



Instituto Politécnico Nacional
Escuela Superior de Cómputo



González Bocio Erik Alexander

2020630163

4CV11

Pineda Guerrero Carlos

Desarrollo de Sistemas Distribuidos

Tarea 3: Chat multicast

Desarrollo

Primero vamos a abordar la conversación que se tuvo con el ChatBot ChatGPT, la cual se muestra a continuación:

Erik González

dame un programa en java que implemente un chat utilizando comunicación multicast mediante datagramas, se debera pasar como parametro al programa el nombre del usuario que va a escribir en el chat. El programa creará un thread el cual recibirá los mensajes del resto de los nodos. Cada mensaje recibido sera impreso hasta el que envía el mismo nodo. En el main, dentro de un ciclo infinito, se desplegará "Escribe tu mensaje: ", entonces se leerá del teclado el mensaje, se deberá enviar el mensaje a los nodos que pertenecen al grupo identificado por la IP 239.0.0.0 a través del puerto 50000. El mensaje a enviar en el datagrama debera tener la siguiente forma "usuario--->mensaje", donde usuario es el usuario que pasamos como parametro y mensaje es lo que se leyo del teclado. Debido a que el protocolo UDP no garantiza el orden en que se reciben los mensajes, no se deberá enviar la longitud del mensaje y luego el mensaje; se deberá utilizar una longitud de mensaje fija. Al desplegar el mensaje recibido se deberán eliminar los espacios adicionales a la derecha (si los hay), además que tiene que respetar los caracteres especiales y acentos, por lo que usa alguna codificacion para que se puedan ver.

ChatGPT

A continuación, te presento un ejemplo de programa en Java que implementa un chat utilizando comunicación multicast mediante datagramas. Este programa recibe como parámetro el nombre del usuario que va a escribir en el chat y crea un thread que se encarga de recibir los mensajes del resto de los nodos. Cada mensaje recibido es impreso hasta que es enviado por el mismo nodo. El programa utiliza la dirección IP 239.0.0.0 y el puerto 50000 para enviar los mensajes a los nodos que pertenecen al grupo multicast.

```
import java.io.IOException;
```

```
import java.net.*;
```

```
public class MulticastChat {
```

```
    public static void main(String[] args) throws IOException {
```

```
        if (args.length < 1) {
```

```
System.out.println("Usage: java MulticastChat <username>");  
return;  
}
```

```
String username = args[0];  
MulticastSocket socket = new MulticastSocket(50000);  
InetAddress group = InetAddress.getByName("239.0.0.0");  
socket.joinGroup(group);
```

```
Thread receiverThread = new Thread(() -> {  
    byte[] buffer = new byte[256];  
    while (true) {  
        DatagramPacket packet = new DatagramPacket(buffer, buffer.length);  
        try {  
            socket.receive(packet);  
            String message = new String(packet.getData(), 0, packet.getLength(), "UTF-8");  
            String sender = message.split("---->")[0];  
            if (!sender.equals(username)) {  
                System.out.println(message);  
            }  
        } catch (IOException e) {  
            e.printStackTrace();  
        }  
    }  
});  
receiverThread.start();
```

```

BufferedReader input = new BufferedReader(new InputStreamReader(System.in));
while (true) {
    System.out.print("Escribe tu mensaje: ");
    String message = input.readLine();
    message = message.trim().replaceAll("\\s+", " ");
    String formattedMessage = username + "--->" + message;
    byte[] data = formattedMessage.getBytes("UTF-8");
    DatagramPacket packet = new DatagramPacket(data, data.length, group, 50000);
    socket.send(packet);
}
}
}

```

Este programa utiliza la clase `MulticastSocket` para crear el socket multicast y unirse al grupo multicast identificado por la dirección IP 239.0.0.0 y el puerto 50000. Luego, crea un thread que se encarga de recibir los mensajes que llegan por el socket y los imprime en la consola. En el main, dentro de un ciclo infinito, se lee del teclado el mensaje que se va a enviar y se construye un datagrama que contiene el nombre del usuario y el mensaje separados por el texto `--->`. El datagrama se envía al grupo multicast utilizando el método `send` de la clase `MulticastSocket`.

Es importante destacar que para eliminar los espacios adicionales a la derecha se utiliza el método `trim` de la clase `String` y para reemplazar los espacios múltiples por un solo espacio se utiliza una expresión regular con el método `replaceAll` de la clase `String`. Además, se utiliza la codificación UTF-8 para asegurar que se puedan ver los caracteres especiales y acentos en los mensajes.

*Aquí me di cuenta del uso de `bufferedReader` y además no usaba la clase `worker` por lo que le pedí cambiar el código.

Erik González

no uses `bufferedReader`, además que para unirse al grupo quiero que uses una clase estática `Worker` que extienda a `Thread` y tenga un método `run`

ChatGPT

Entendido, a continuación te presento un ejemplo actualizado de programa en Java que implementa un chat utilizando comunicación multicast mediante datagramas, sin usar `BufferedReader` y con unirse al grupo mediante una clase estática `Worker` que extiende `Thread` y tiene un método `run`.

```
import java.io.IOException;
```

```
import java.net.*;
```

```
public class MulticastChat {
```

```
    static class Worker extends Thread {
```

```
        MulticastSocket socket;
```

```
        String username;
```

```
        public Worker(MulticastSocket socket, String username) {
```

```
            this.socket = socket;
```

```
            this.username = username;
```

```
        }
```

```
        @Override
```

```
        public void run() {
```

```
            byte[] buffer = new byte[256];
```

```
            while (true) {
```

```
                DatagramPacket packet = new DatagramPacket(buffer, buffer.length);
```

```
                try {
```

```
                    socket.receive(packet);
```

```
                    String message = new String(packet.getData(), 0, packet.getLength(), "UTF-8");
```

```
                    String sender = message.split("--->")[0];
```

```

        if (!sender.equals(username)) {
            System.out.println(message);
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
}
}
}
}

```

```

public static void main(String[] args) throws IOException {
    if (args.length < 1) {
        System.out.println("Usage: java MulticastChat <username>");
        return;
    }
}

```

```

String username = args[0];
MulticastSocket socket = new MulticastSocket(50000);
InetAddress group = InetAddress.getByName("239.0.0.0");
socket.joinGroup(group);

```

```

Worker worker = new Worker(socket, username);
worker.start();

```

```

while (true) {
    byte[] buffer = new byte[256];
    System.out.print("Escribe tu mensaje: ");
}

```

```

String message = System.console().readLine();

message = message.trim().replaceAll("\\s+", " ");

String formattedMessage = username + "--->" + message;

byte[] data = formattedMessage.getBytes("UTF-8");

DatagramPacket packet = new DatagramPacket(data, data.length, group, 50000);

socket.send(packet);

    }

}

}

```

En este programa, se utiliza la clase estática Worker que extiende Thread y tiene un método run que se encarga de recibir los mensajes que llegan por el socket y los imprime en la consola. En el main, se crea una instancia de Worker y se la ejecuta con el método start. Luego, se entra en un ciclo infinito en el cual se lee el mensaje del usuario utilizando System.console().readLine() y se construye un datagrama que contiene el nombre del usuario y el mensaje separados por el texto --->. El datagrama se envía al grupo multicast utilizando el método send de la clase MulticastSocket.

Es importante destacar que para eliminar los espacios adicionales a la derecha se utiliza el método trim de la clase String y para reemplazar los espacios múltiples por un solo espacio se utiliza una expresión regular con el método replaceAll de la clase String. Además, se utiliza la codificación UTF-8 para asegurar que se puedan ver los caracteres especiales y acentos en los mensajes.

*En esta parte me di cuenta que al compilarlo no me mostraba el mensaje en el nodo donde se mandó, por lo que le pedí cambiara eso, además que se amontonaban los mensajes con el mensaje "Escribe tu mensaje: "

Erik González

ok ahora solo necesito que el mensaje se muestre en todos los nodos, hasta en el que escribio el mensaje, además luego de esto pida de nuevo un mensaje en todos los nodos, además despues el mensaje "Escribe tu mensaje: " dale un salto de linea por favor

ChatGPT

Entendido, a continuación te presento el programa actualizado en Java que muestra los mensajes en todos los nodos, incluyendo el que escribió el mensaje, pide un nuevo mensaje en todos los nodos después de mostrar el mensaje, agrega un salto de línea al mensaje "Escribe tu mensaje:" y además elimina el uso de la clase `System.console()` y en su lugar usa la clase `Scanner`.

```
import java.io.IOException;

import java.net.*;

import java.util.Scanner;

public class MulticastChat {

    static class Worker extends Thread {

        MulticastSocket socket;

        String username;

        public Worker(MulticastSocket socket, String username) {

            this.socket = socket;

            this.username = username;

        }

        @Override

        public void run() {

            byte[] buffer = new byte[256];

            while (true) {

                DatagramPacket packet = new DatagramPacket(buffer, buffer.length);

                try {

                    socket.receive(packet);

                    String message = new String(packet.getData(), 0, packet.getLength(), "UTF-8");

                    System.out.println(message);
```



```

        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
}

```

```

public static void main(String[] args) throws IOException {
    if (args.length < 1) {
        System.out.println("Usage: java MulticastChat <username>");
        return;
    }

```

```

    String username = args[0];
    MulticastSocket socket = new MulticastSocket(50000);
    InetAddress group = InetAddress.getByName("239.0.0.0");
    socket.joinGroup(group);

```

```

    Worker worker = new Worker(socket, username);
    worker.start();

```

```

    Scanner scanner = new Scanner(System.in);
    while (true) {
        System.out.println("Escribe tu mensaje: ");
        String message = scanner.nextLine();
        message = message.trim().replaceAll("\\s+", " ");
        String formattedMessage = username + "--->" + message;
    }

```

```

        byte[] data = formattedMessage.getBytes("UTF-8");

        DatagramPacket packet = new DatagramPacket(data, data.length, group, 50000);

        socket.send(packet);
    }
}

```

En este programa, se eliminó la condición que comparaba el nombre de usuario del mensaje recibido con el nombre de usuario actual para decidir si mostrar el mensaje en la consola. Ahora todos los mensajes recibidos son mostrados en la consola, incluyendo el que fue enviado por el usuario actual.

Además, se agregó un salto de línea después del mensaje "Escribe tu mensaje:", para mejorar la legibilidad del programa.

Para leer la entrada del usuario, se utiliza la clase Scanner y su método `nextLine()`, en lugar de la clase `System.console()`.

*Aquí cometí el error de probar el código en una distribución UNIX por lo que el programa se ejecutaba muy bien al usar UTF-8, debido a que la máquina virtual con la que íbamos a trabajar era Windows probe el código en Windows 10 y note que no funcionaba, por lo que le pedí cambiara eso.

Erik González

con UTF-8 no me permite ver los caracteres especiales y acentos en los mensajes el cmd de Windows

ChatGPT

Entiendo. El problema se debe a que el cmd de Windows no tiene soporte para la codificación UTF-8 por defecto. Una posible solución es cambiar la codificación a "ISO-8859-1", que es una codificación que incluye los caracteres acentuados utilizados en español.

Para hacer esto, puedes reemplazar todas las instancias de "UTF-8" en el código por "ISO-8859-1". El código actualizado quedaría así:

```
import java.io.IOException;
```

```
import java.net.*;
```

```
import java.util.Scanner;
```

```
public class MulticastChat {
```

```
    static class Worker extends Thread {
```

```
        MulticastSocket socket;
```

```
        String username;
```

```
        public Worker(MulticastSocket socket, String username) {
```

```
            this.socket = socket;
```

```
            this.username = username;
```

```
        }
```

```
        @Override
```

```
        public void run() {
```

```
            byte[] buffer = new byte[256];
```

```
            while (true) {
```

```
                DatagramPacket packet = new DatagramPacket(buffer, buffer.length);
```

```
                try {
```

```
                    socket.receive(packet);
```

```
                    String message = new String(packet.getData(), 0, packet.getLength(), "ISO-8859-1");
```

```
                    System.out.println(message);
```

```
                } catch (IOException e) {
```

```

        e.printStackTrace();
    }
}
}
}

```

```

public static void main(String[] args) throws IOException {
    if (args.length < 1) {
        System.out.println("Usage: java MulticastChat <username>");
        return;
    }

```

```

    String username = args[0];
    MulticastSocket socket = new MulticastSocket(50000);
    InetAddress group = InetAddress.getByName("239.0.0.0");
    socket.joinGroup(group);

```

```

    Worker worker = new Worker(socket, username);
    worker.start();

```

```

    Scanner scanner = new Scanner(System.in);
    while (true) {
        System.out.println("Escribe tu mensaje:");
        String message = scanner.nextLine();
        message = message.trim().replaceAll("\\s+", " ");
        String formattedMessage = username + "--->" + message;
        byte[] data = formattedMessage.getBytes("ISO-8859-1");
    }
}

```

```
DatagramPacket packet = new DatagramPacket(data, data.length, group, 50000);  
socket.send(packet);  
}  
}  
}
```

De esta manera, los caracteres especiales y acentos deberían ser mostrados correctamente en el cmd de Windows.

*Al probar lo que me dijo tampoco funciono, por lo que probe preguntándole de nuevo.

Erik González

sigue sin servir

ChatGPT

Si aún tienes problemas para ver los caracteres especiales y acentos, puede ser que el problema esté relacionado con la configuración regional de tu sistema operativo.

Puedes intentar cambiar la página de códigos de la consola de Windows ejecutando el siguiente comando en el cmd:

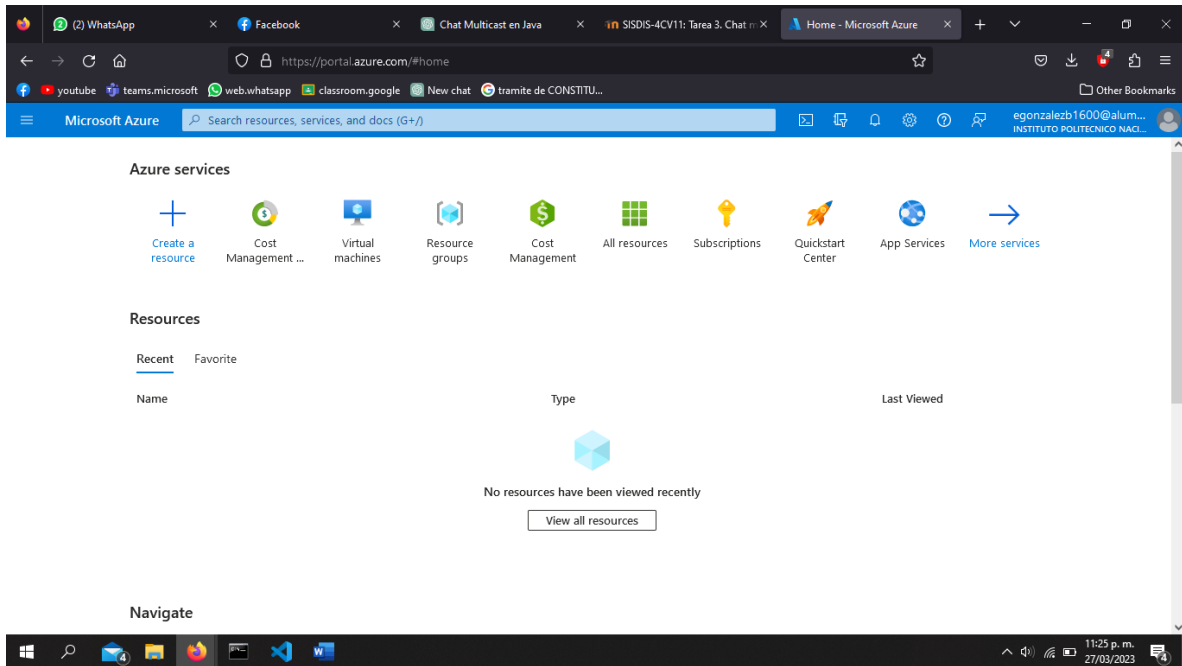
chcp 1252

Esto cambia la página de códigos a "Windows 1252", que es una codificación que incluye los caracteres acentuados utilizados en español. Luego puedes volver a ejecutar el programa.

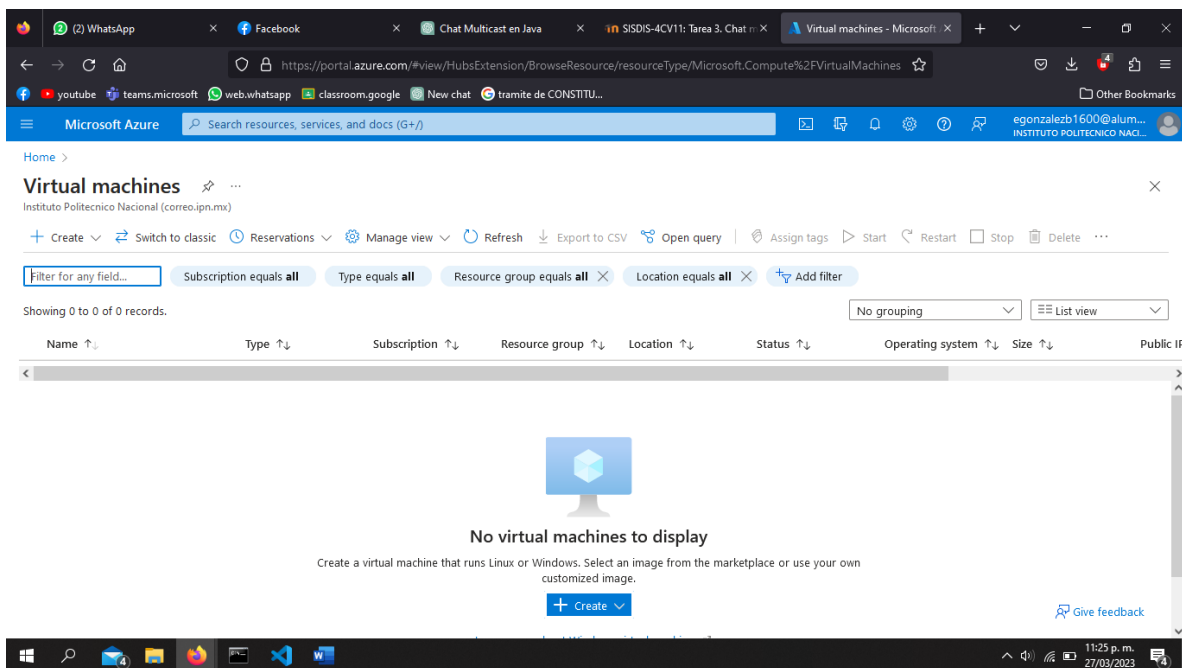
Si aún así no funciona, puede que el problema sea más complejo y necesites cambiar la configuración regional de tu sistema operativo.

*Finalmente funciono usando este comando en todas las terminales que ocupaba.

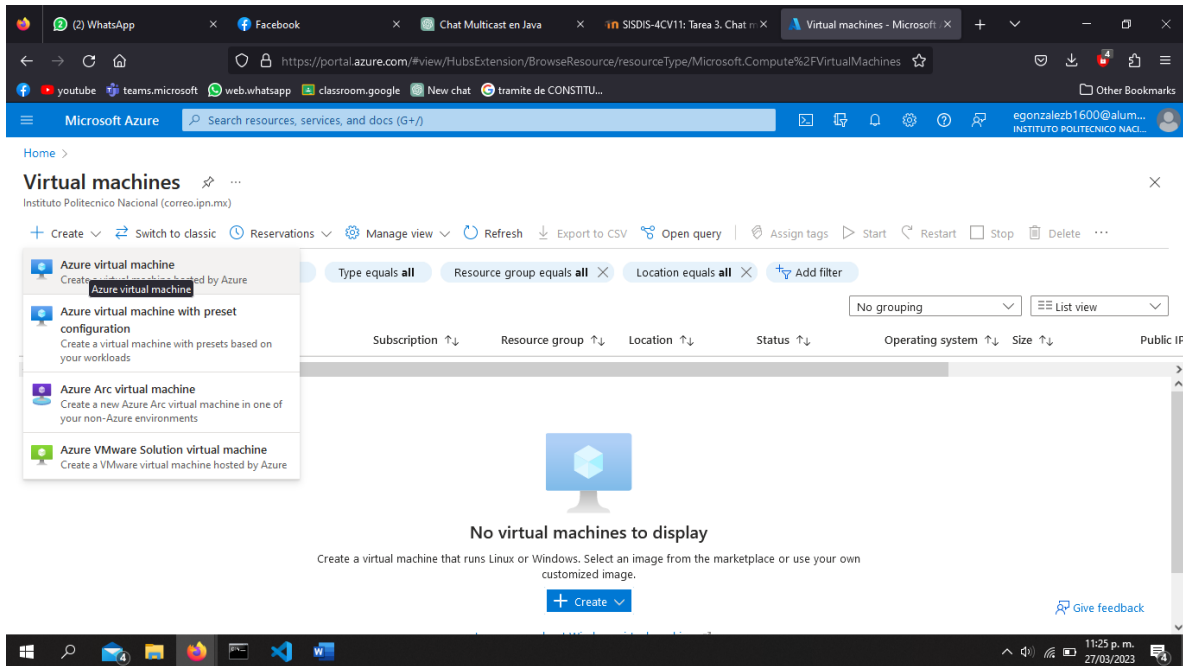
Ahora vamos con la creación de la máquina virtual. Por lo que vamos al enlace <https://portal.azure.com/#home>, si no tienes iniciada tu sesión, inicia sesión, si no es el caso, te aparecerá una pantalla como la siguiente, por lo que damos clic en Virtual Machines



Nos mostrara una ventana como la siguiente, por lo que seleccionamos Create



Y del menú que se despliega seleccionamos Azure virtual machine



Entonces nos mostrara una ventana como la siguiente, aquí tendremos que colocar todos los recursos de la máquina, el nombre, la región, la imagen, la CPU, etc. En este caso colocamos los datos como se muestran en las siguientes ventanas.

Microsoft Azure portal interface showing the "Create a virtual machine" wizard. The browser tabs include WhatsApp, Facebook, Chat Multicast en Java, and the current page "Create a virtual machine - Microsoft Azure". The URL is <https://portal.azure.com/#create/Microsoft.VirtualMachine-ARM>.

Subscription *

Resource group * [Create new](#)

Instance details

Virtual machine name *

Region *

Availability options

Security type

Image * [See all images](#) [Configure VM generation](#)

☒ There is a generation 2 version of this image available which has higher feature compatibility. [Click here to swap to the generation 2 version](#)

[Review + create](#) [< Previous](#) [Next : Disks >](#) [Give feedback](#)

Microsoft Azure portal interface showing the "Create a virtual machine" wizard. The browser tabs include WhatsApp, Facebook, Chat Multicast en Java, and the current page "Create a virtual machine - Microsoft Azure". The URL is <https://portal.azure.com/#create/Microsoft.VirtualMachine>.

VM architecture ☐ Arm64 ☒ x64

☒ Arm64 is not supported with the selected image.

Run with Azure Spot discount ☐

Size * [See all sizes](#)

Administrator account

Username *

Password *

Confirm password *

Inbound port rules

[Review + create](#) [< Previous](#) [Next : Disks >](#) [Give feedback](#)

Home >

Create a virtual machine

Password *

Confirm password *

Inbound port rules

Select which virtual machine network ports are accessible from the public internet. You can specify more limited or granular network access on the Networking tab.

Public inbound ports * ☐ None ☒ Allow selected ports

Select inbound ports *

Warning: This will allow all IP addresses to access your virtual machine. This is only recommended for testing. Use the Advanced controls in the Networking tab to create rules to limit inbound traffic to known IP addresses.

[Review + create](#) [< Previous](#) [Next: Disks >](#) [Give feedback](#)

Luego de colocar todos estos datos le damos a Next: Disk >

Home >

Create a virtual machine

OS disk

OS disk type *
The selected VM size supports premium disks. We recommend Premium SSD for high IOPS workloads. Virtual machines with Premium SSD disks qualify for the 99.9% connectivity SLA.

Delete with VM * ☒

Key management *

Enable Ultra Disk compatibility * ☐
Ultra disk is supported in Availability Zone(s) 1,2,3 for the selected VM size Standard_B1ms.

Data disks for T3-2020630163

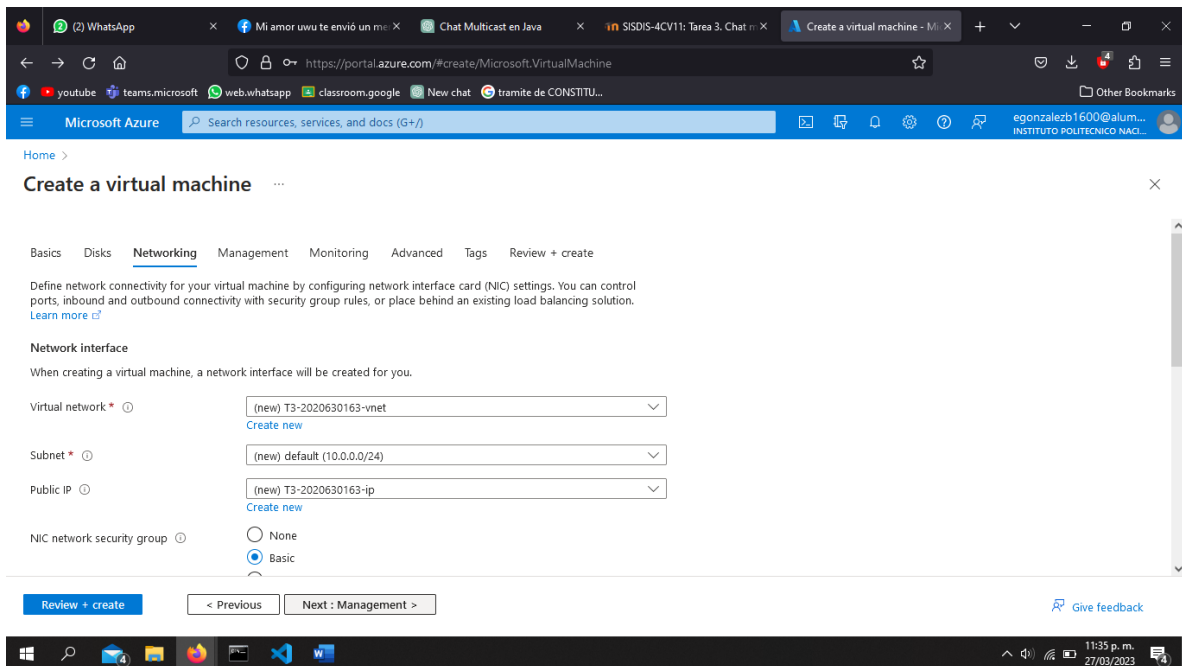
You can add and configure additional data disks for your virtual machine or attach existing disks. This VM also comes with a temporary disk.

LUN	Name	Size (GiB)	Disk type	Host cache	Delete with VM
-----	------	------------	-----------	------------	----------------

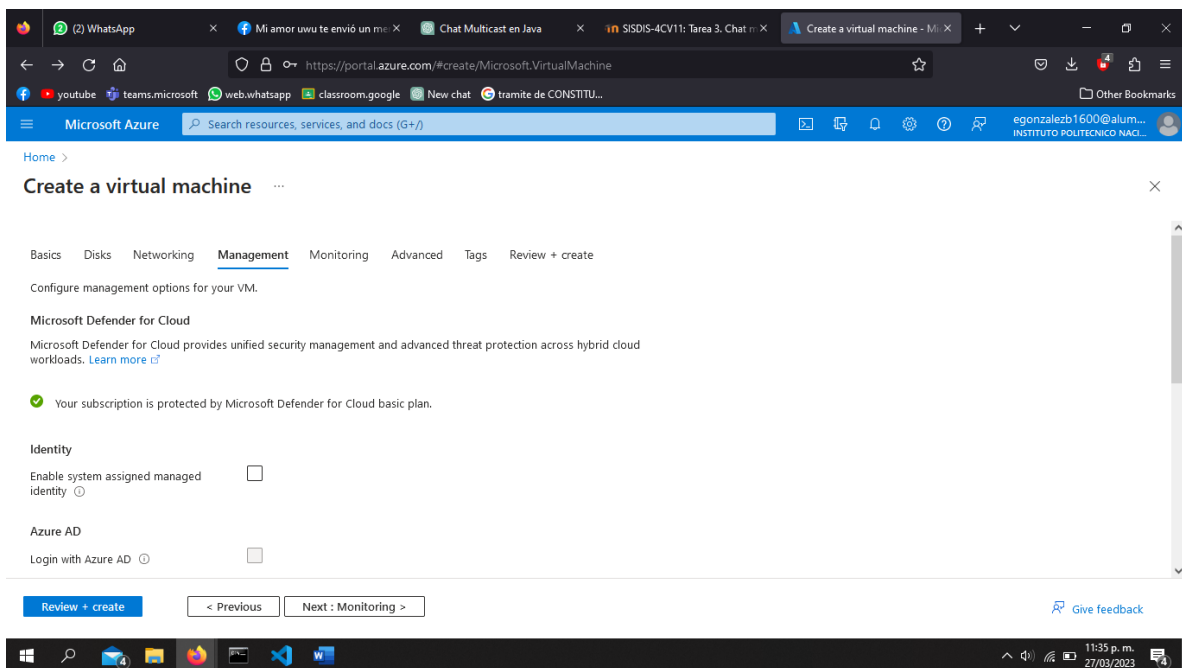
[Create and attach a new disk](#) [Attach an existing disk](#)

[Review + create](#) [< Previous](#) [Next: Networking >](#) [Give feedback](#)

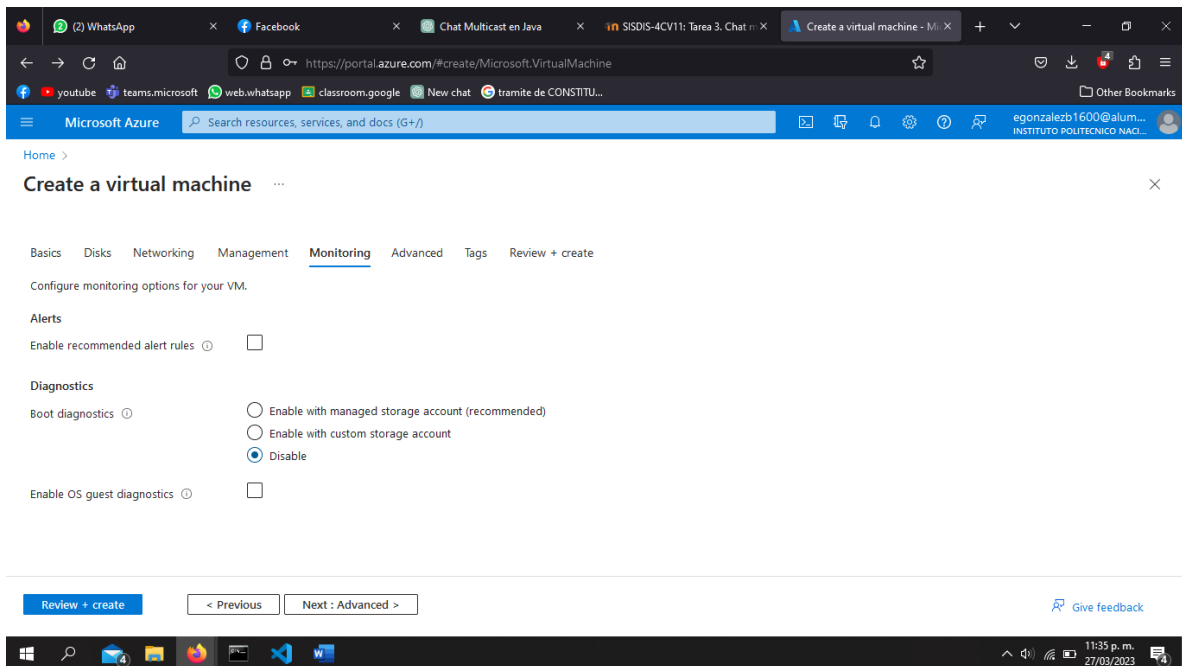
Aquí colocamos el disco como Standard HDD y le damos a Next: Networking >



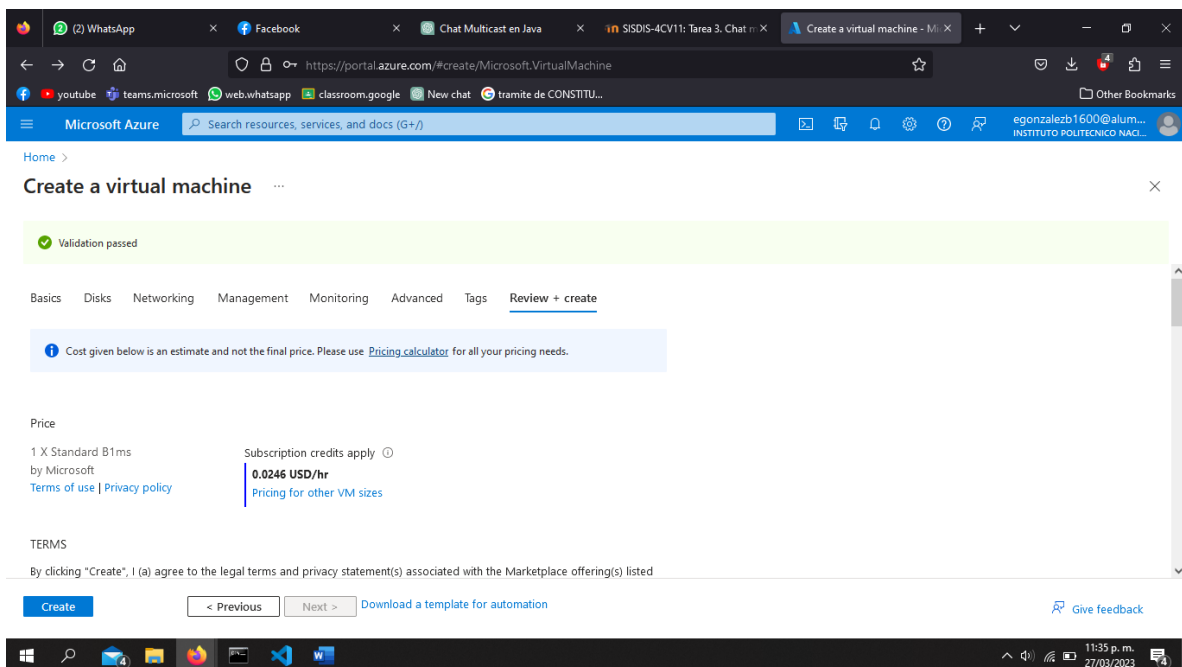
Mantenemos la configuración que se tiene por defecto y le damos a Next: Managment >



Del mismo modo mantenemos la configuración que se tiene por defecto y le damos a Next: Monitoring >



Aquí es importante que en la opción Boot diagnostics lo coloquemos como Disable, así le damos a Review+create y se obtiene la pantalla siguiente:



Aquí se nos muestran los datos de la máquina y el costo que se va a tener de esta por hora, si estamos de acuerdo le damos a Create y se muestra la pantalla siguiente:

The screenshot shows the Microsoft Azure portal interface. The browser address bar displays the URL: `https://portal.azure.com/#view/HubsExtension/DeploymentDetailsBlade/~/overview/id/%2Fsubscriptions%2Fba1c955-c0b1-455b-a32c-127344a78844`. The page title is "CreateVm-MicrosoftWindowsServer.WindowsServer-201-20230327233007 | Overview". The left sidebar contains navigation links: Overview, Inputs, Outputs, and Template. The main content area shows a deployment in progress with the title "Deployment is in progress". It includes the deployment name "CreateVm-MicrosoftWindowsServer.WindowsS...", subscription "Azure for Students", and resource group "T3-2020630163_group". The start time is "3/27/2023, 11:36:15 PM" and the correlation ID is "094b5b14-a9be-48bc-86d6-12". A table lists the deployment resources:

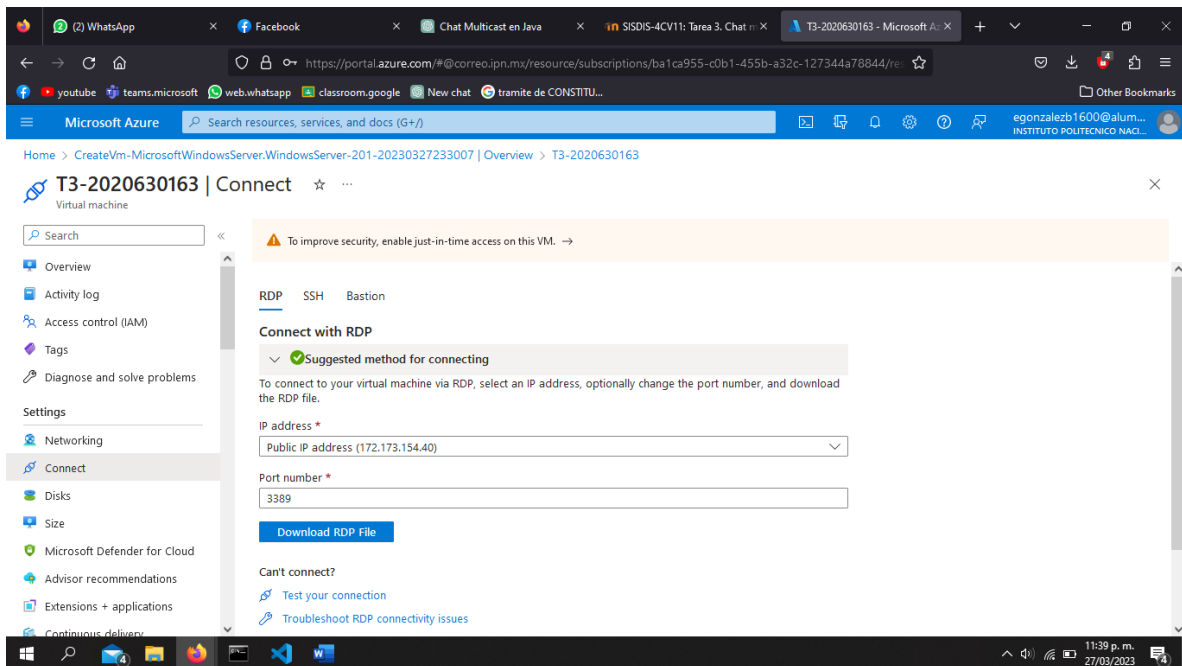
Resource	Type	Status	Operation details
T3-2020630163	Microsoft.Compute/virtua...	Created	Operation details
t3-2020630163783	Microsoft.Network/netwo...	Created	Operation details
T3-2020630163-vnet	Microsoft.Network/virtua...	OK	Operation details
T3-2020630163-nsg	Microsoft.Network/netwo...	OK	Operation details
T3-2020630163-ip	Microsoft.Network/public...	OK	Operation details

On the right side, there are recommendations for "Microsoft Defender for Cloud", "Free Microsoft tutorials", and "Work with an expert". The bottom status bar shows the time "11:36 p. m." and date "27/03/2023".

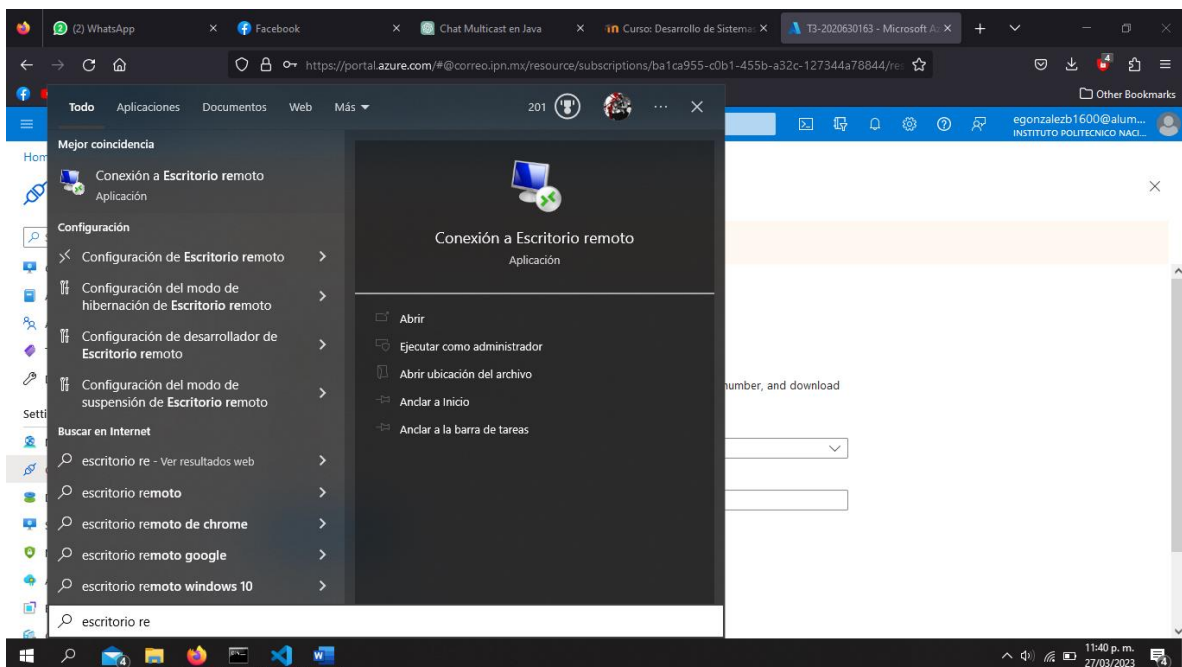
Esperamos a que acabe el proceso y se muestra lo siguiente:

The screenshot shows the Microsoft Azure portal interface after the deployment is complete. The browser address bar displays the URL: `https://portal.azure.com/#view/HubsExtension/DeploymentDetailsBlade/~/overview/id/%2Fsubscriptions%2Fba1c955-c0b1-455b-a32c-127344a78844`. The page title is "CreateVm-MicrosoftWindowsServer.WindowsServer-201-20230327233007 | Overview". The left sidebar contains navigation links: Overview, Inputs, Outputs, and Template. The main content area shows a deployment complete with the title "Your deployment is complete". It includes the deployment name "CreateVm-MicrosoftWindowsServer.WindowsS...", subscription "Azure for Students", and resource group "T3-2020630163_group". The start time is "3/27/2023, 11:36:15 PM" and the correlation ID is "094b5b14-a9be-48bc-86d6-12". Under "Next steps", there are recommendations: "Setup auto-shutdown", "Monitor VM health, performance and network dependencies", and "Run a script inside the virtual machine". There are buttons for "Go to resource" and "Create another VM". On the right side, there are recommendations for "Cost Management", "Microsoft Defender for Cloud", "Free Microsoft tutorials", and "Work with an expert". The bottom status bar shows the time "11:39 p. m." and date "27/03/2023".

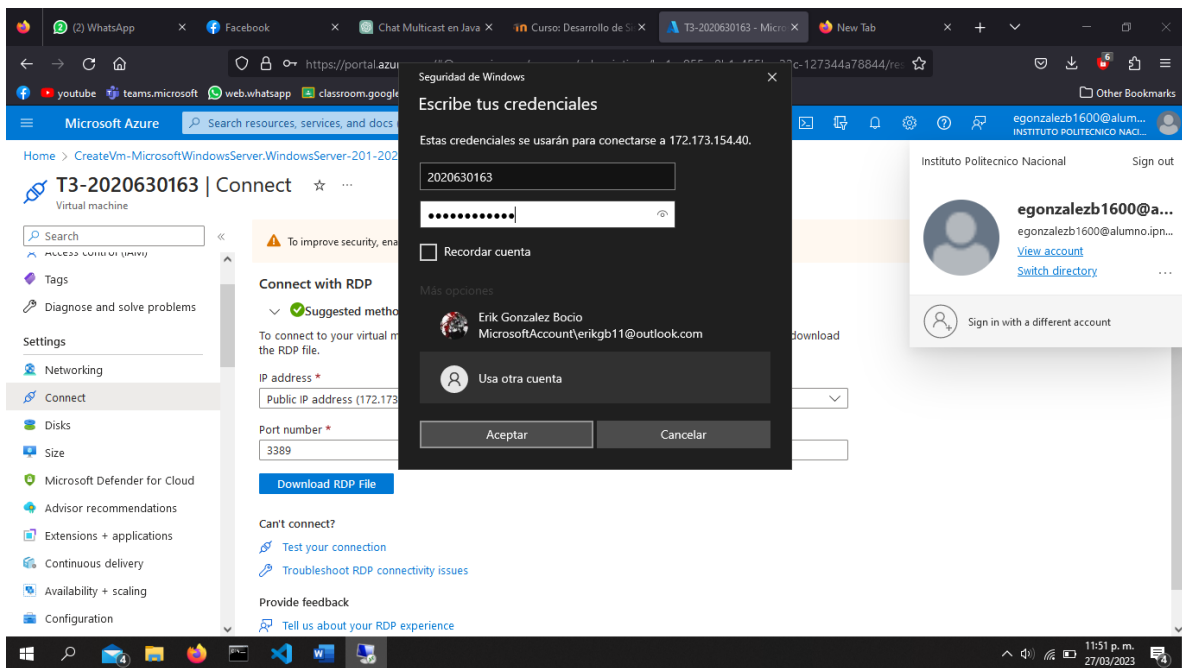
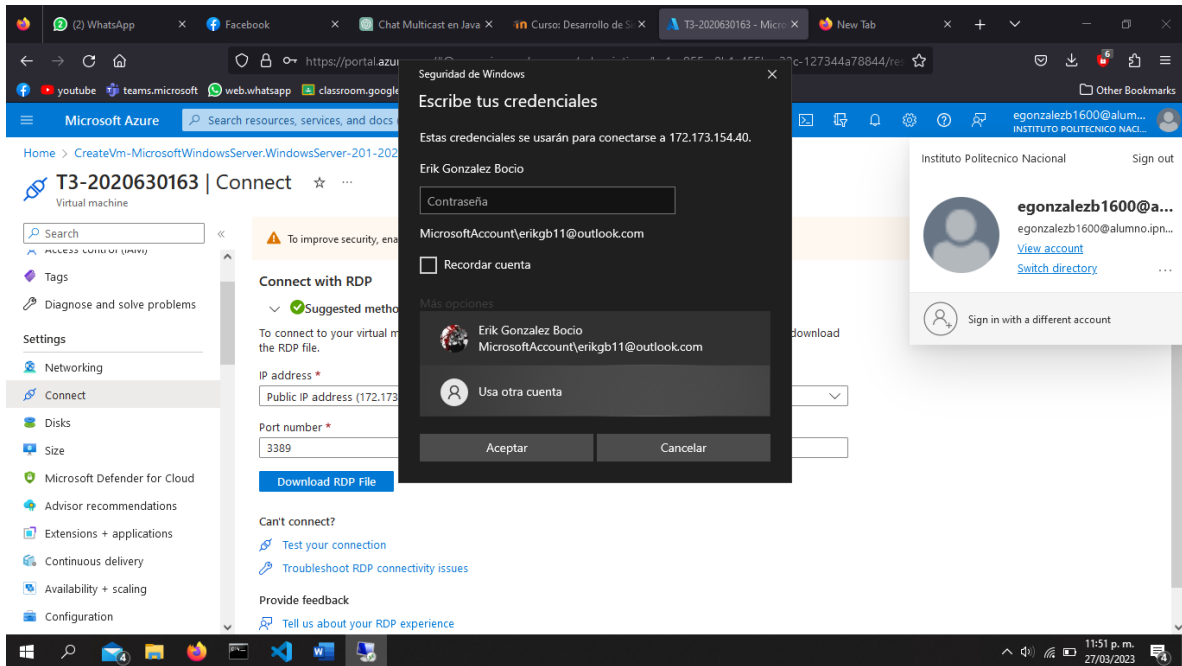
Aquí colocamos Go to resource y nos muestra la siguiente pantalla:



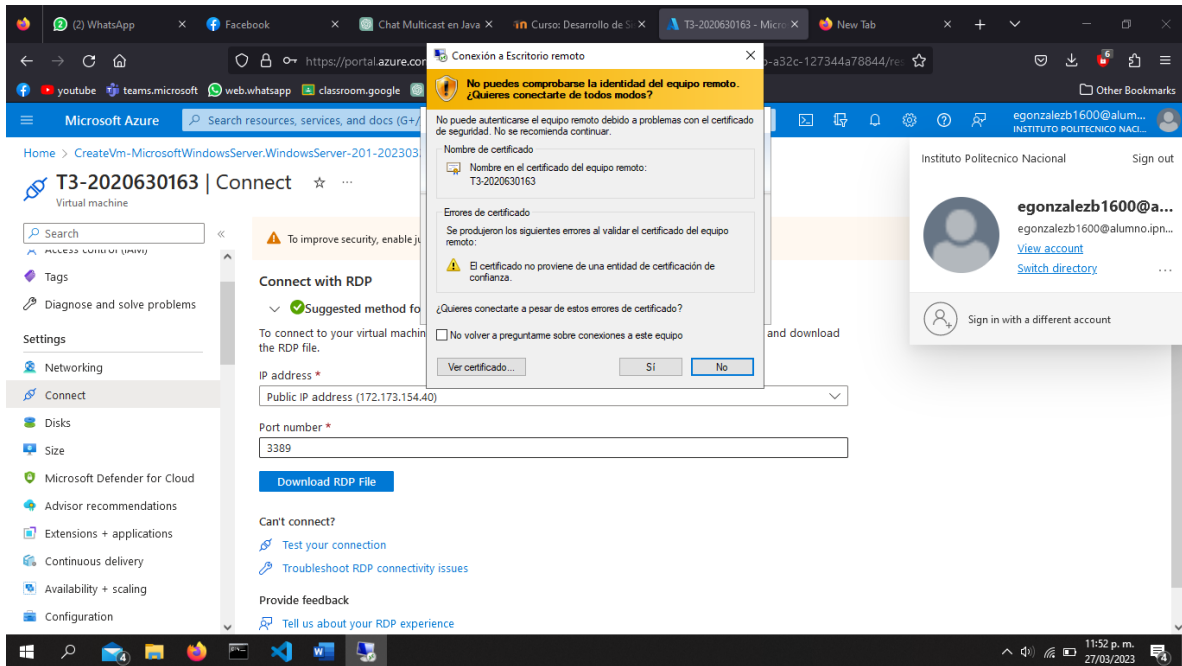
Nos desplazamos a conect y nos muestra los datos RDP, por lo que en nuestra máquina física buscamos Conexión a Escritorio Remoto.



Luego nos pedirá nuestras credenciales, por default nos mostrará nuestro usuario de la máquina física, por lo que tendremos que cambiarlo por el nombre que le dimos al usuario y colocar su contraseña como se ve en las siguientes capturas:



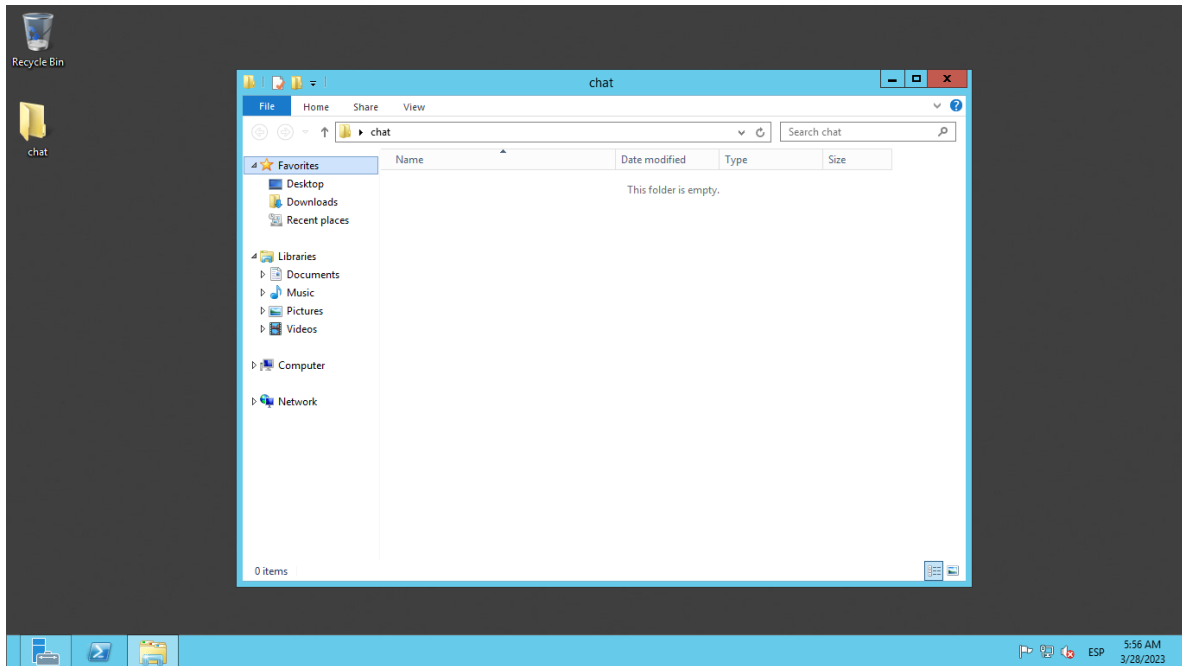
Luego se nos mostrara una ventana de seguridad, seleccionamos Si



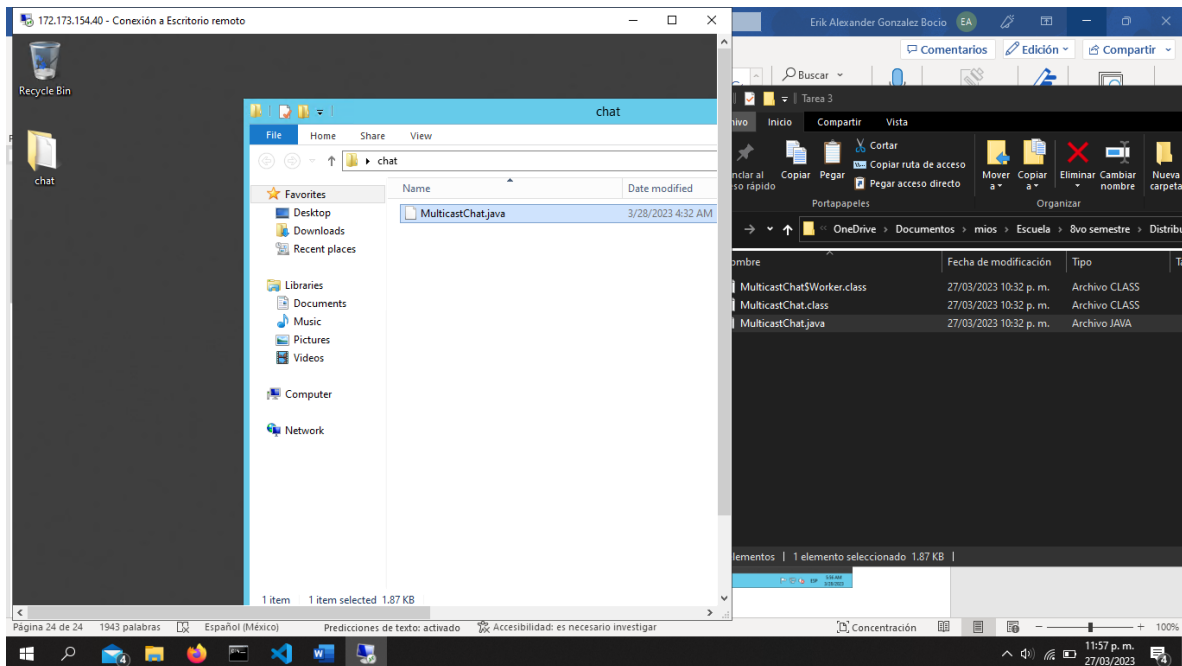
Y se abre una ventana nueva en pantalla completa, aquí se mostrará la máquina virtual por lo que iniciara y esperamos hasta ver la siguiente pantalla:



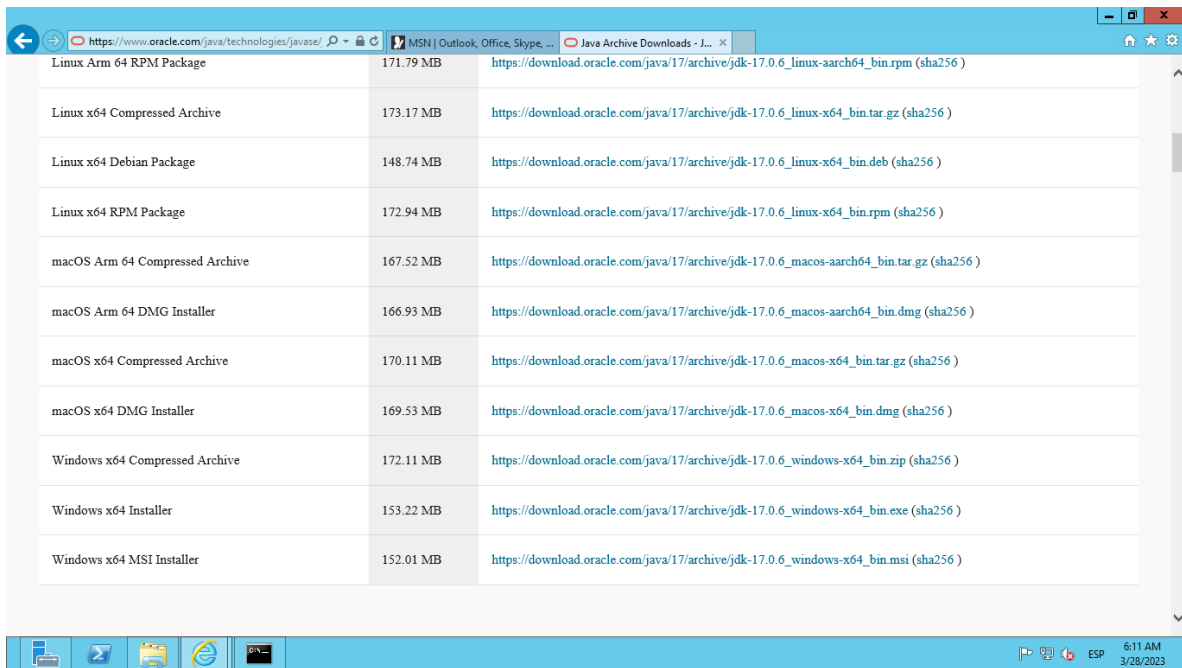
Creamos una carpeta en el escritorio donde colocaremos el programa



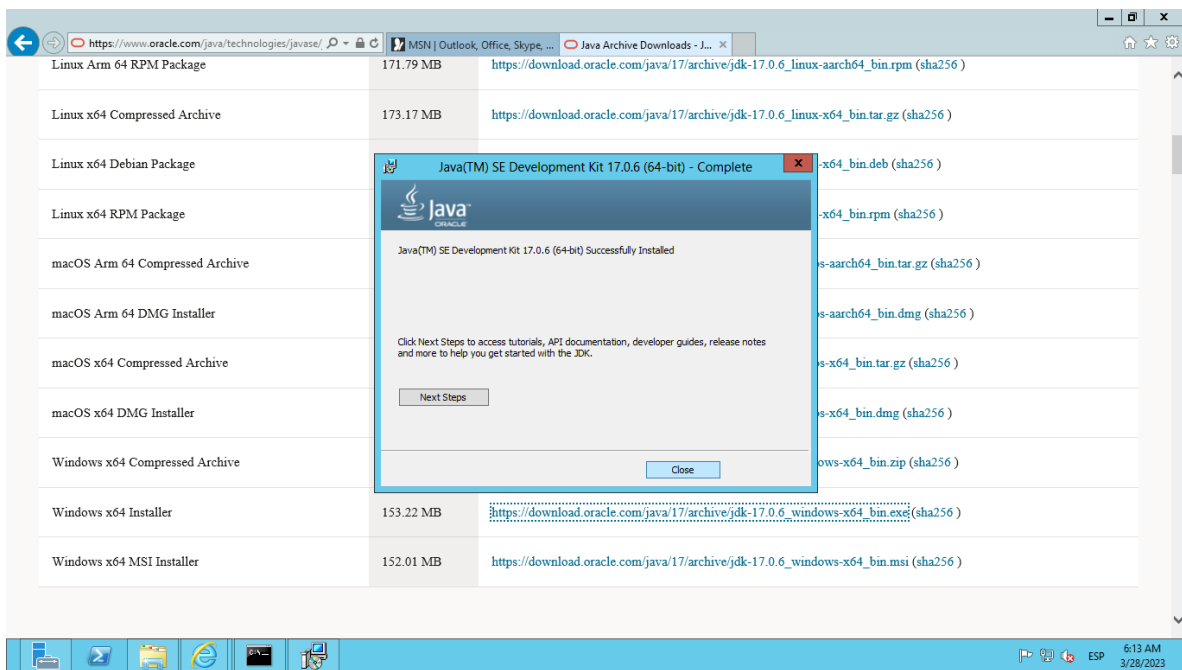
Y abrimos en nuestra máquina física el directorio donde se encuentra el programa, lo copiamos y en la máquina virtual lo pegamos, podemos usar el clásico CTRL C y CTRL V.



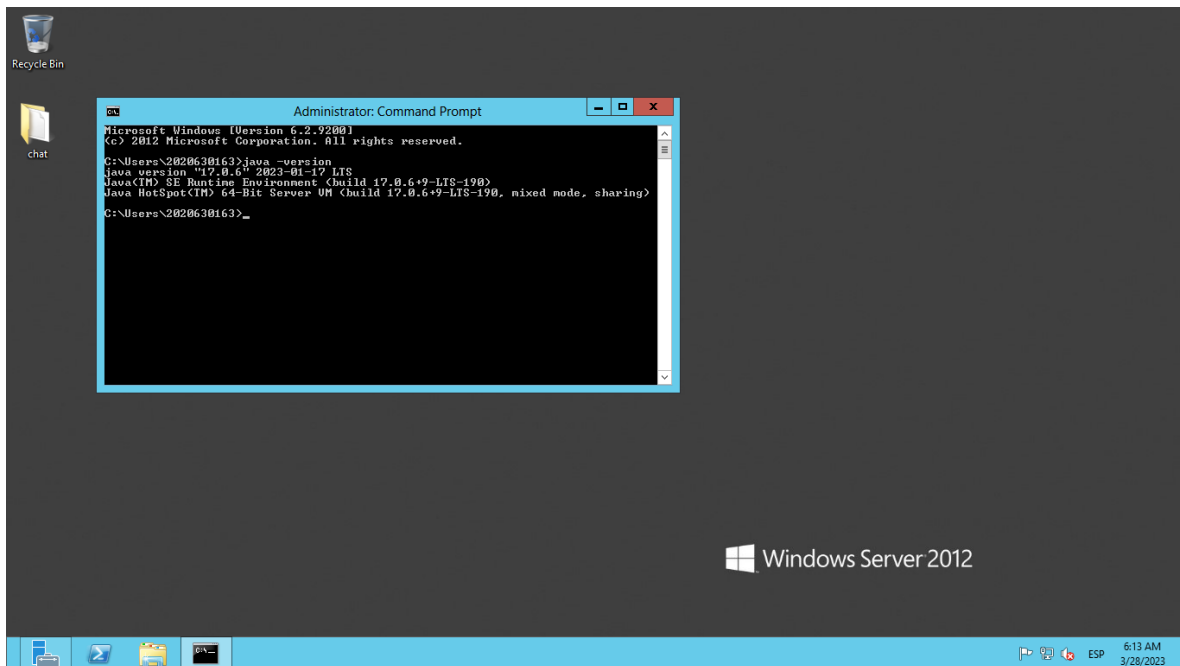
Es necesario que nuestra máquina virtual tenga el jdk de Java, por lo que es necesario instalarlo, por lo que abrimos Internet Explorer y buscamos el JDK de Java, lo descargamos y lo instalamos.



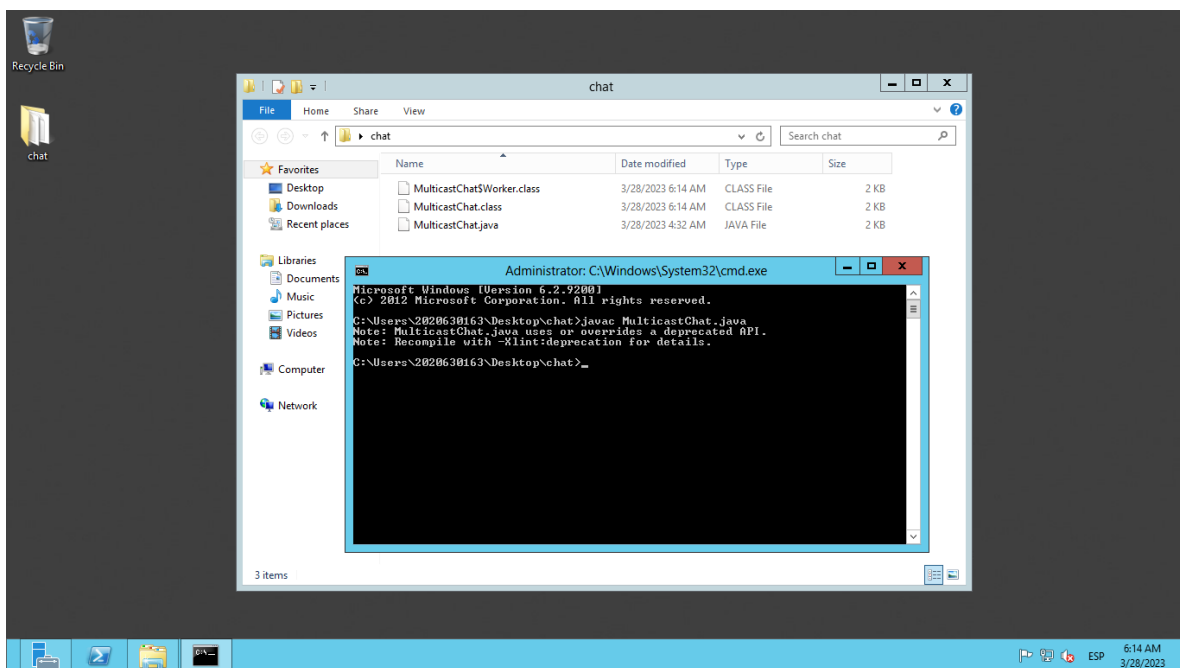
Se instala



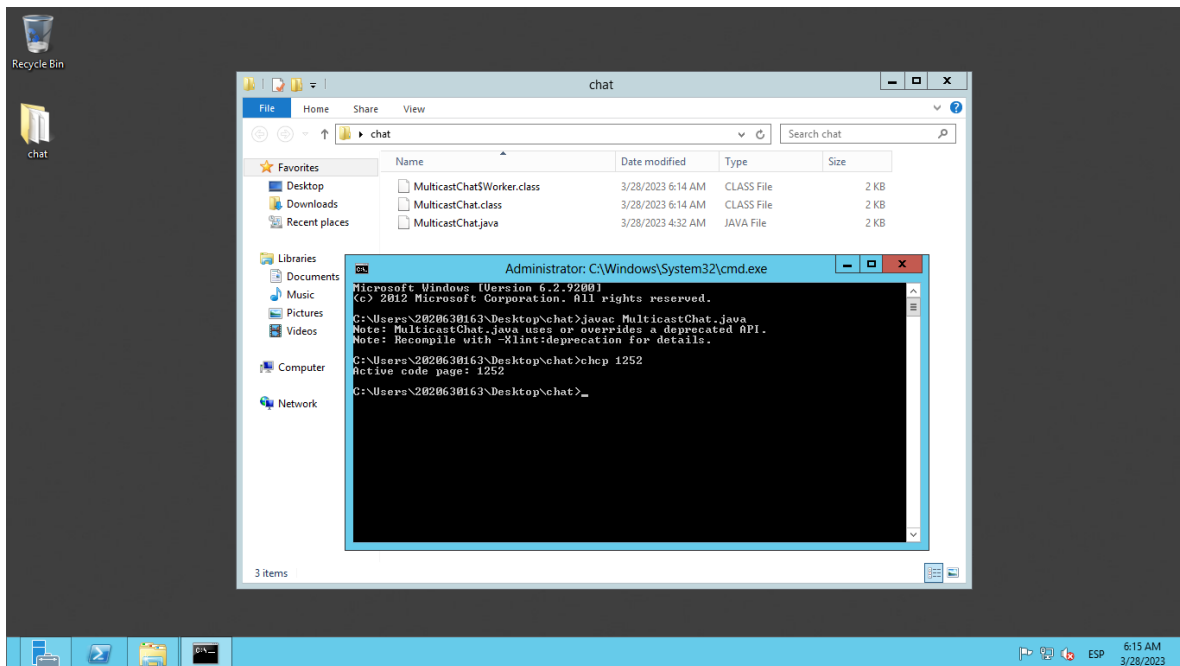
Y ahora con el comando `java -version` en el cmd podemos ver la versión de Java instalada.



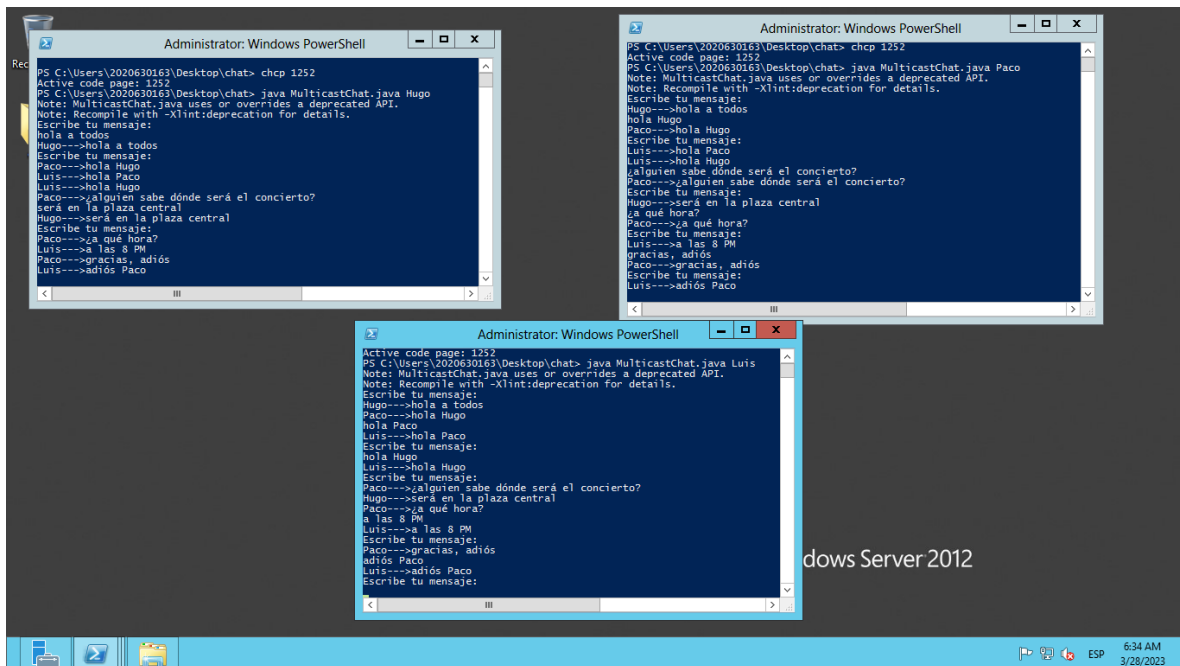
Una vez hecho esto usamos el comando javac para compilar el archivo, aquí podemos notar que en este caso se muestra un mensaje, esto es debido al programa ya que usa un API deprecado, pero no afecta la funcionalidad.



Como pudimos ver en la plática con ChatGPT es necesario que en cada terminal que usemos se coloque el comando chcp 1252, esto para que se pueden notar los caracteres especiales y acentos.



Finalmente, con 3 terminales abiertas y ejecutando el programa para los 3 personajes del chat se muestra el siguiente resultado:



Conclusión

En esta ocasión se pudo poner en practica el uso de datagramas para mandar mensajes a un grupo usando un chat multicast, me pareció interesante ya que se pudo observar las diferencias de usar datagramas a sockets de flujo, además que es interesante como trabaja el chat multicast haciendo que todos sus nodos reciban el mensaje que se envía.

También me pareció interesante la implementación de una máquina virtual con Windows Server ya que es un poco diferente la manera en que se hace la implementación con respecto a las máquinas que hacíamos con Ubuntu, además me gusto la forma en la que nos conectamos a nuestra máquina virtual ya que el usarla como si fuera una ventana es como si usáramos un hipervisor para trabajar.

Finalmente, esta práctica fue más sencilla trabajarla con ChatGPT ya que era un programa más pequeño y por lo tanto no había problemas con la perdida de código y era más fácil plantearle errores o cambios que necesitáramos.