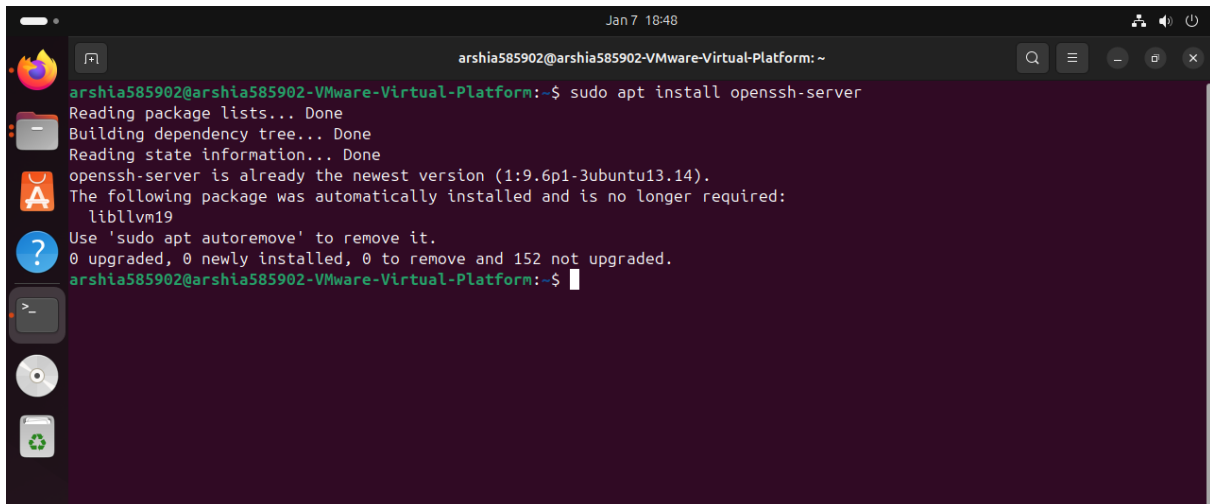


Template Week 6 – Networking

Student number: 585902

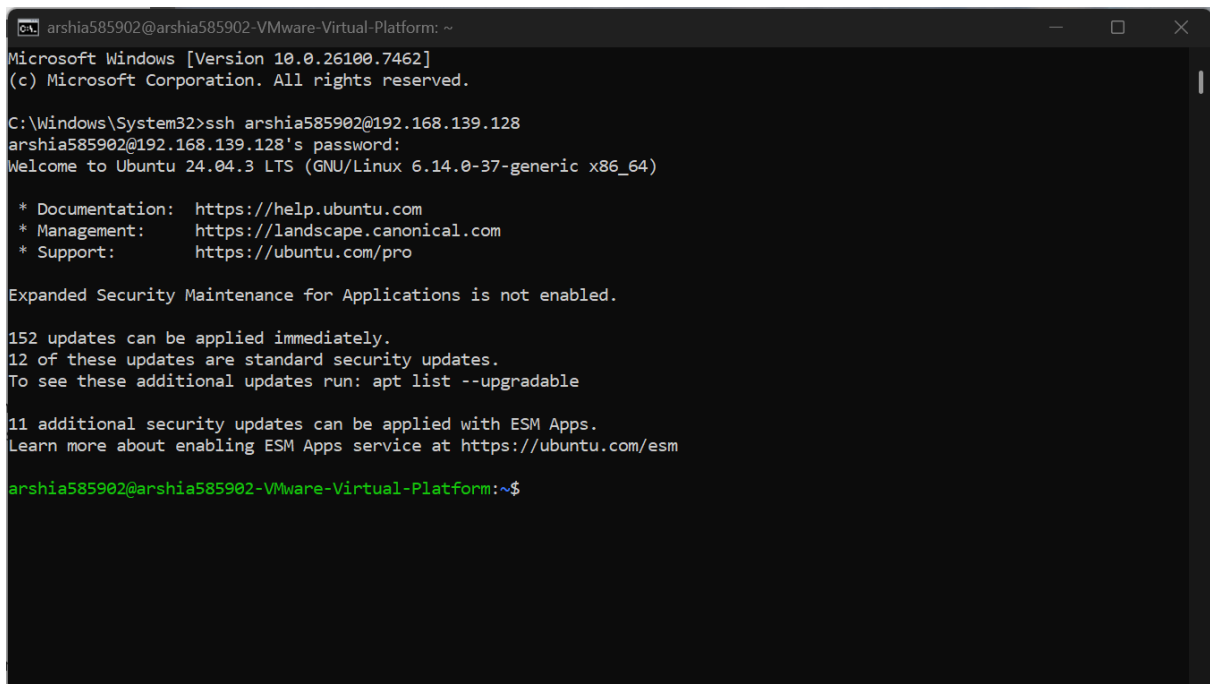
Assignment 6.1: Working from home

Screenshot installation openssh-server:



```
arshia585902@arshia585902-VMware-Virtual-Platform: ~  
arshia585902@arshia585902-VMware-Virtual-Platform:~$ sudo apt install openssh-server  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
openssh-server is already the newest version (1:9.6p1-3ubuntu13.14).  
The following package was automatically installed and is no longer required:  
  libllvm19  
Use 'sudo apt autoremove' to remove it.  
0 upgraded, 0 newly installed, 0 to remove and 152 not upgraded.  
arshia585902@arshia585902-VMware-Virtual-Platform:~$
```

Screenshot successful SSH command execution:



```
arshia585902@arshia585902-VMware-Virtual-Platform: ~  
Microsoft Windows [Version 10.0.26100.7462]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Windows\System32>ssh arshia585902@192.168.139.128  
arshia585902@192.168.139.128's password:  
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-37-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/pro  
  
Expanded Security Maintenance for Applications is not enabled.  
  
152 updates can be applied immediately.  
12 of these updates are standard security updates.  
To see these additional updates run: apt list --upgradable  
  
11 additional security updates can be applied with ESM Apps.  
Learn more about enabling ESM Apps service at https://ubuntu.com/esm  
  
arshia585902@arshia585902-VMware-Virtual-Platform:~$
```

Screenshot successful execution SCP command:

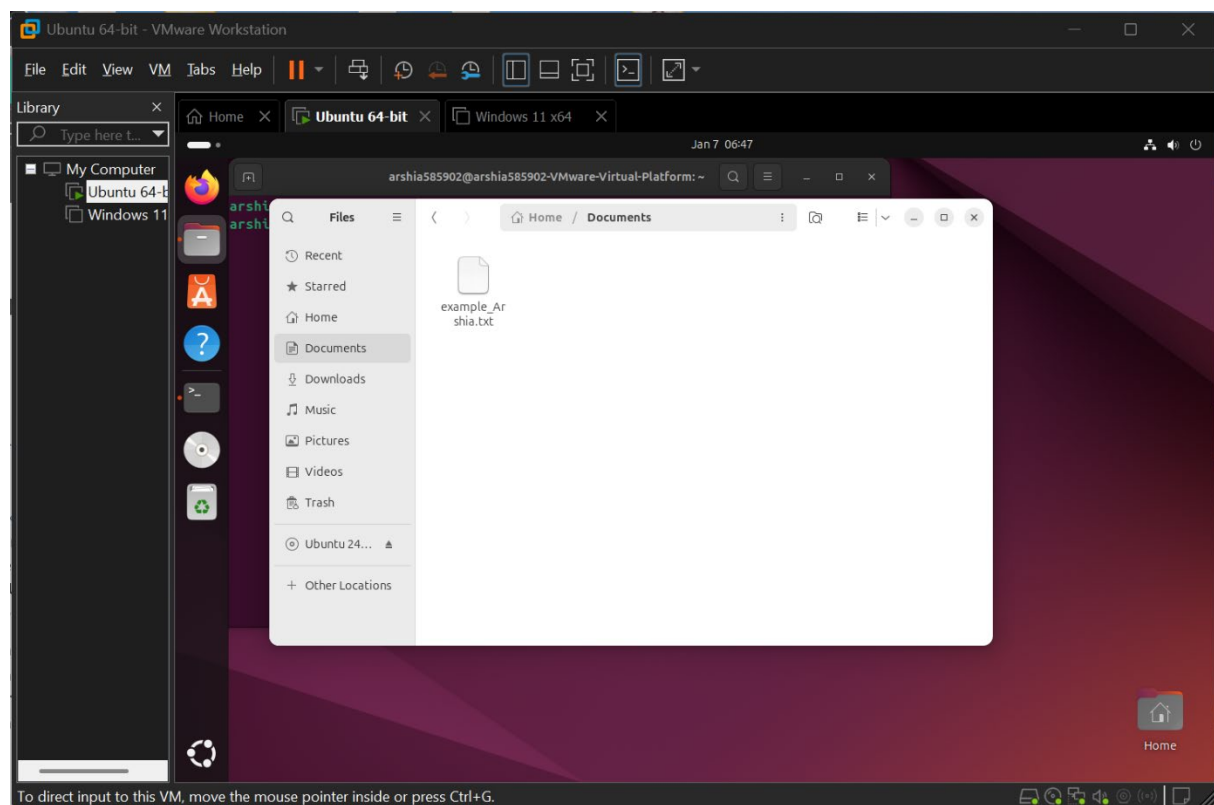
```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.26100.7462]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>scp "C:\Users\shahr\Desktop\example_Arshia.txt" arshia585902@192.168.139.128:~/Documents
arshia585902@192.168.139.128's password:
scp: dest open "~/Documents": No such file or directory
scp: failed to upload file C:/Users/shahr/Desktop/example_Arshia.txt to ~/Documents

C:\Windows\System32>scp "C:\Users\shahr\Desktop\example_Arshia.txt" arshia585902@192.168.139.128:/home/arshia585902/Docu
ments
arshia585902@192.168.139.128's password:
example_Arshia.txt                                100%   0   0.0KB/s   00:00

C:\Windows\System32>
```

Proof that the file was transferred successfully:



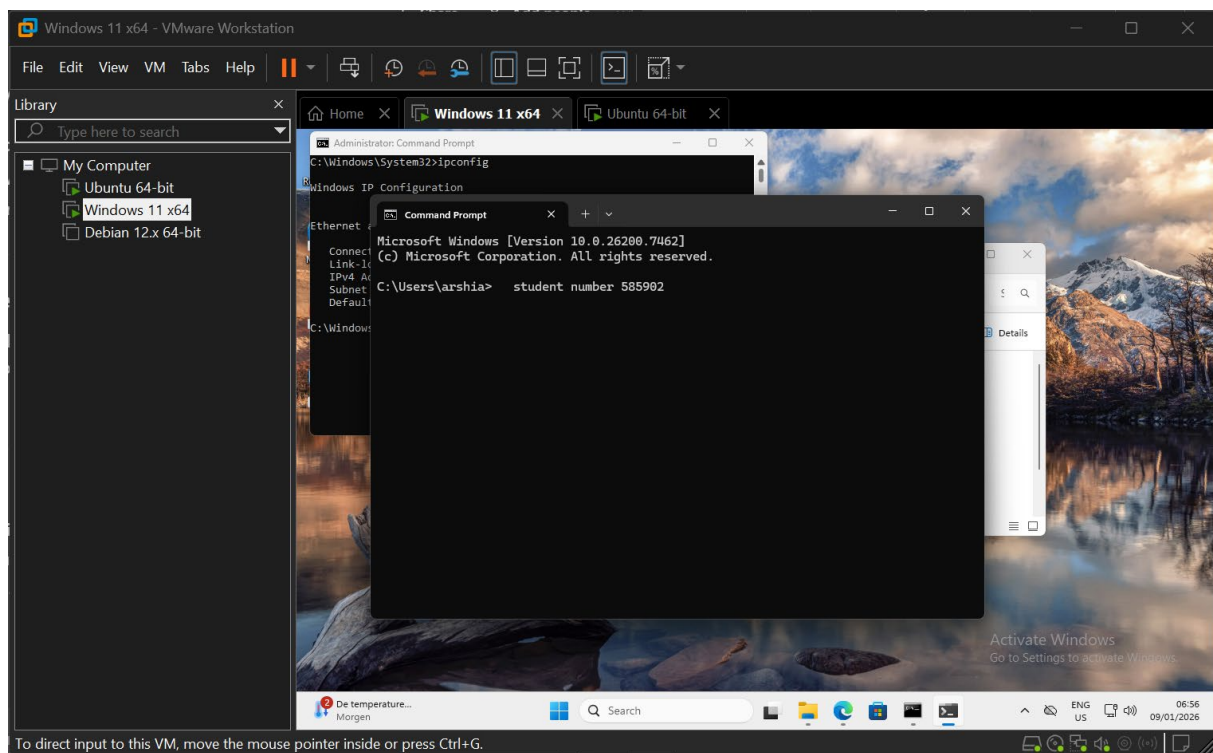
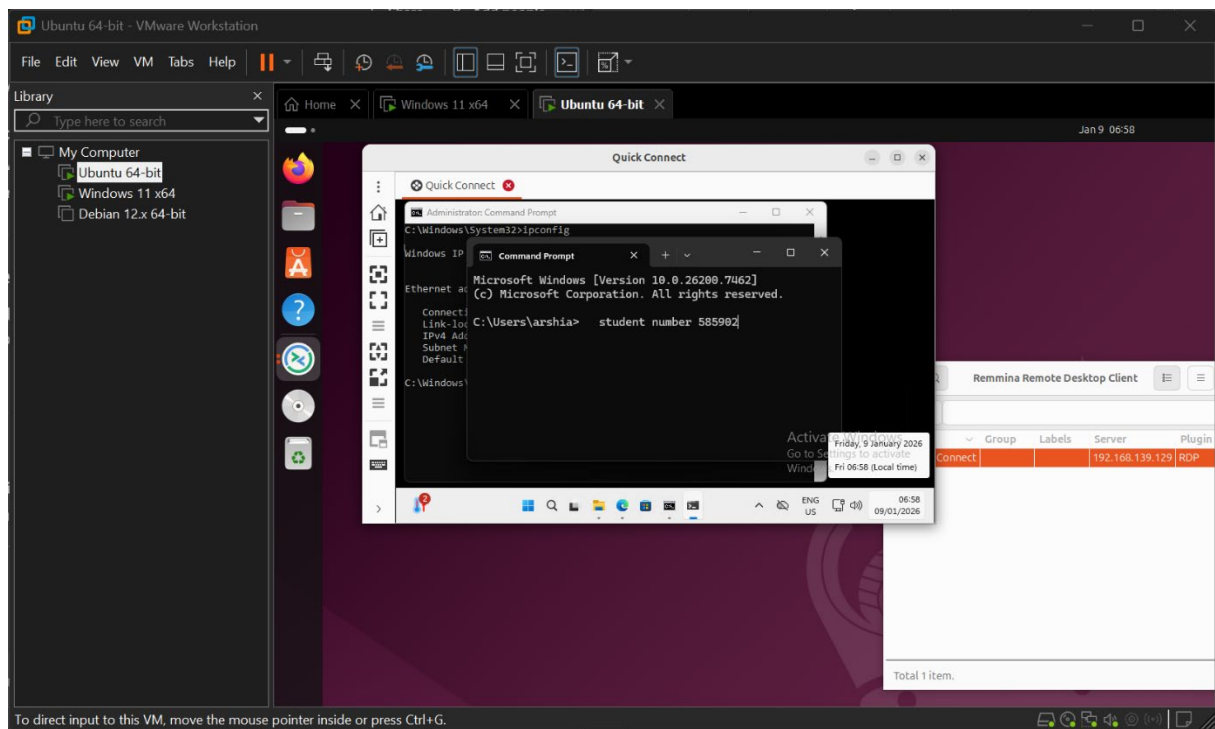
On windows I went to settings > system > remote desktop and enabled that which resulted in RDP server being enabled on Windows

In the same section there was a place where I could add accounts that I wanted to grant access but since im using admin on Ubuntu, I didn't need to add anything in there; because admins are allowed by default

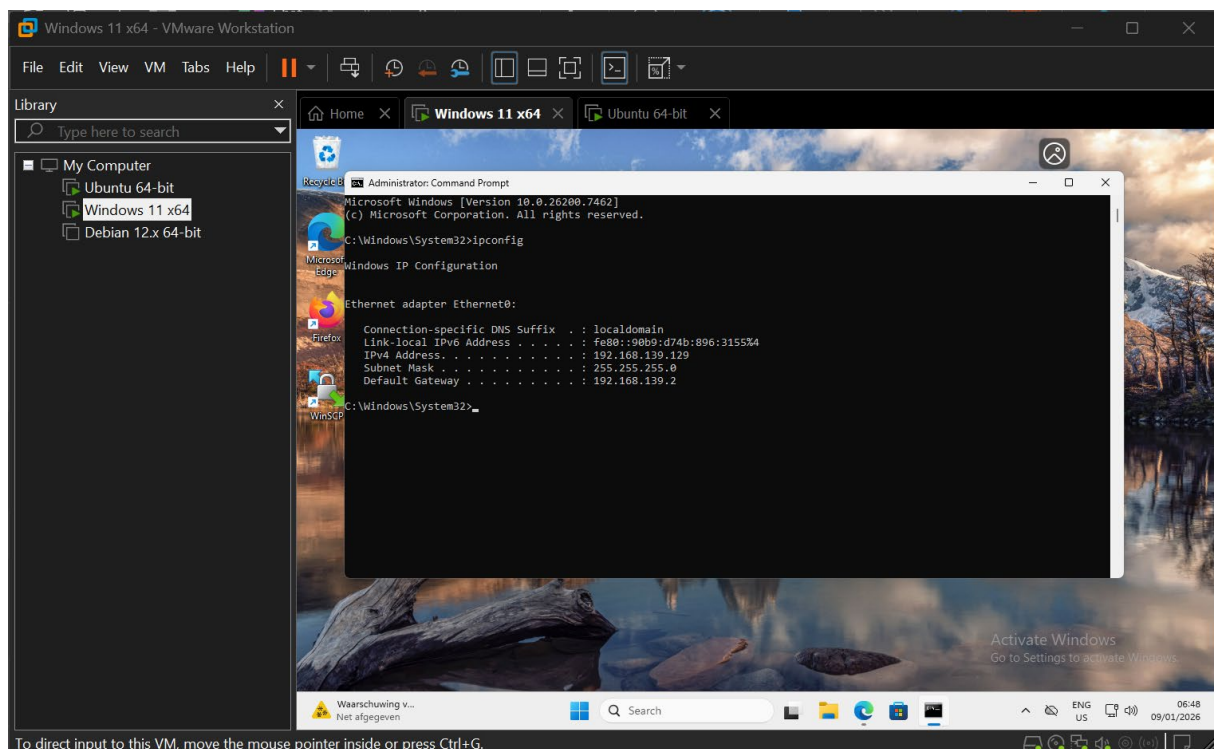
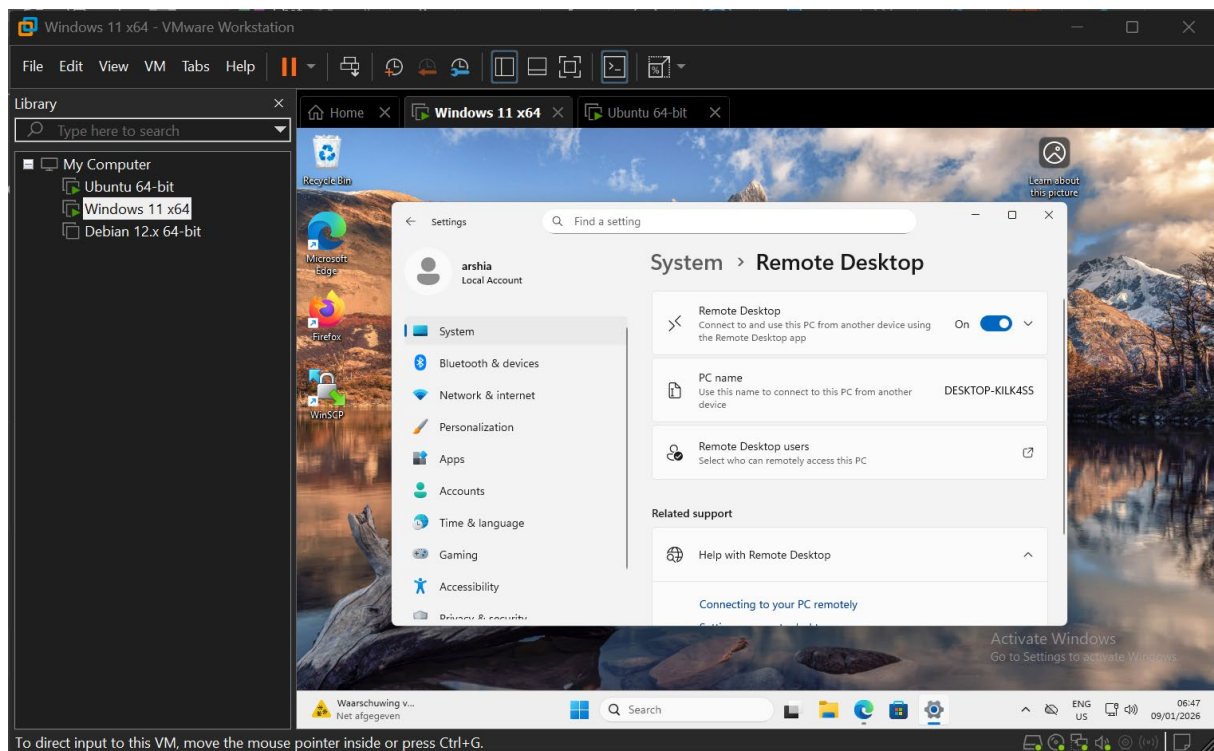
I then proceeded to enter that IP address in Remmina On Ubuntu where it applied

As primary proof, I took screen shots of the configurations I set both on Ubuntu and windows, The setup that resulted in successfully accessing the Windows VM

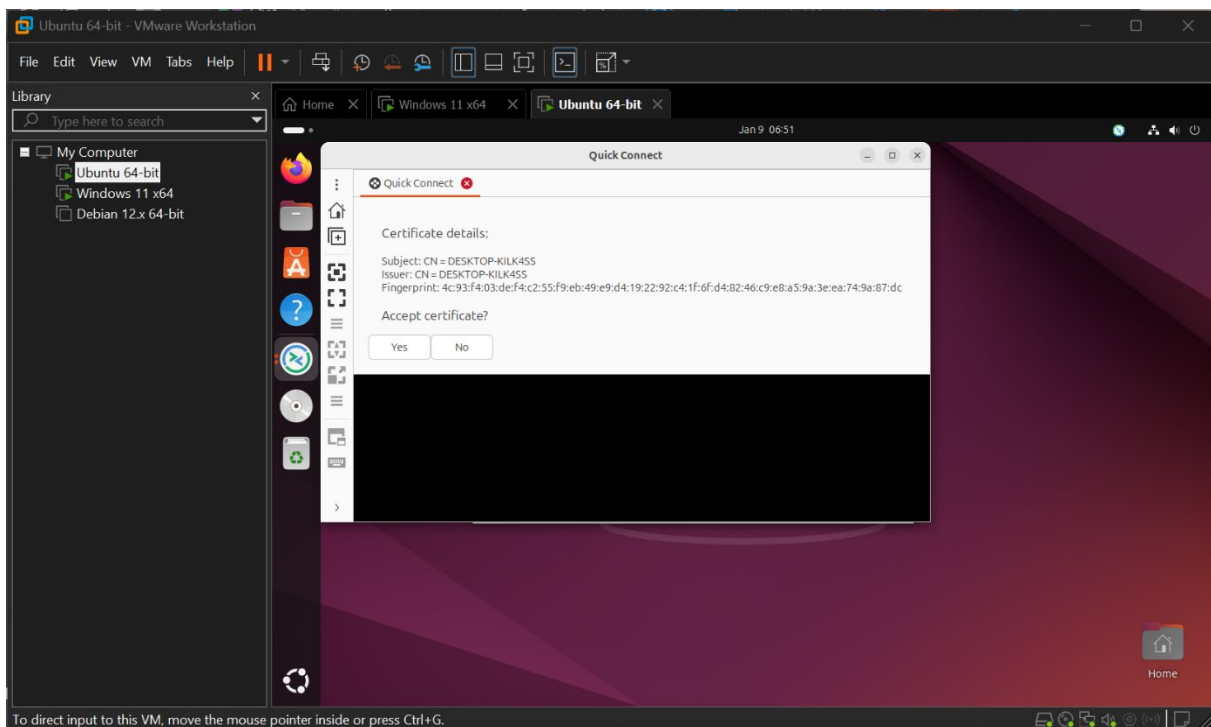
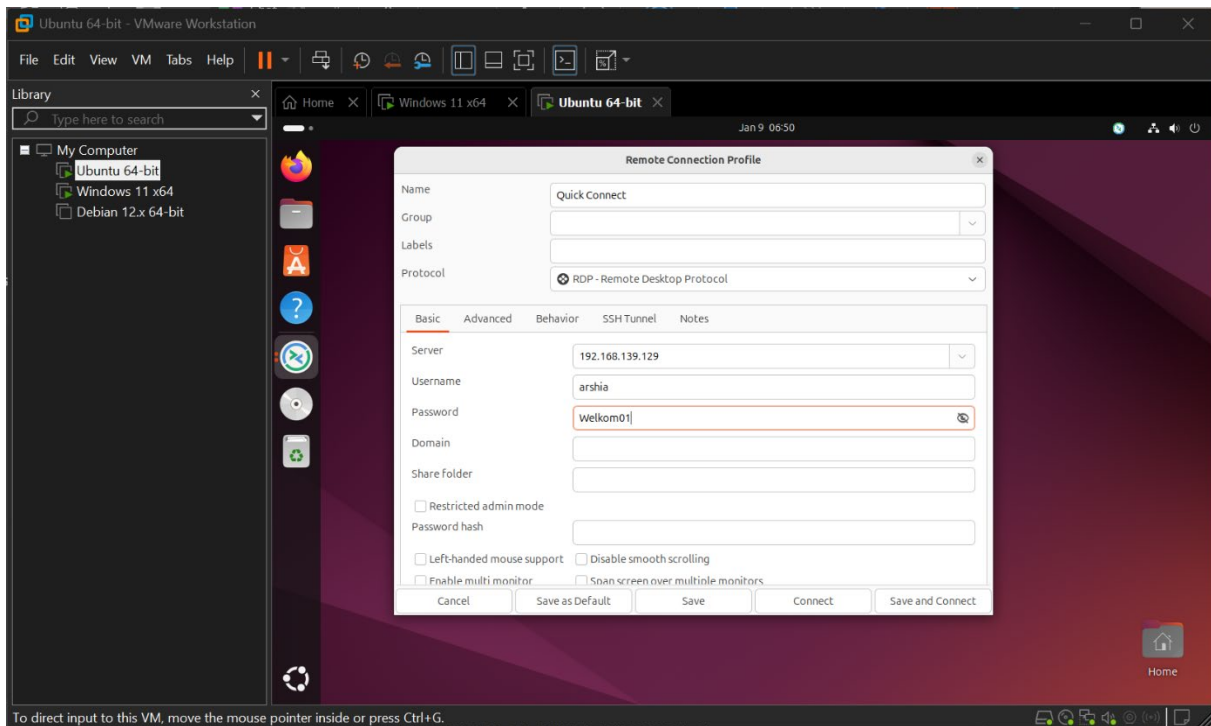
Generally the concept is as follows: remote desktop must be enabled on Windows and both VMs should be connected to the same network. Remmina must connect with RDP protocol, using the IP address of the Windows VM



Applied settings :



Remmina configuration:



Assignment 6.2: IP addresses websites

Relevant screenshots nslookup command:

```
Administrator: Command Prompt - nslookup
Microsoft Windows [Version 10.0.26100.7462]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>nslookup
Default Server: UnKnown
Address: 10.40.66.177

> amazon.com
Server: UnKnown
Address: 10.40.66.177

Non-authoritative answer:
Name: amazon.com
Addresses: 98.87.170.71
          98.82.161.185
          98.87.170.74

> google.com
Server: UnKnown
Address: 10.40.66.177

Non-authoritative answer:
Name: google.com
Addresses: 2a00:1450:400e:80f::200e
          142.251.39.142

> one.one.one.one
Server: UnKnown
Address: 10.40.66.177
```

```
Administrator: Command Prompt - nslookup
Non-authoritative answer:
Name: one.one.one.one
Addresses: 2606:4700:4700::1001
          2606:4700:4700::1111
          1.0.0.1
          1.1.1.1

> dns.google.com
Server: UnKnown
Address: 10.40.66.177

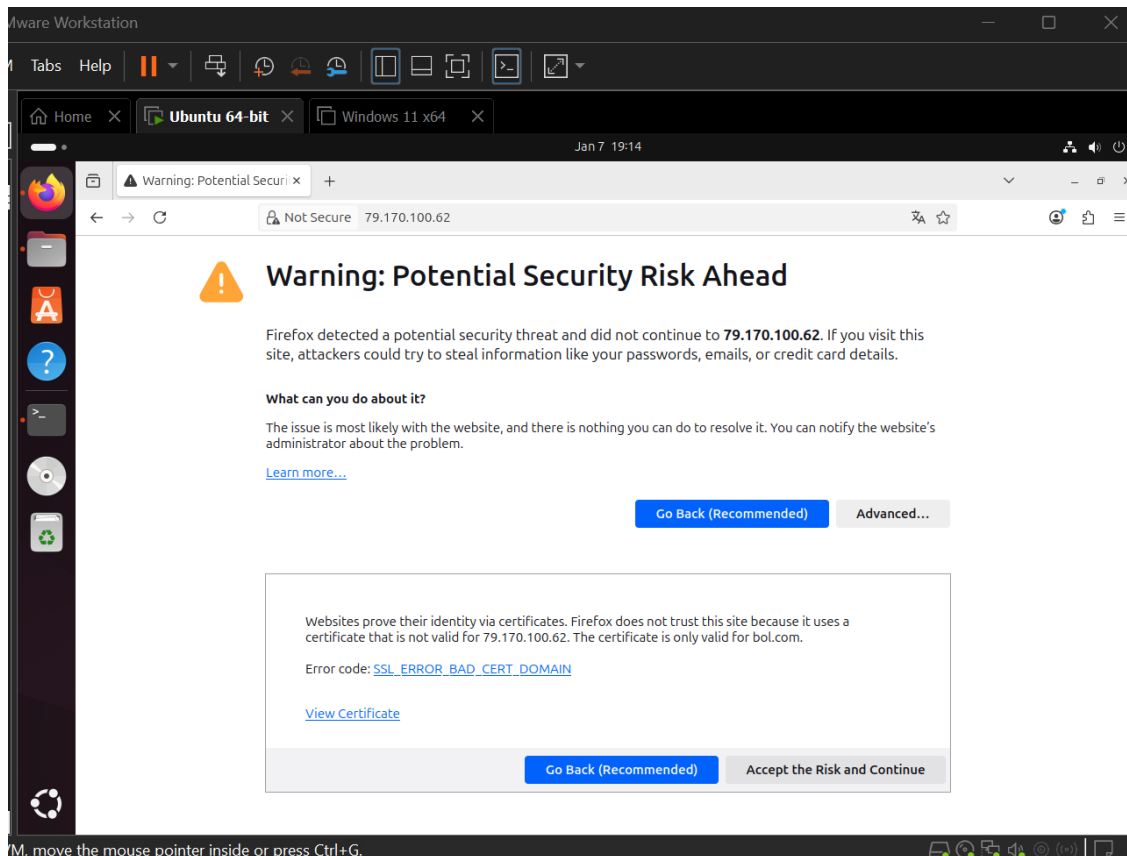
Non-authoritative answer:
Name: dns.google.com
Addresses: 2001:4860:4860::8888
          2001:4860:4860::8844
          8.8.4.4
          8.8.8.8

> bol.com
Server: UnKnown
Address: 10.40.66.177

Non-authoritative answer:
Name: bol.com
Address: 79.170.100.62

> w3schools.com
Server: UnKnown
Address: 10.40.66.177
```

Screenshot website visit via IP address:



Assignment 6.3: subnetting

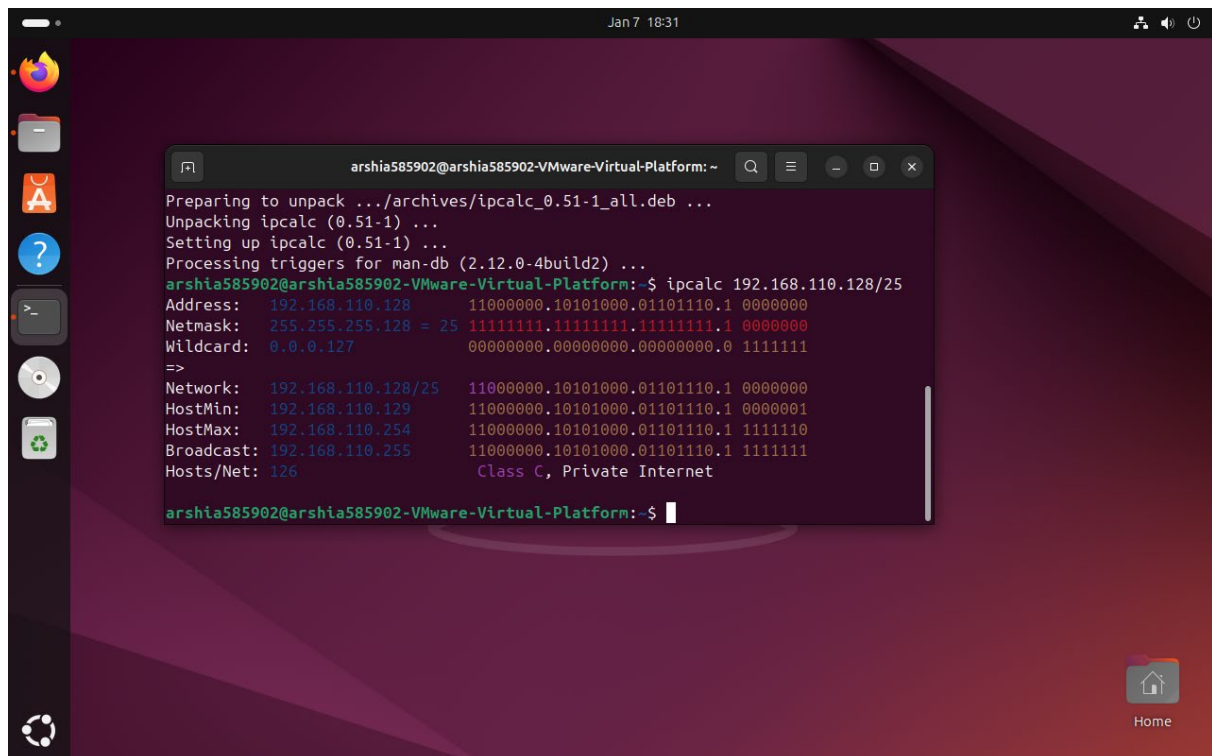
How many IP addresses are in this network configuration 192.168.110.128/25?

128 addresses, 126 of which usable for hosts because the first one (being network ip) and the last one (being broadcast ip) are not usable for that – [calculation explained bellow]

What is the usable IP range to hand out to the connected computers?

From 192.168.110.129/25 to 192.168.110.254/25

Check your two previous answers with this Linux command: `ipcalc 192.168.110.128/25`



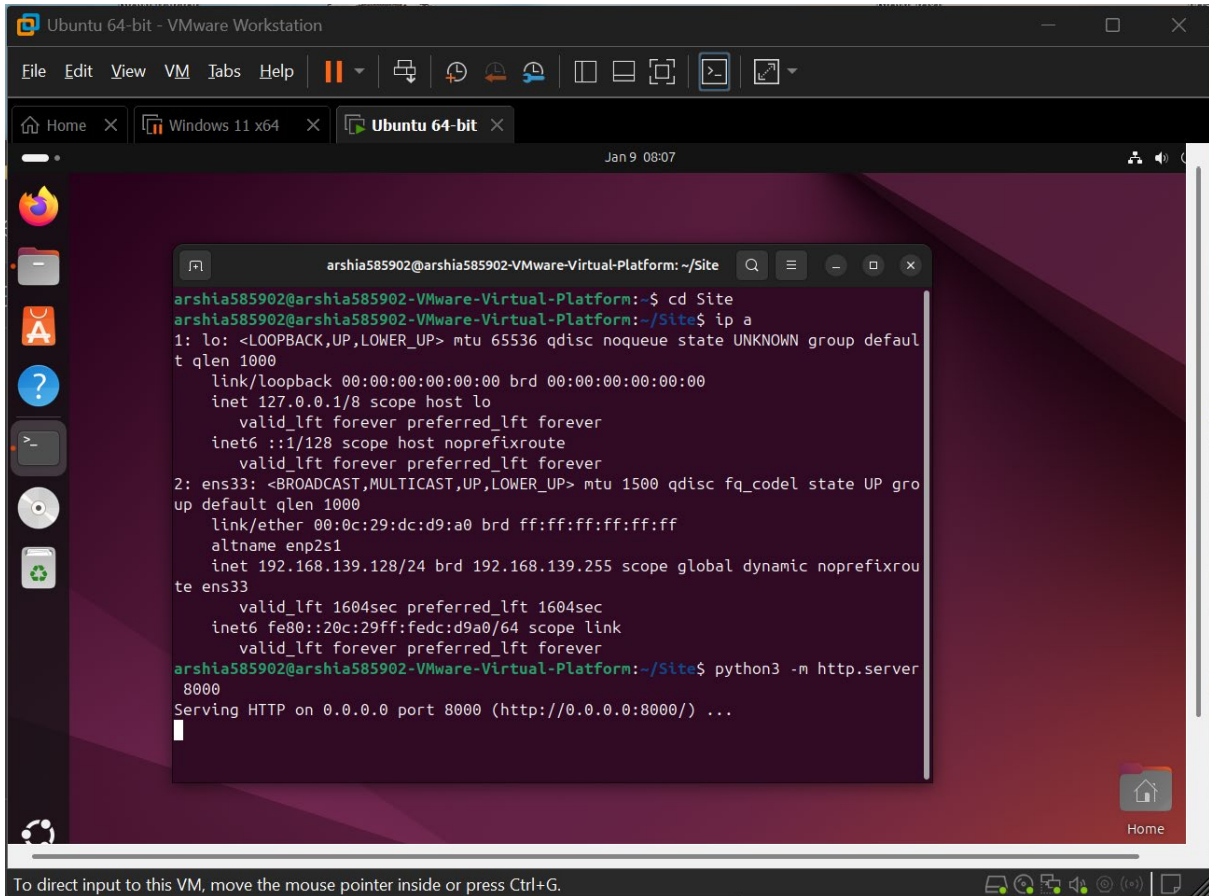
```
arshia585902@arshia585902-VMware-Virtual-Platform: ~
Preparing to unpack .../archives/ipcalc_0.51-1_all.deb ...
Unpacking ipcalc (0.51-1) ...
Setting up ipcalc (0.51-1) ...
Processing triggers for man-db (2.12.0-4build2) ...
arshia585902@arshia585902-VMware-Virtual-Platform:~$ ipcalc 192.168.110.128/25
Address: 192.168.110.128 11000000.10101000.01101110.1 0000000
Netmask: 255.255.255.128 = 25 11111111.11111111.11111111.1 00000000
Wildcard: 0.0.0.127 00000000.00000000.00000000.0 11111111
=>
Network: 192.168.110.128/25 11000000.10101000.01101110.1 0000000
HostMin: 192.168.110.129 11000000.10101000.01101110.1 0000001
HostMax: 192.168.110.254 11000000.10101000.01101110.1 1111110
Broadcast: 192.168.110.255 11000000.10101000.01101110.1 1111111
Hosts/Net: 126 Class C, Private Internet
arshia585902@arshia585902-VMware-Virtual-Platform:~$
```

Explain the above calculation in your own words.

If it wasn't a '/25' subnet, there would have been four octets or groups of 8 bits which together would have formed 32 bits which would have been 2^{32} addresses. But now, 25 bit is subtracted from 32 since those 2^{25} addresses are reserved for the network and can't be given to hosts. that leaves us with 7 bits or 2^7 addresses to be given to hosts; (two of which are reduced again, for network and broadcast). That '/25' subnet mask makes the block size 128 (or 2^7) and the boundaries of that subnet should be multiples of the block size which gives us two possible subnets per last octet, 0-127 and 128-255. the last octet determines the network address and that address is the beginning of the range. Since that says 128 then it means that the range should be from 128 to 255 but the first one and the last one cant be used; therefore what actually goes to hosts are from ~.129/25 to ~.254/25

Assignment 6.4: HTML

Screenshot IP address Ubuntu VM:

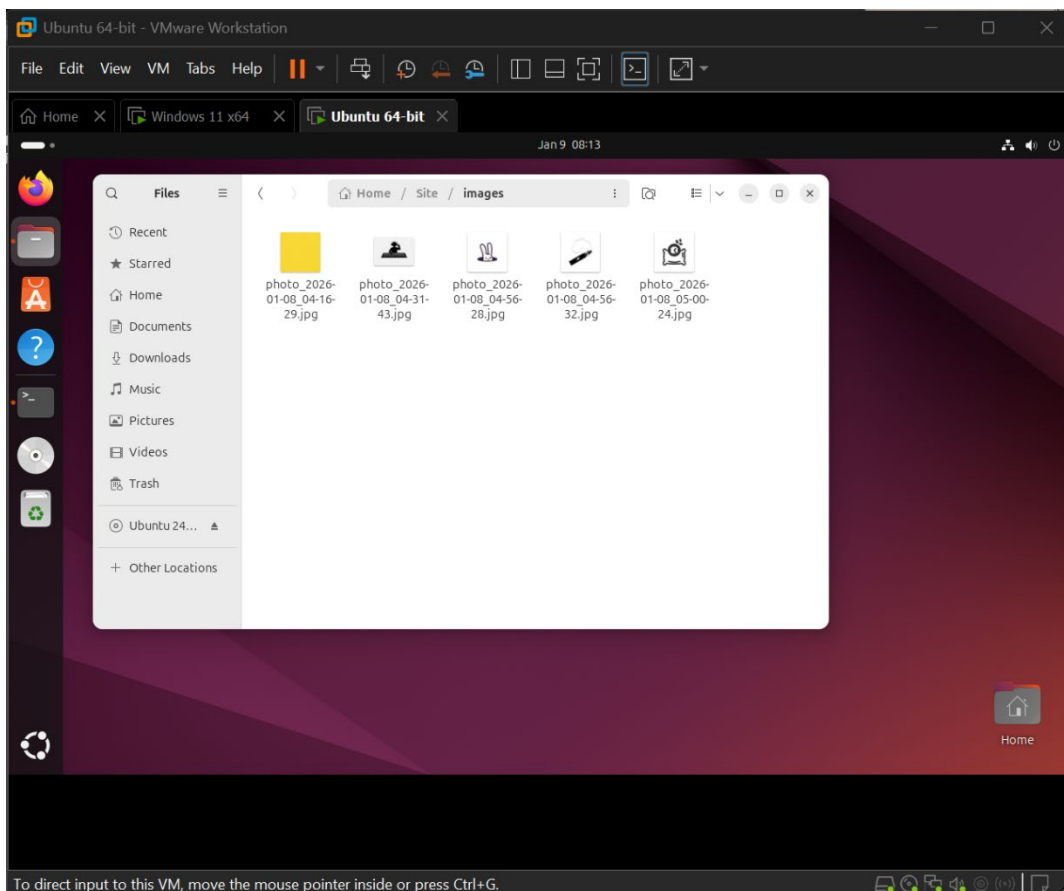
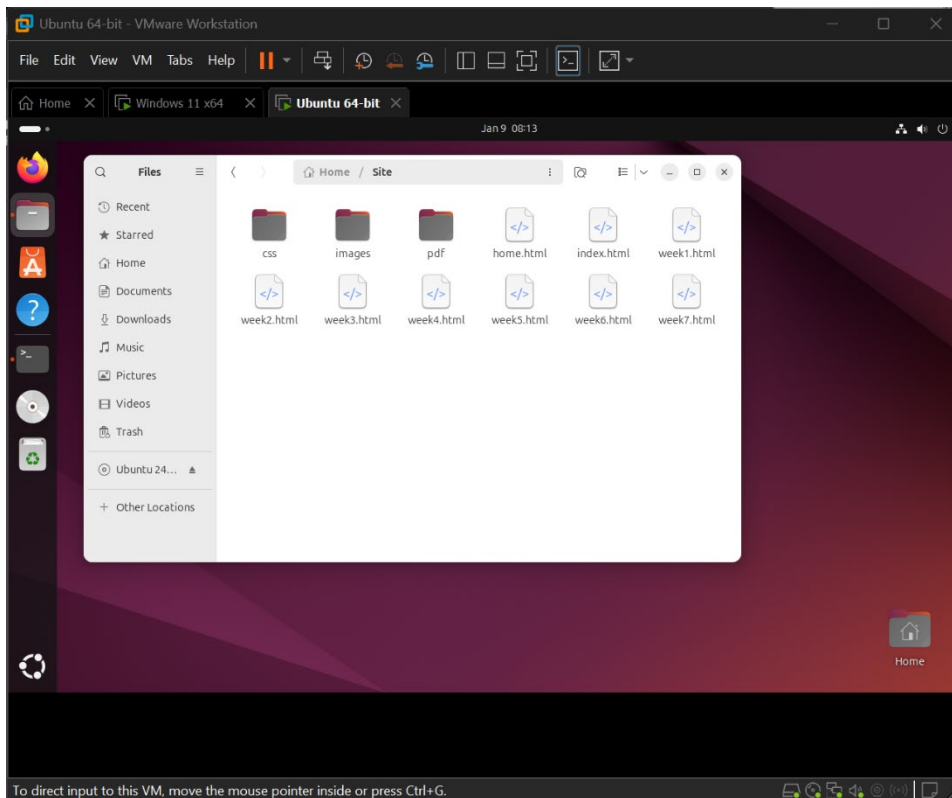


The screenshot shows a VMware Workstation window titled "Ubuntu 64-bit - VMware Workstation". Inside the VM, the Ubuntu desktop is visible with a terminal window open. The terminal shows the following commands and output:

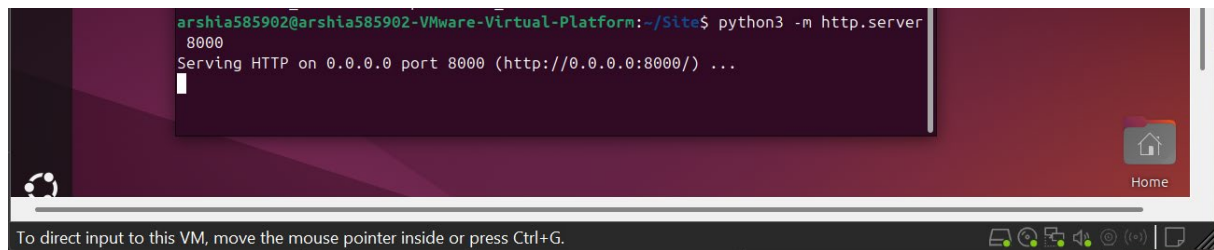
```
arshia585902@arshia585902-VMware-Virtual-Platform: ~/Site
arshia585902@arshia585902-VMware-Virtual-Platform:~/Site$ cd Site
arshia585902@arshia585902-VMware-Virtual-Platform:~/Site$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:dc:d9:a0 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.139.128/24 brd 192.168.139.255 scope global dynamic noprefixroute ens33
        valid_lft 1604sec preferred_lft 1604sec
    inet6 fe80::20c:29ff:fedc:d9a0/64 scope link
        valid_lft forever preferred_lft forever
arshia585902@arshia585902-VMware-Virtual-Platform:~/Site$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

The terminal window is titled "arshia585902@arshia585902-VMware-Virtual-Platform: ~/Site". The output of the 'ip a' command shows the configuration for the loopback interface 'lo' and the ethernet interface 'ens33'. The 'ens33' interface is configured with the IP address 192.168.139.128/24. The output of the 'python3 -m http.server 8000' command shows that the web server is running on port 8000.

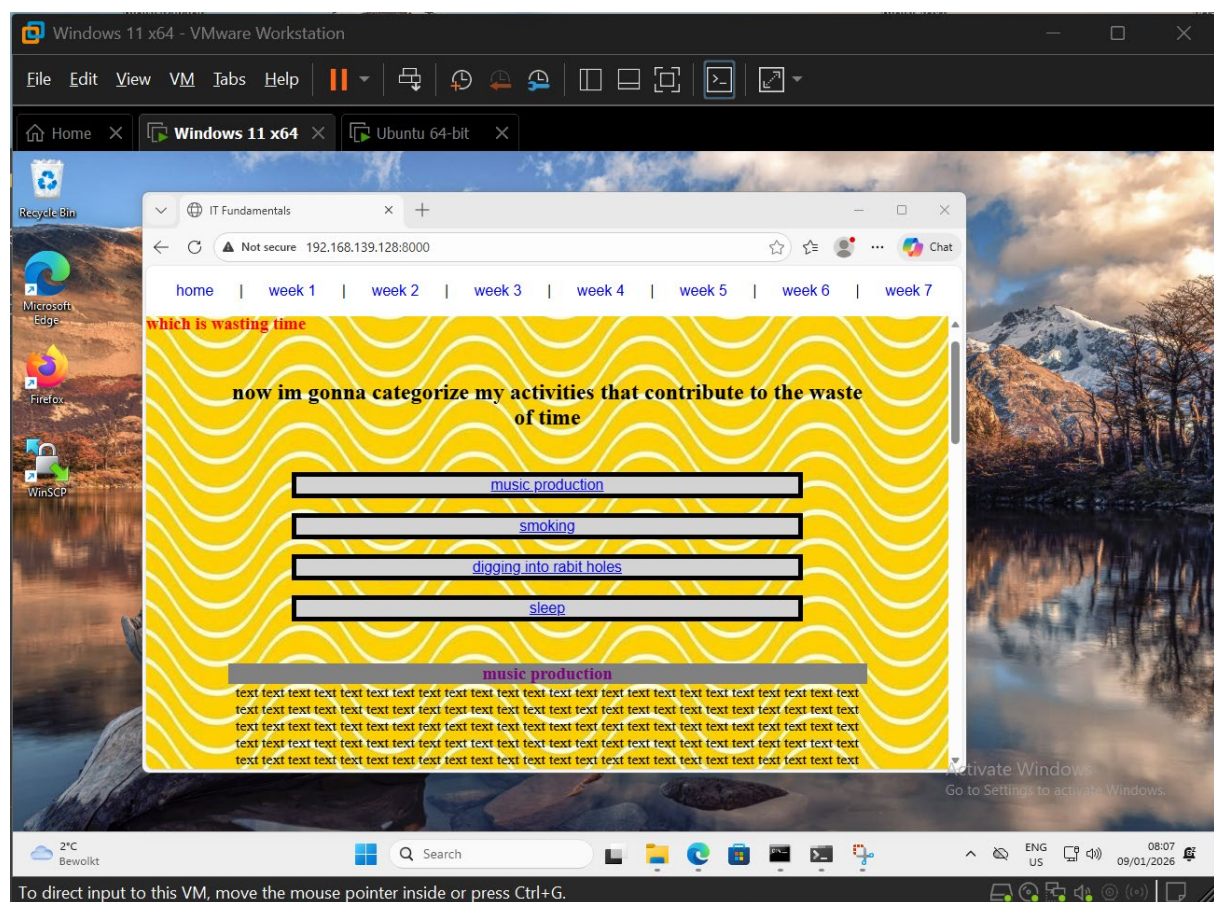
Screenshot of Site directory contents:



Screenshot python3 webserver command:



Screenshot web browser visits your site



Assignment 6.5: Network segment

Remember that bitwise java application you've made in week 2? Expand that application so that you can also calculate a network segment as explained in the PowerPoint slides of week 6. Use the bitwise & AND operator. You need to be able to input two Strings. An IP address and a subnet.

IP: 192.168.1.100 and subnet: 255.255.255.224 for /27

Example: 192.168.1.100/27

Calculate the network segment

IP Address: 11000000.10101000.00000001.01100100

Subnet Mask: 11111111.11111111.11111111.11100000

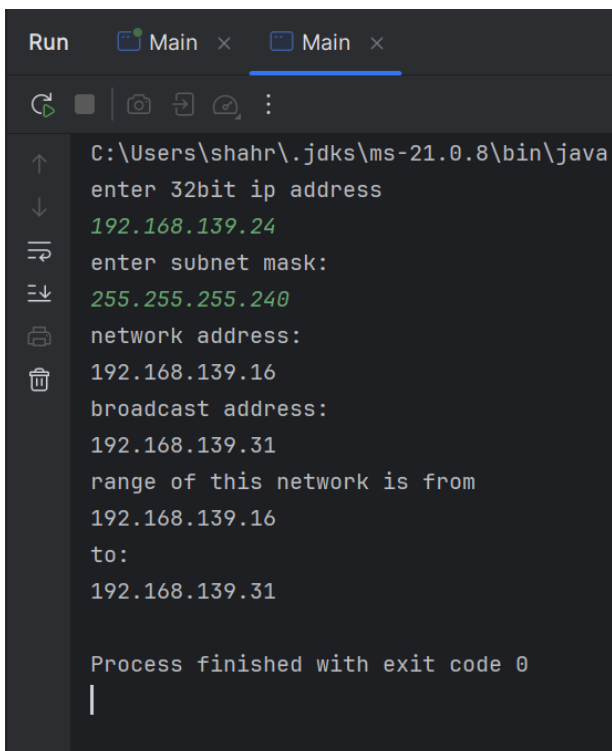
Network Addr: 11000000.10101000.00000001.01100000

This gives 192.168.1.96 in decimal as the network address.

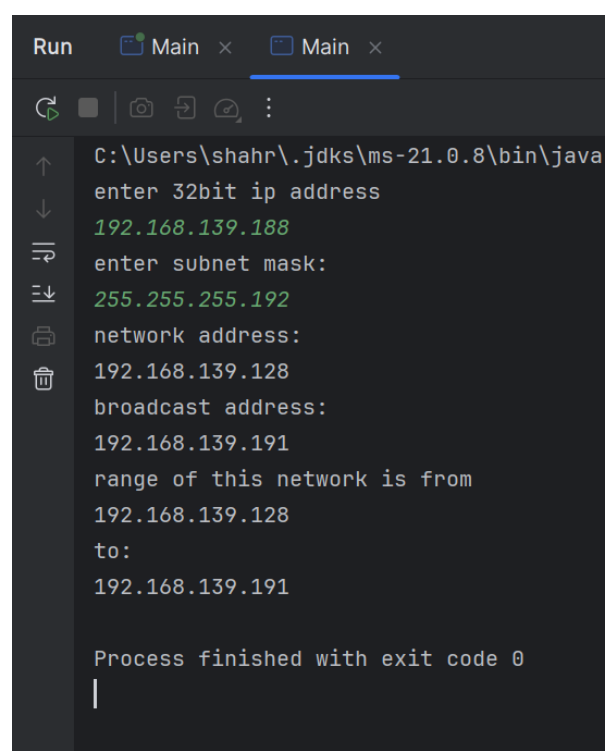
For a /27 subnet, each segment (or subnet) has 32 IP addresses (2^5).

The range of this network segment is from 192.168.1.96 to 192.168.1.127.

Paste source code here, with a screenshot of a working application.



```
Run Main x Main x
C:\Users\shahr\.jdk\ms-21.0.8\bin\java
enter 32bit ip address
192.168.139.24
enter subnet mask:
255.255.255.240
network address:
192.168.139.16
broadcast address:
192.168.139.31
range of this network is from
192.168.139.16
to:
192.168.139.31
Process finished with exit code 0
```



```
Run Main x Main x
C:\Users\shahr\.jdk\ms-21.0.8\bin\java
enter 32bit ip address
192.168.139.188
enter subnet mask:
255.255.255.192
network address:
192.168.139.128
broadcast address:
192.168.139.191
range of this network is from
192.168.139.128
to:
192.168.139.191
Process finished with exit code 0
```

```

import java.util.ArrayList;
import java.util.Scanner;
public class Main {
    static Scanner scanner = new Scanner(System.in);
    public static void main(String[] args){
        System.out.println("enter 32bit ip address");
        String ipString = scanner.nextLine().trim();
        int[] ip = addressToArray(ipString);
        System.out.println("enter subnet mask: ");
        String sMask = scanner.nextLine().trim();
        int[] subnetMask = addressToArray(sMask);
        int[] network = new int[4];
        int[] broadcast = new int[4];
        for(int i = 0; i<4; i++){
            network[i] = subnetMask[i] & ip[i];
            broadcast[i] = (~subnetMask[i] & 255) | network[i];
        }
        System.out.println("network address: ");
        System.out.println(arrayToAddress(network));
        System.out.println("broadcast address: ");
        System.out.println(arrayToAddress(broadcast));
        System.out.println("range of this network is from ");
        System.out.println(arrayToAddress(network));
        System.out.println("to: ");
        System.out.println(arrayToAddress(broadcast));
    }

    static int[] addressToArray(String addressString){
        int[] temp = new int[]{4,4,4,4};
        int j =0;
        int begin = 0;
        for (int i = 0; i < addressString.length(); i++) {
            if (addressString.charAt(i) == '.') {
                temp[j] =
Integer.parseInt(addressString.substring(begin, i));
                j++;
                begin = i+1;
            }
        }
        temp[j] = Integer.parseInt(addressString.substring(begin));
        return temp;
    }

    static String arrayToAddress(int[] address){
        String addressString = "";
        for(int i = 0; i < address.length; i++){
            addressString = addressString + address[i] + ".";
        }
        addressString = addressString.substring(0,
addressString.length()-1);
        return addressString;
    }
}

```

Ready? Save this file and export it as a pdf file with the name: [week6.pdf](#)