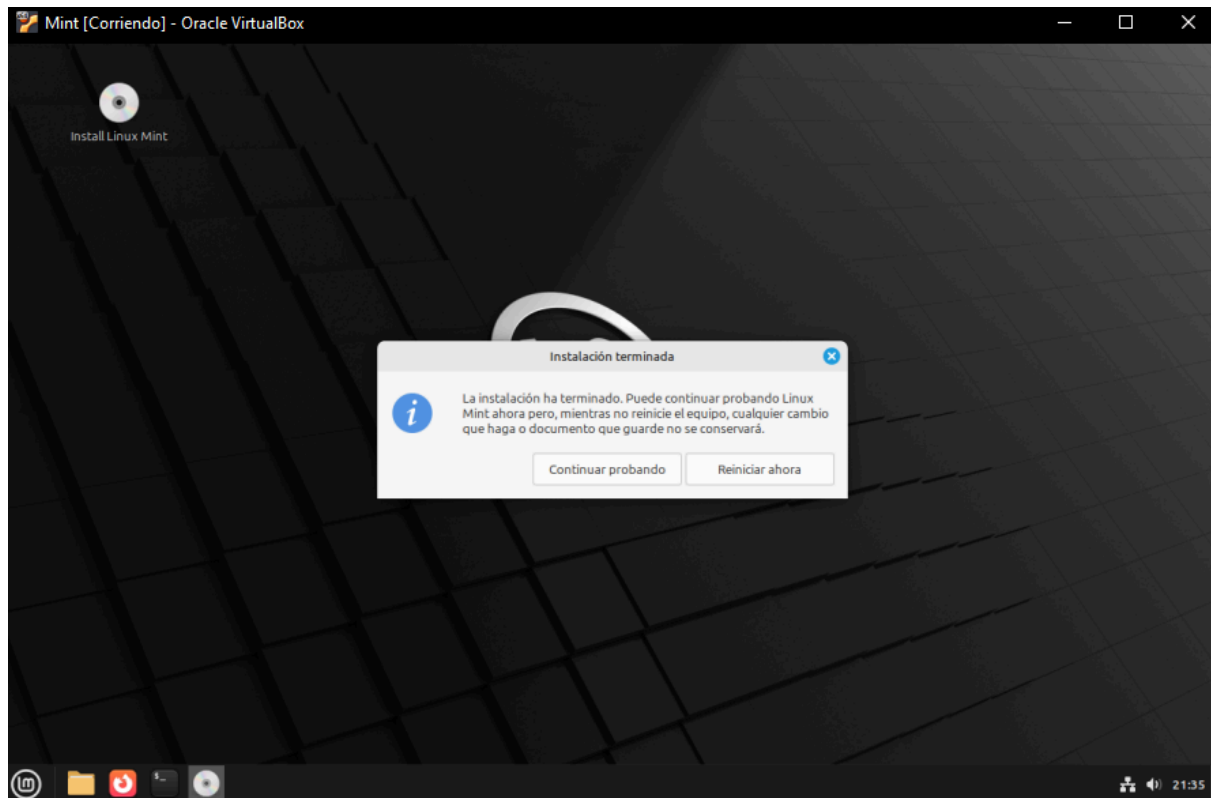
Se cargan los datos necesarios para el arranque

```
[ OK ] Started fstrim.timer - Discard unused filesystem blocks once a week.
[ OK ] Started fwupd-refresh.timer - Refresh fwupd metadata regularly.
[ OK ] Started logrotate.timer - Daily rotation of log files.
[ OK ] Started man-db.timer - Daily man-db regeneration.
[ OK ] Started mtd-news.timer - Message of the Day.
[ OK ] Started plocate-updatedb.timer - Update the plocate database daily.
[ OK ] Started systemd-tmpfiles-clean.timer - Daily Cleanup of Temporary Directories.
[ OK ] Reached target paths.target - Path Units.
[ OK ] Reached target timers.target - Timer Units.
[ OK ] Listening on avahi-daemon.socket - Avahi mDNS/DNS-SD Stack Activation Socket.
[ OK ] Listening on cups.socket - CUPS Scheduler.
[ OK ] Listening on dbus.socket - D-Bus System Message Bus Socket.
[ OK ] Listening on uuid.socket - UUID daemon activation socket.
[ OK ] Reached target sockets.target - Socket Units.
[ OK ] Reached target basic.target - Basic System.
Starting accounts-daemon.service - Accounts Service...
anacron.service
[ OK ] Started anacron.service - Run anacron Jobs.
Starting avahi-daemon.service - Avahi mDNS/DNS-SD Stack...
Starting bluedevil.service - Bluetooth management mechanism...
cron.service
[ OK ] Started cron.service - Regular background program processing daemon.
Starting dbus.service - D-Bus System Message Bus...
[ OK ] Started dmesg.service - Save initial kernel messages after boot.
dmesg.service
Starting e2scrub_reap.service - Remove Stale Online ext4 Metadata Check Snapshots...
[ OK ] Reached target getty.target - Login Prompts.
Starting gpu-manager.service - Detect the available GPUs and deal with any system changes...
irqbalance.service
[ OK ] Started irqbalance.service - irqbalance daemon.
Starting lm-sensors.service - Initialize hardware monitoring sensors...
[ OK ] Started mintsytem.service.
mintsytem.service
Starting polkit.service - Authorization Manager...
Starting power-profiles-daemon.service - Power Profiles daemon...
Starting rsyslog.service - System Logging Service...
Starting switcheroo-control.service - Switcheroo Control Proxy service...
Starting systemd-logind.service - User Login Management...
[ OK ] Started touchegg.service - Touchegg Daemon.
touchegg.service
Starting ubiquity.service - Linux Mint live CD installer...
Starting ubuntu-system-adjustments.service - Ubuntu system adjustments...
Starting udisks2.service - Disk Manager...
[ OK ] Finished e2scrub_reap.service - Remove Stale Online ext4 Metadata Check Snapshots.
lm-sensors.service
[ OK ] Finished lm-sensors.service - Initialize hardware monitoring sensors.
[ OK ] Finished ubiquity.service - Linux Mint live CD installer.
[ OK ] Started systemd-logind.service - User Login Management.
systemd-logind.service
```

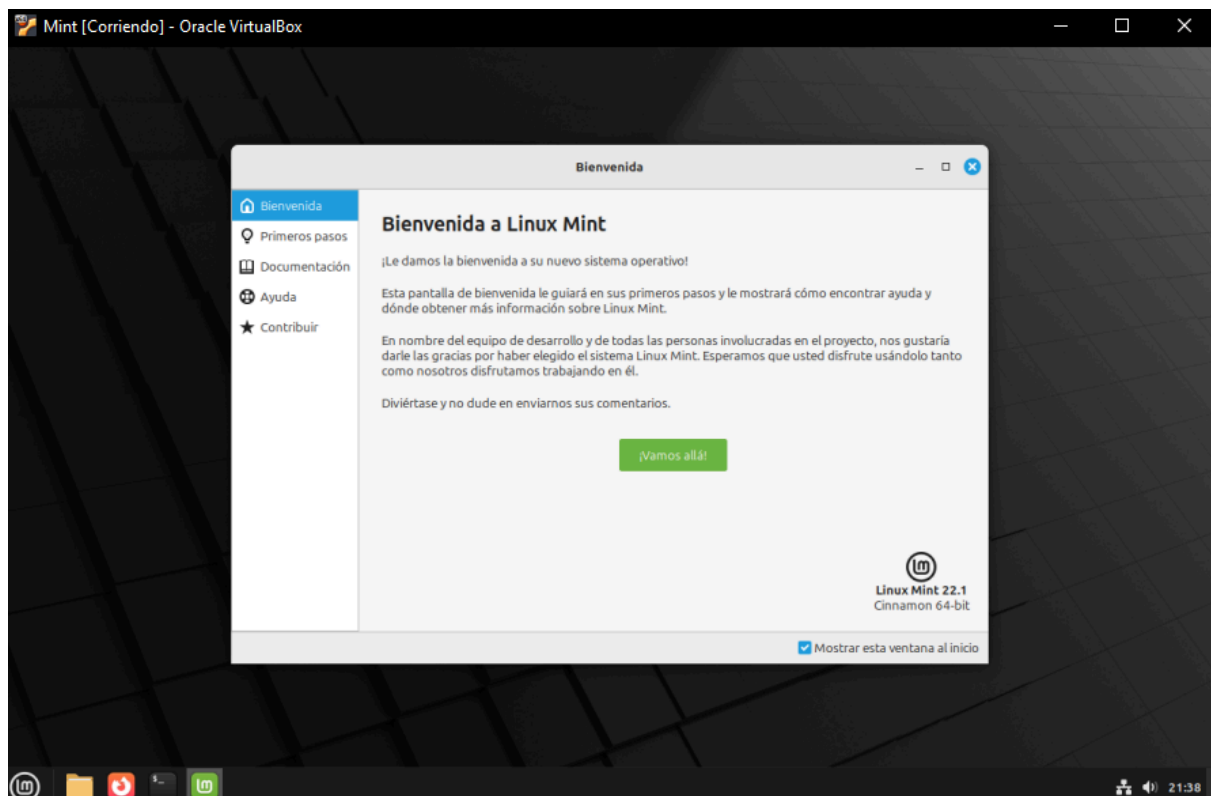
Una vez finalizado podemos ver un preview del sistema operativo el cual para continuar deberemos presionar ***Install Linux Mint***




Una vez finalizada la instalación se reinicia la máquina para tomar los cambios








Una vez iniciada ya estamos en condiciones de usarla




En github creamos un repositorio el cual más adelante hacemos un git clone

 **ayso-integrador-virtualizacion** Public

 Pin  Watch 0  Fork 0  Star 0

**Start coding with Codespaces**
Add a README file and start coding in a secure, configurable, and dedicated development environment.
[Create a codespace](#)

**Add collaborators to this repository**
Search for people using their GitHub username or email address.
[Invite collaborators](#)

Quick setup — if you've done this kind of thing before
[Set up in Desktop](#) or [HTTPS](#) [SSH](#) <https://github.com/julioxr/ayso-integrador-virtualizacion.git>
Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# ayso-integrador-virtualizacion" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/julioxr/ayso-integrador-virtualizacion.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/julioxr/ayso-integrador-virtualizacion.git
git branch -M main
git push -u origin main
```

Se inicia git y se suben todos los archivos al repositorio de github

```
Julio@DESKTOP-ROULFRS MINGW64 /e/UTN/AyS/Integrador_virtualizacion
• $ git init
Initialized empty Git repository in E:/UTN/AyS/Integrador_virtualizacion/.git/

Julio@DESKTOP-ROULFRS MINGW64 /e/UTN/AyS/Integrador_virtualizacion (master)
• $ git add .

Julio@DESKTOP-ROULFRS MINGW64 /e/UTN/AyS/Integrador_virtualizacion (master)
• $ git commit -m "Primer commit"
[master (root-commit) 4adf3be] Primer commit
 2 files changed, 82 insertions(+)
 create mode 100644 README.md
 create mode 100644 script_notas.py

Julio@DESKTOP-ROULFRS MINGW64 /e/UTN/AyS/Integrador_virtualizacion (master)
• $ git branch -M main

Julio@DESKTOP-ROULFRS MINGW64 /e/UTN/AyS/Integrador_virtualizacion (main)
• $ git remote add origin https://github.com/julioxr/ayso-integrador-virtualizacion.git

Julio@DESKTOP-ROULFRS MINGW64 /e/UTN/AyS/Integrador_virtualizacion (main)
• $ git push -u origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 16 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 1.47 KiB | 1.47 MiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/julioxr/ayso-integrador-virtualizacion.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```


El repositorio ya se puede utilizar.

<https://github.com/julioxr/ayso-integrador-virtualizacion>

The screenshot shows a GitHub repository page for 'ayso-integrador-virtualizacion' by user 'julioxr'. The repository is public and has 0 stars, 0 forks, and 0 watchers. It has 1 branch (main) and 0 tags. The commit history shows two commits: 'Update README.md' (e639ad3, 2 days ago) and 'Primer commit' (script_notas.py, 2 days ago). The README file is selected, showing the title 'Trabajo Integrador Final – Virtualización'. The content includes the course 'Materia: Arquitectura y Sistemas Operativos', the topic 'Tema: Virtualización con VirtualBox y desarrollo en Python', the authors 'Integrantes: Cristian Emmanuel Rivero Corradi - Julio Cesar Roja', and the date 'Fecha: 01/06/2025'. The 'Objetivo del Proyecto' section states the goal is to understand and apply virtualization using VirtualBox. The 'Contenidos Abordados' section lists topics like virtualization, VirtualBox installation, Linux Mint in VM, and Linux terminal use. The right sidebar shows 'About' (AySO - TUP - Virtualización con VirtualBox y desarrollo en Python), 'Releases' (no releases published), 'Packages' (no packages published), 'Languages' (Python 100.0%), and 'Suggested workflows' (Pylint and Python application).

ayso-integrador-virtualizacion (Public)

main 1 Branch 0 Tags

Go to file Add file Code

Update README.md e639ad3 · 2 days ago 2 Commits

README.md Update README.md 2 days ago

script_notas.py Primer commit 2 days ago

README

Trabajo Integrador Final – Virtualización

Materia: Arquitectura y Sistemas Operativos
Tema: Virtualización con VirtualBox y desarrollo en Python
Integrantes: Cristian Emmanuel Rivero Corradi - Julio Cesar Roja **Fecha:** 01/06/2025

Objetivo del Proyecto

El objetivo de este trabajo es comprender y aplicar el concepto de virtualización mediante el uso de VirtualBox. Se busca instalar y configurar una máquina virtual con un sistema operativo Linux, y dentro de ella ejecutar un programa en Python desarrollado por el equipo.

Contenidos Abordados

- Virtualización y entornos aislados (sandbox)
- Instalación y configuración de VirtualBox
- Instalación de Linux Mint en VM
- Uso de la terminal de Linux

About

AySO - TUP - Virtualización con VirtualBox y desarrollo en Python

Readme

Activity

0 stars

0 watching

0 forks

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Languages

Python 100.0%

Suggested workflows

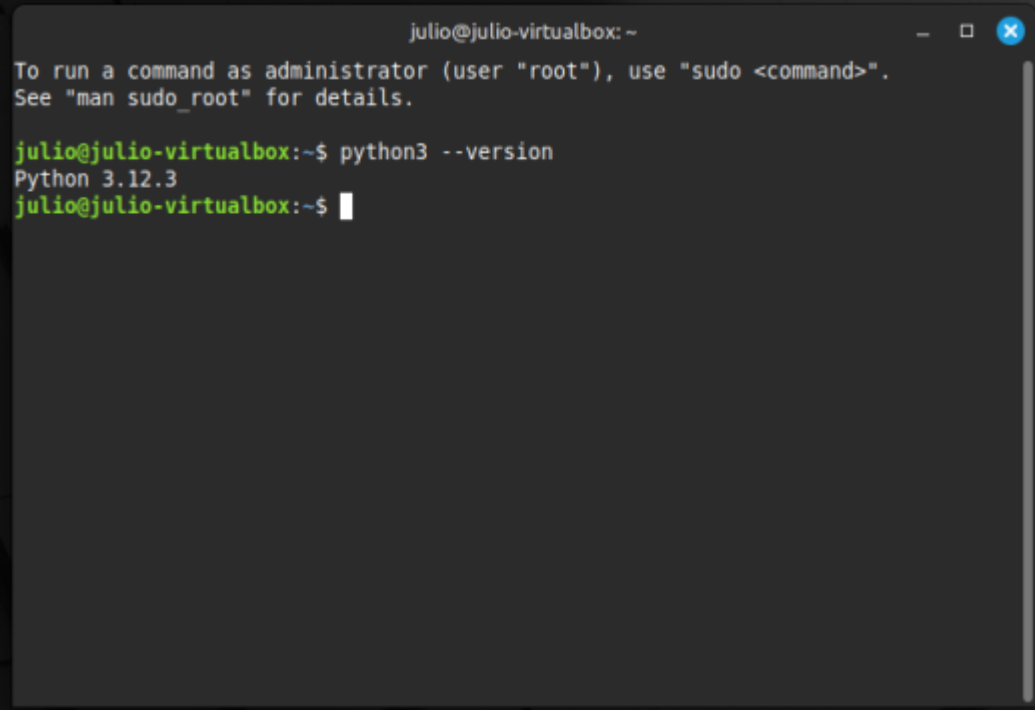
Based on your tech stack

Pylint [Configure](#)

Lint a Python application with pylint.

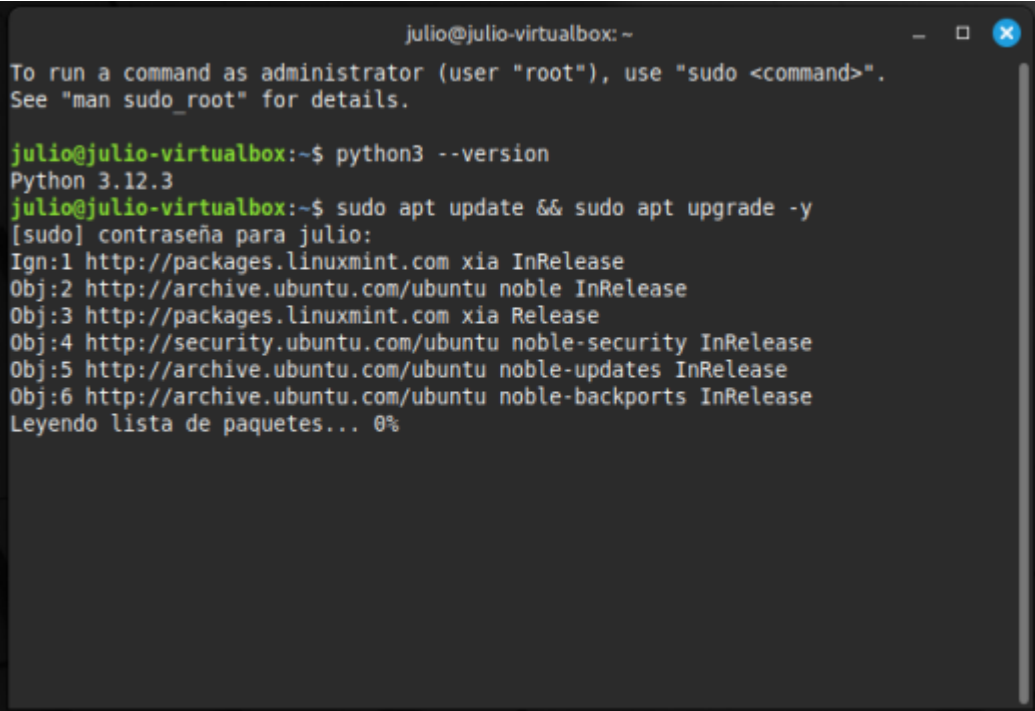
Python application [Configure](#)

En la máquina virtual abrimos la terminal y con el comando ***python3 --version*** chequeamos que tenemos instalado Python

A terminal window titled 'julio@julio-virtualbox: ~'. It contains the text: 'To run a command as administrator (user "root"), use "sudo <command>". See "man sudo_root" for details.' followed by the command 'python3 --version' and its output 'Python 3.12.3'. The prompt 'julio@julio-virtualbox:~\$' is followed by a cursor.

```
julio@julio-virtualbox: ~  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
julio@julio-virtualbox:~$ python3 --version  
Python 3.12.3  
julio@julio-virtualbox:~$
```

Luego ejecutamos los comandos ***sudo apt update*** && ***sudo apt upgrade -y*** . El primero actualiza la lista de paquetes disponibles y el segundo instala las últimas versiones de los paquetes, haciendo todo automáticamente gracias al parámetro -y.

A terminal window titled 'julio@julio-virtualbox: ~'. It contains the text: 'To run a command as administrator (user "root"), use "sudo <command>". See "man sudo_root" for details.' followed by the command 'python3 --version' and its output 'Python 3.12.3'. Then, the command 'sudo apt update && sudo apt upgrade -y' is entered. The output shows the password prompt '[sudo] contraseña para julio:', followed by a list of package sources and their release status, and finally 'Leyendo lista de paquetes... 0%'.

```
julio@julio-virtualbox: ~  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
julio@julio-virtualbox:~$ python3 --version  
Python 3.12.3  
julio@julio-virtualbox:~$ sudo apt update && sudo apt upgrade -y  
[sudo] contraseña para julio:  
Ign:1 http://packages.linuxmint.com xia InRelease  
Obj:2 http://archive.ubuntu.com/ubuntu noble InRelease  
Obj:3 http://packages.linuxmint.com xia Release  
Obj:4 http://security.ubuntu.com/ubuntu noble-security InRelease  
Obj:5 http://archive.ubuntu.com/ubuntu noble-updates InRelease  
Obj:6 http://archive.ubuntu.com/ubuntu noble-backports InRelease  
Leyendo lista de paquetes... 0%
```

El siguiente paso es instalar git en la máquina, con el comando ***sudo apt install git*** instalaremos git desde los repositorios oficiales

```
julio@julio-virtualbox: ~  
julio@julio-virtualbox:~$ sudo apt install git  
[sudo] contraseña para julio:  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias... Hecho  
Leyendo la información de estado... Hecho  
Se instalarán los siguientes paquetes adicionales:  
  git-man liberror-perl  
Paquetes sugeridos:  
  git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb  
  git-cvs git-mediawiki git-svn  
Se instalarán los siguientes paquetes NUEVOS:  
  git git-man liberror-perl  
0 actualizados, 3 nuevos se instalarán, 0 para eliminar y 1 no actualizados.  
Se necesita descargar 4.804 kB de archivos.  
Se utilizarán 24,5 MB de espacio de disco adicional después de esta operación.  
¿Desea continuar? [S/n] s  
Des:1 http://archive.ubuntu.com/ubuntu noble/main amd64 liberror-perl all 0.1702  
9-2 [25,6 kB]  
Des:2 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 git-man all 1:2.  
43.0-lubuntu7.2 [1.100 kB]  
Des:3 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 git amd64 1:2.43  
.0-lubuntu7.2 [3.679 kB]  
37% [3 git 293 kB/3.679 kB 8%]
```

Verificamos que tenemos instalado correctamente git con el comando ***git --version***

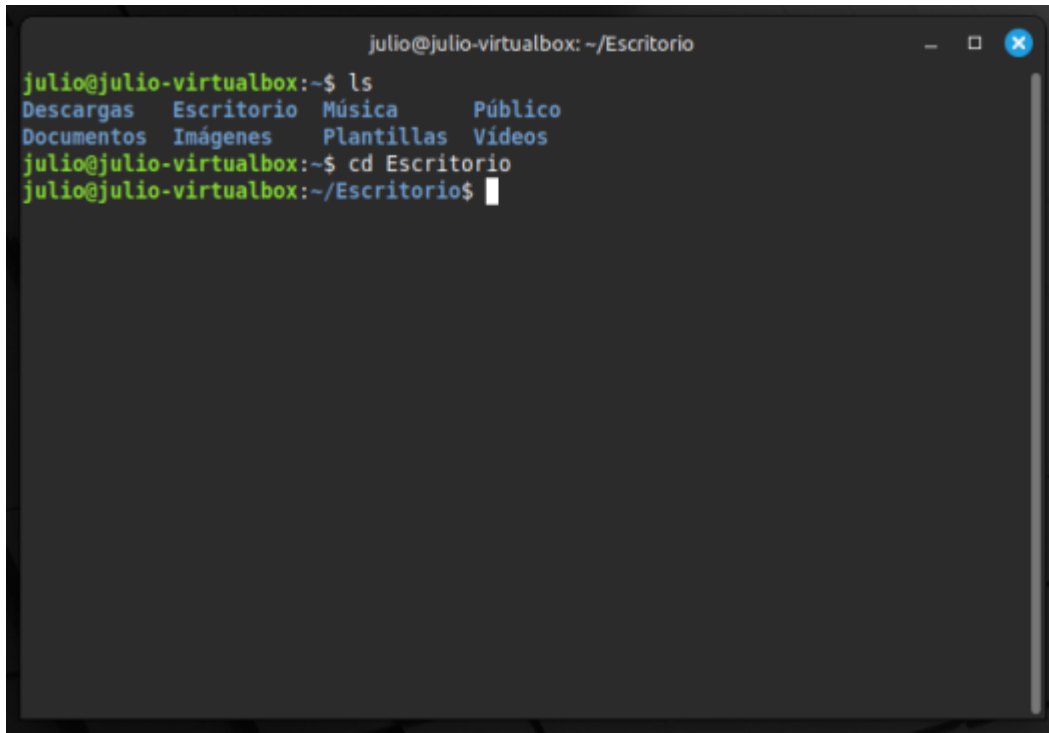
```
julio@julio-virtualbox: ~  
9-2 [25,6 kB]  
Des:2 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 git-man all 1:2.  
43.0-lubuntu7.2 [1.100 kB]  
Des:3 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 git amd64 1:2.43  
.0-lubuntu7.2 [3.679 kB]  
Descargados 4.804 kB en 18s (268 kB/s)  
Seleccionando el paquete liberror-perl previamente no seleccionado.  
(Leyendo la base de datos ... 496841 ficheros o directorios instalados actualmen  
te.)  
Preparando para desempaquetar .../liberror-perl_0.17029-2_all.deb ...  
Desempaquetando liberror-perl (0.17029-2) ...  
Seleccionando el paquete git-man previamente no seleccionado.  
Preparando para desempaquetar .../git-man_1%3a2.43.0-lubuntu7.2_all.deb ...  
Desempaquetando git-man (1:2.43.0-lubuntu7.2) ...  
Seleccionando el paquete git previamente no seleccionado.  
Preparando para desempaquetar .../git_1%3a2.43.0-lubuntu7.2_amd64.deb ...  
Desempaquetando git (1:2.43.0-lubuntu7.2) ...  
Configurando liberror-perl (0.17029-2) ...  
Configurando git-man (1:2.43.0-lubuntu7.2) ...  
Configurando git (1:2.43.0-lubuntu7.2) ...  
Procesando disparadores para man-db (2.12.0-4build2) ...  
julio@julio-virtualbox:~$ git --version  
git version 2.43.0  
julio@julio-virtualbox:~$
```

El siguiente paso es configurar git globalmente en la máquina para ello usaremos los comandos ***git config --global user.name*** y ***git config --global user.email***, de esta manera el nombre y el email se asociarán a los commits realizados desde este equipo.

A terminal window titled 'julio@julio-virtualbox: ~' with standard window controls. It shows two commands being executed: 'git config --global user.name "Julio Roja"' and 'git config --global user.email "julioroja987@gmail.com"'. The prompt 'julio@julio-virtualbox:~\$' is visible before each command. A cursor is at the end of the second command line.

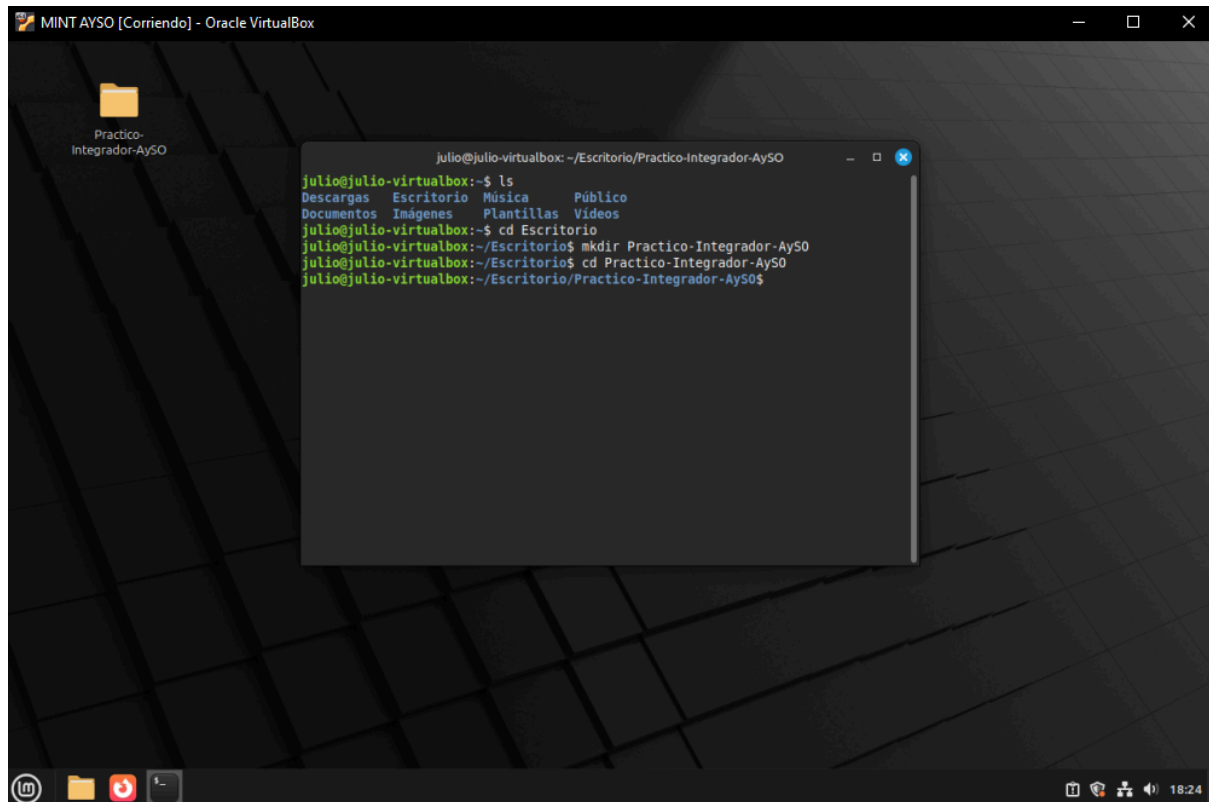
```
julio@julio-virtualbox: ~
julio@julio-virtualbox:~$ git config --global user.name "Julio Roja"
julio@julio-virtualbox:~$ git config --global user.email "julioroja987@gmail.com"
```

Ejecutamos el comando **ls** para ver el contenido del directorio en el que nos encontramos actualmente, y vamos a posicionarnos en el escritorio con el comando **cd Escritorio**

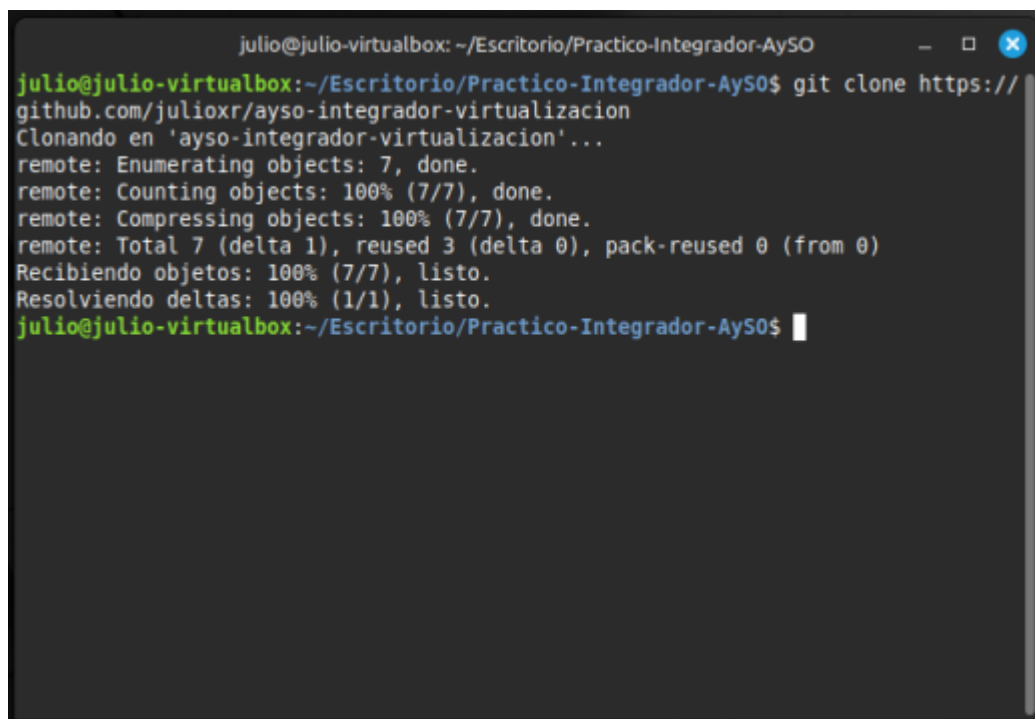
A terminal window titled 'julio@julio-virtualbox: ~/Escritorio' with standard window controls. The terminal shows the execution of 'ls' and 'cd Escritorio' commands. The output of 'ls' lists standard Linux desktop directories: Descargas, Escritorio, Música, Público, Documentos, Imágenes, Plantillas, and Videos. The prompt changes from '~\$' to '~/Escritorio\$' after the 'cd' command.

```
julio@julio-virtualbox: ~/Escritorio
julio@julio-virtualbox:~$ ls
Descargas  Escritorio  Música     Público
Documentos Imágenes    Plantillas Videos
julio@julio-virtualbox:~$ cd Escritorio
julio@julio-virtualbox:~/Escritorio$
```

Luego con el comando **mkdir** creamos la carpeta Practico-Integrador-AySO y ya la veremos en el escritorio creada. Luego con el comando **cd**, accedemos a la carpeta



Una vez dentro de la carpeta vamos a clonar el repositorio de github con el comando **git clone** <https://github.com/julioxr/ayso-integrador-virtualizacion>



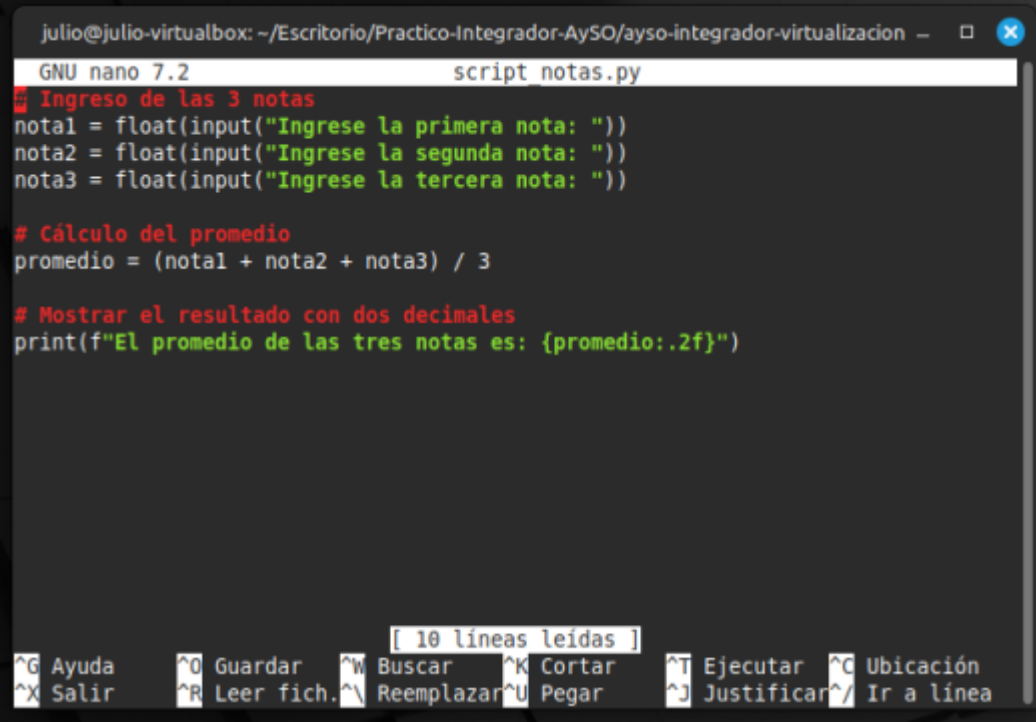
Accedemos a la carpeta que se nos creó del repositorio con el comando **cd** y luego listamos nuevamente con **ls** los archivos.

```
julio@julio-virtualbox: ~/Escritorio/Practico-Integrador-AySO/ayso-integrador-virtualizacion — □ ×
julio@julio-virtualbox:~/Escritorio/Practico-Integrador-AySO$ cd ays0-integrador-virtualizacion
julio@julio-virtualbox:~/Escritorio/Practico-Integrador-AySO/ayso-integrador-virtualizacion$ ls
README.md  script_notas.py
julio@julio-virtualbox:~/Escritorio/Practico-Integrador-AySO/ayso-integrador-virtualizacion$
```

Vemos que tenemos el archivo **script_notas.py**. Para editarlo, lo abrimos con el editor de texto nano usando el siguiente comando **nano script_notas.py**

```
julio@julio-virtualbox: ~/Escritorio/Practico-Integrador-AySO/ayso-integrador-virtualizacion — □ ×
julio@julio-virtualbox:~/Escritorio/Practico-Integrador-AySO/ayso-integrador-virtualizacion$ nano script_notas.py
```

Allí podremos observar nuestro código de Python, el cual ejecuta un programa que sirve para calcular el promedio de 3 notas ingresadas por el usuario. El programa solicita al usuario que introduzca tres valores numéricos, realiza la suma de estos valores y luego calcula el promedio, mostrando el resultado en pantalla.



The image shows a terminal window titled "julio@julio-virtualbox: ~/Escritorio/Practico-Integrador-AySO/ayso-integrador-virtualizacion". Inside the terminal, the GNU nano 7.2 editor is open, editing a file named "script notas.py". The script contains the following Python code:

```
# Ingreso de las 3 notas
nota1 = float(input("Ingrese la primera nota: "))
nota2 = float(input("Ingrese la segunda nota: "))
nota3 = float(input("Ingrese la tercera nota: "))

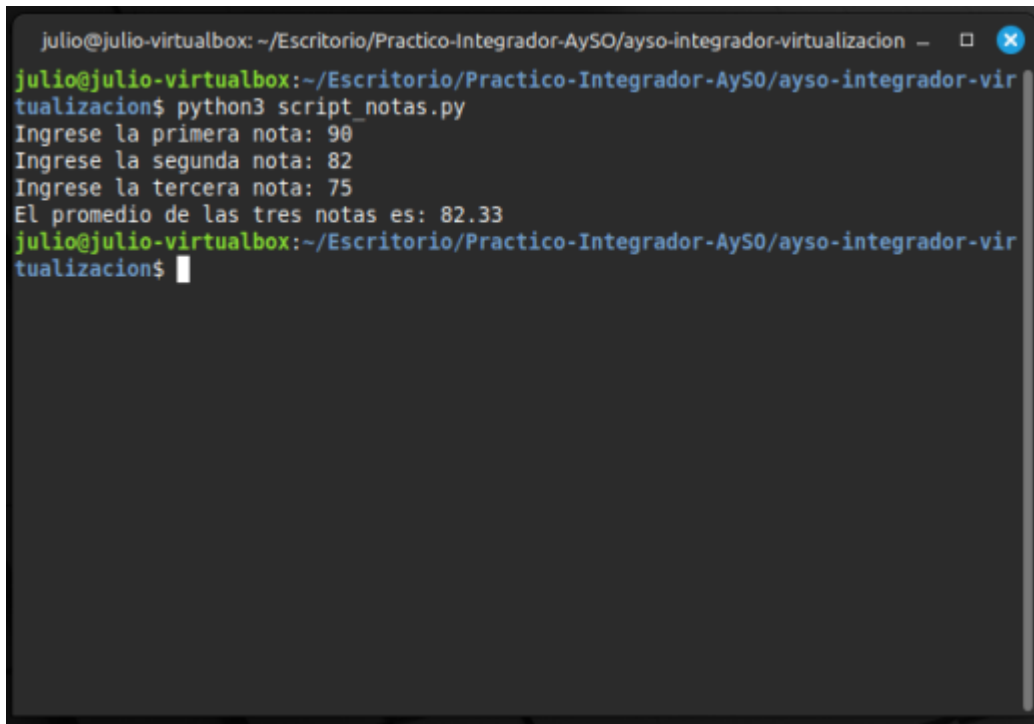
# Cálculo del promedio
promedio = (nota1 + nota2 + nota3) / 3

# Mostrar el resultado con dos decimales
print(f"El promedio de las tres notas es: {promedio:.2f}")
```

At the bottom of the terminal, there is a status bar showing "[10 líneas leídas]" and a table of keyboard shortcuts:

[^] G Ayuda	[^] O Guardar	[^] W Buscar	[^] K Cortar	[^] T Ejecutar	[^] C Ubicación
[^] X Salir	[^] R Leer fich.	[^] \ Reemplazar	[^] U Pegar	[^] J Justificar	[^] / Ir a línea

Salimos con **control+x** y dentro de la carpeta del proyecto vamos a ejecutar el programa con el comando **python3 script_notas.py**

A terminal window titled 'julio@julio-virtualbox: ~/Escritorio/Practico-Integrador-AySO/ayso-integrador-virtualizacion'. The prompt is 'julio@julio-virtualbox:~/Escritorio/Practico-Integrador-AySO/ayso-integrador-virtualizacion\$'. The user enters 'python3 script_notas.py'. The script prompts for three notes: 'Ingrese la primera nota: 90', 'Ingrese la segunda nota: 82', and 'Ingrese la tercera nota: 75'. It then calculates and displays 'El promedio de las tres notas es: 82.33'. The prompt returns to 'julio@julio-virtualbox:~/Escritorio/Practico-Integrador-AySO/ayso-integrador-virtualizacion\$' with a cursor.

```
julio@julio-virtualbox: ~/Escritorio/Practico-Integrador-AySO/ayso-integrador-virtualizacion
julio@julio-virtualbox:~/Escritorio/Practico-Integrador-AySO/ayso-integrador-virtualizacion$ python3 script_notas.py
Ingrese la primera nota: 90
Ingrese la segunda nota: 82
Ingrese la tercera nota: 75
El promedio de las tres notas es: 82.33
julio@julio-virtualbox:~/Escritorio/Practico-Integrador-AySO/ayso-integrador-virtualizacion$
```

4) METODOLOGÍA UTILIZADA

El desarrollo del trabajo se llevó a cabo siguiendo los siguientes pasos y criterios metodológicos.

Investigación previa

Se realizó una revisión de documentación oficial y tutoriales en línea para comprender los conceptos de virtualización, el uso de hipervisores y la instalación de sistemas operativos en entornos virtualizados.

Herramientas y recursos utilizados

- VirtualBox: para la creación y administración de la máquina virtual
- Linux Mint: como sistema operativo
- Python 3: Para el desarrollo y ejecución del script
- Nano: como editor de código dentro de la máquina virtual
- Capturador de pantalla: para documentar visualmente cada paso del proceso
- Git y Github: Para gestionar los archivos del proyecto.

Diseño y desarrollo

Se definió la estructura del trabajo y se elaboró una lista de pasos a seguir, que incluyó la instalación de VirtualBox, configuración de la VM, instalación de Linux, desarrollo y prueba del script en Python. Cada etapa fue documentada con capturas de pantalla y breves descripciones.

Trabajo colaborativo

El trabajo se realizó de manera colaborativa dividiendo las tareas como la instalación del software, la redacción del informe, el desarrollo del código y la obtención de capturas de fotos y video.

5) RESULTADOS OBTENIDOS

Se logró instalar y configurar correctamente una máquina virtual en Oracle VirtualBox, utilizando Linux Mint como sistema operativo. Todo el proceso fue documentado con capturas de pantalla y video, desde la creación de la VM hasta la ejecución del programa en Python.

El script funcionó según lo esperado: permitió ingresar tres notas, calcular el promedio y mostrar el resultado en pantalla sin errores. Las pruebas realizadas confirmaron el correcto funcionamiento tanto del entorno virtualizado como del programa desarrollado.

La principal dificultad fue la lentitud de la máquina virtual debido a los recursos limitados del equipo anfitrión, lo que generó demoras al trabajar con el entorno gráfico. Sin embargo, esto no impidió completar el trabajo ni cumplir con los objetivos planteados.

6) CONCLUSIONES

La realización de este trabajo integrador permitió aplicar de forma práctica los conceptos de virtualización. La utilización de Oracle VirtualBox facilitó la creación y configuración de un entorno virtual, demostrando su accesibilidad y facilidad de uso para usuarios sin experiencia previa en estas tecnologías.

El trabajo en equipo resultó fundamental para organizar las tareas y resolver dudas que surgieron durante el proceso, favoreciendo el aprendizaje colaborativo.

Además, se constató la importancia de contar con un hardware adecuado, ya que el rendimiento de la máquina virtual depende directamente de los recursos disponibles en el equipo anfitrión. Esta experiencia puso de manifiesto la necesidad de optimizar la asignación de recursos.

En resumen, el proyecto permitió consolidar conocimientos clave sobre virtualización, la administración básica de sistemas y el desarrollo de scripts sencillos en Python, aportando herramientas útiles para futuros desafíos académicos y profesionales.

7) BIBLIOGRAFÍA

- Oracle. (s.f.). *VirtualBox User Manual*. <https://www.virtualbox.org/manual/>
- Linux Mint. (s.f.). *Documentación oficial de Linux Mint*.
<https://linuxmint.com/documentation.php>

8) ANEXOS