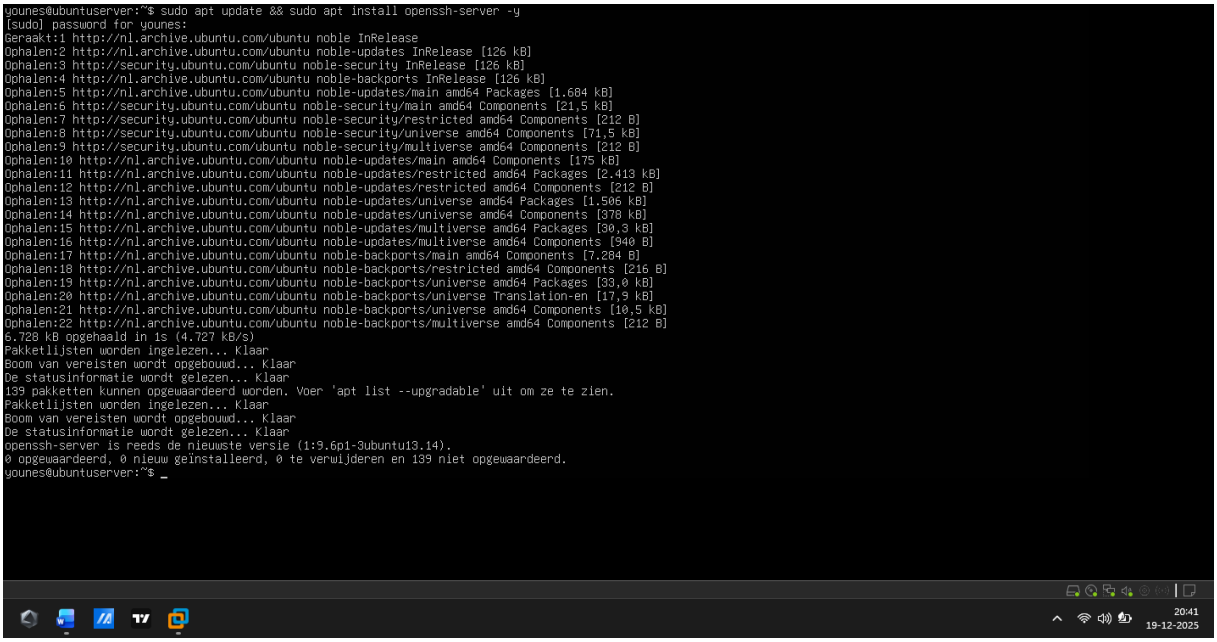
Template Week 6 – Networking

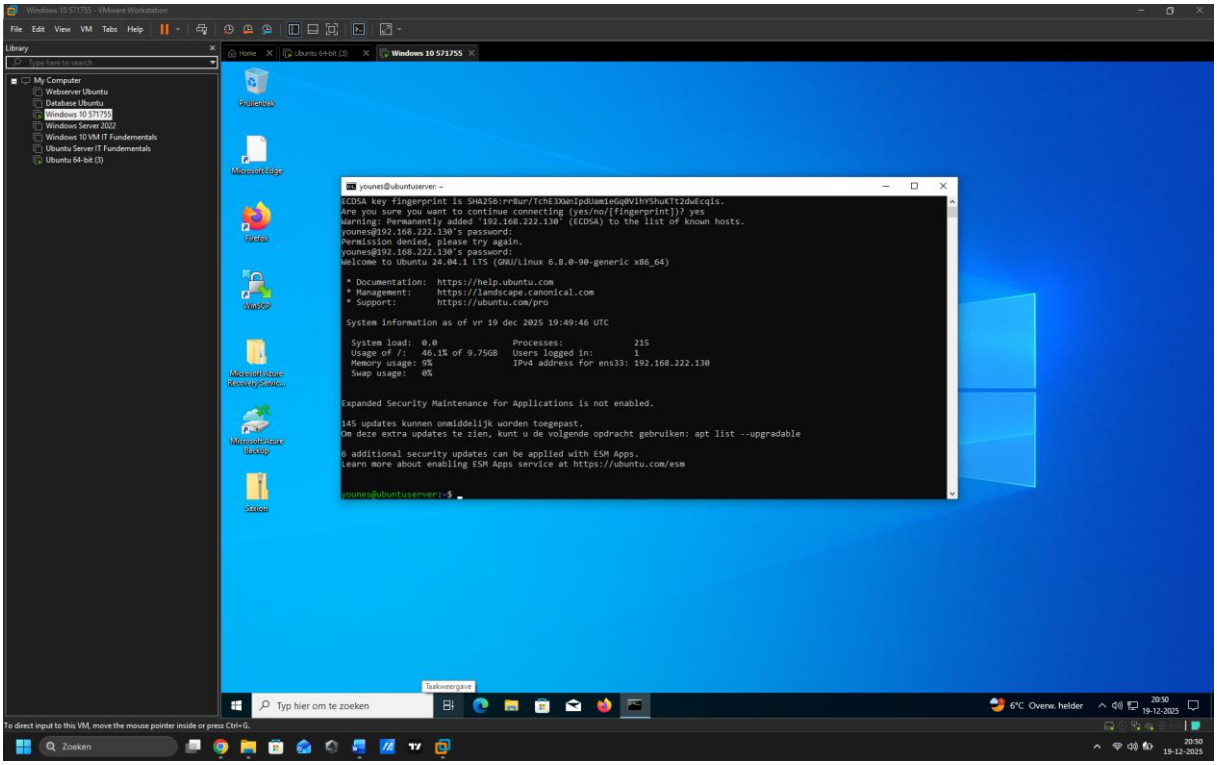
Student number: 571755

Assignment 6.1: Working from home

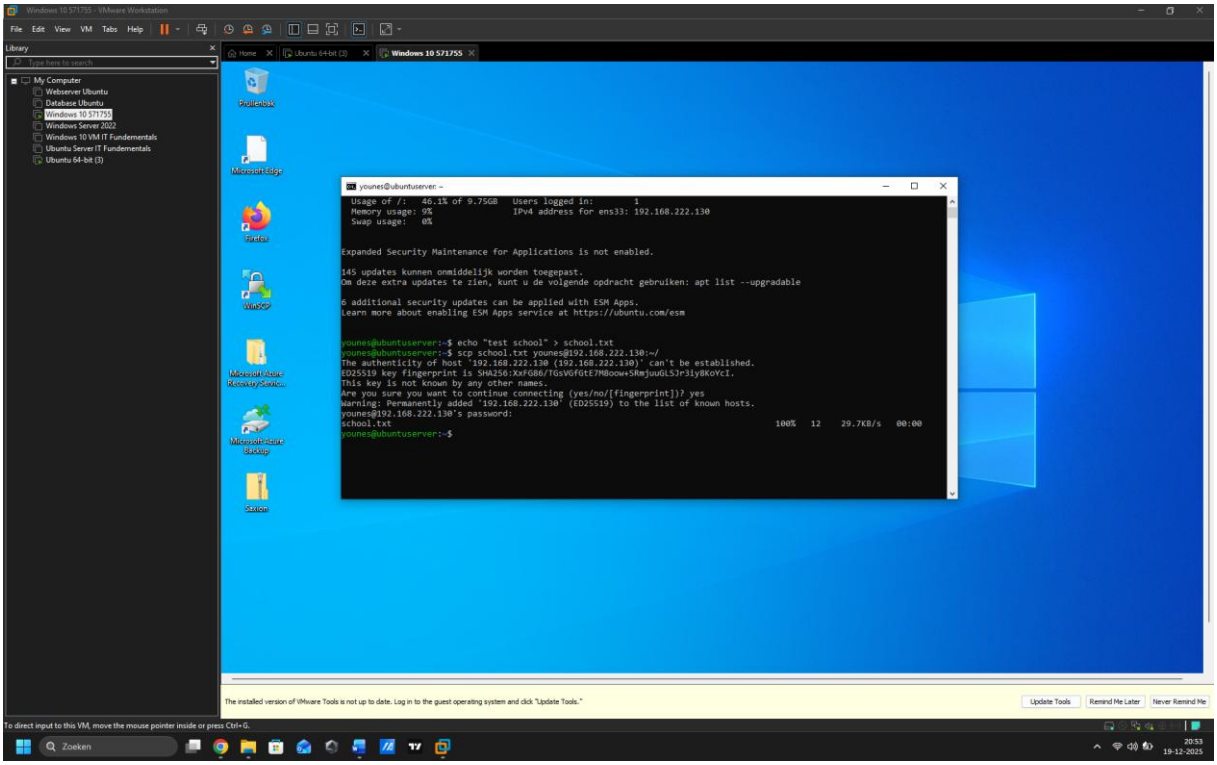
Screenshot installation openssh-server:



Screenshot successful SSH command execution:



Screenshot successful execution SCP command:



Screenshot remmina:

Deze kreeg ik niet aan de praat, ik weet precies hoe het moest, dit is wat ik had gedaan.

Windows VM -> Instellingen -> Systeem -> Extern bureaublad aan

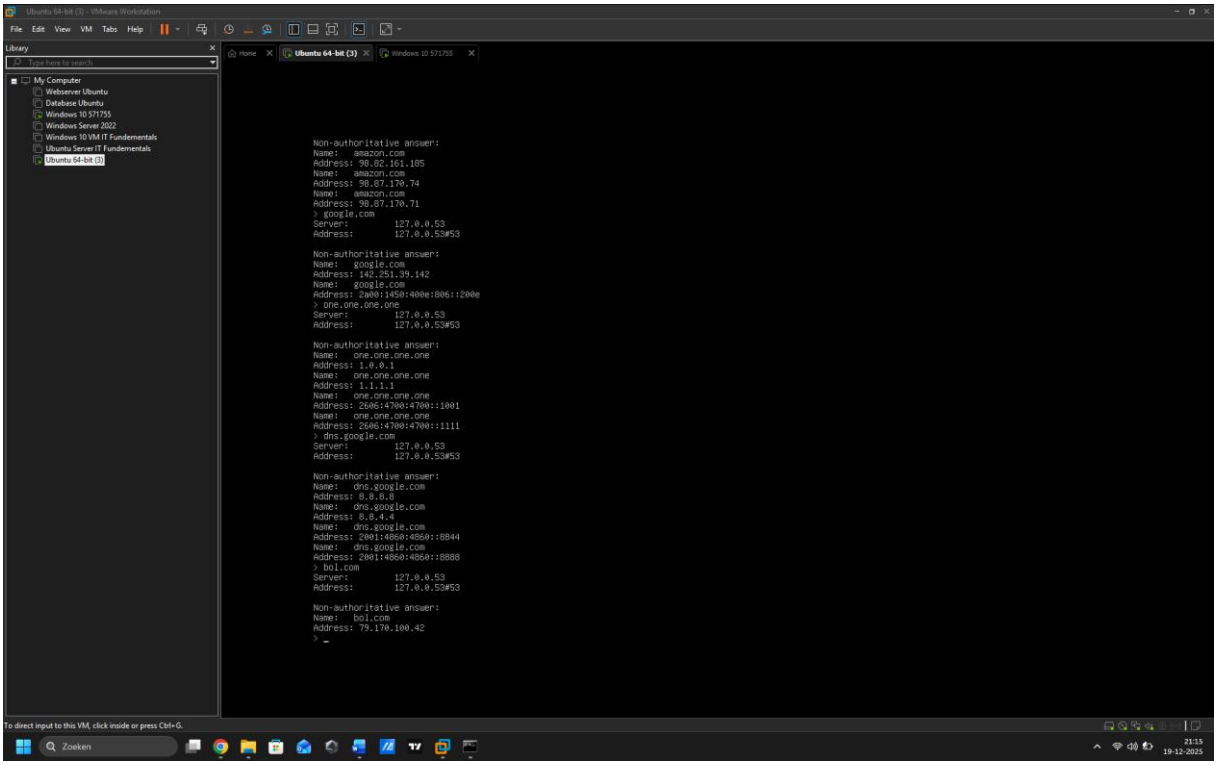
Ubuntu VM. **Remmina** openen

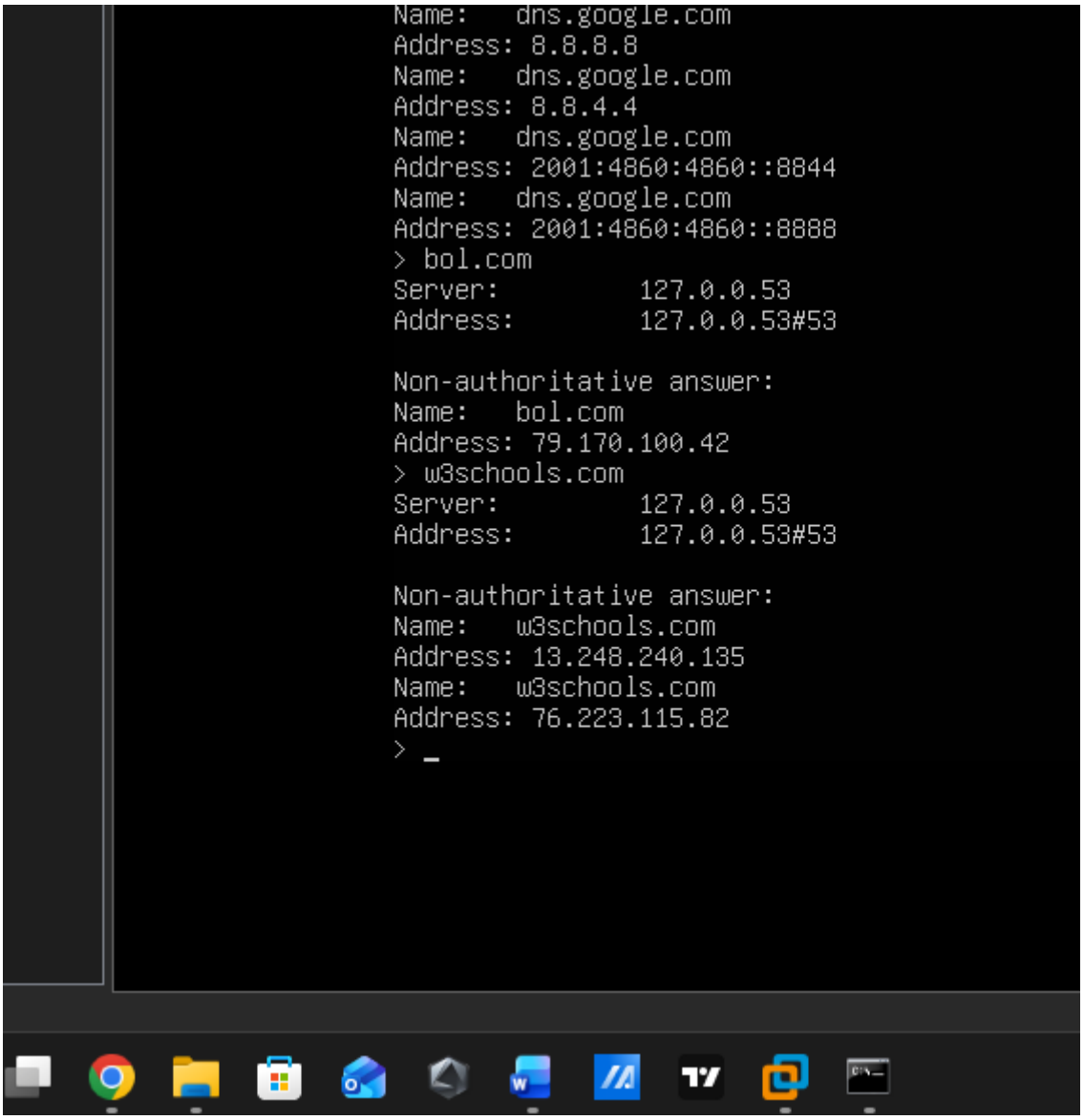
Windows VM in (zoek dit op Windows op met ipconfig) en klik op verbinden.

Via daar ben je dan connected, maar kreeg steeds error terwijl ik alle ip adres goed had

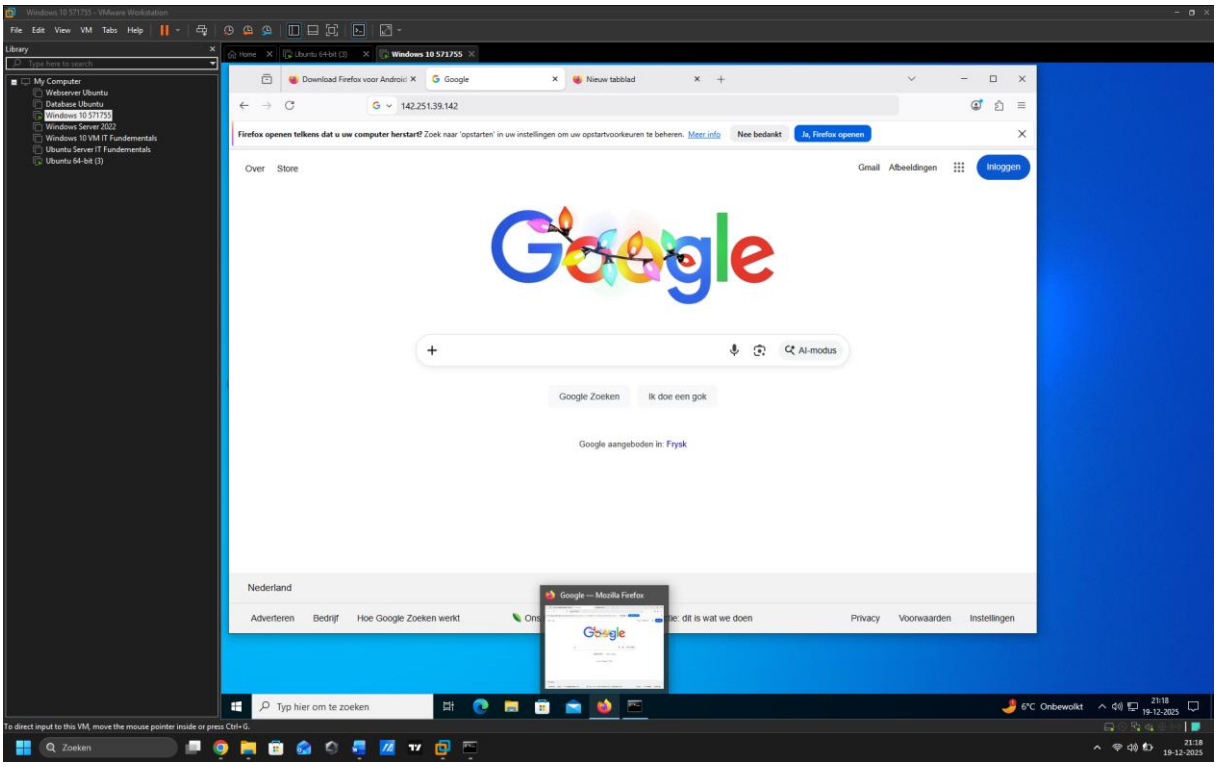
Assignment 6.2: IP addresses websites

Relevant screenshots nslookup command:




Name: dns.google.com
Address: 8.8.8.8
Name: dns.google.com
Address: 8.8.4.4
Name: dns.google.com
Address: 2001:4860:4860::8844
Name: dns.google.com
Address: 2001:4860:4860::8888
> bol.com
Server: 127.0.0.53
Address: 127.0.0.53#53
Non-authoritative answer:
Name: bol.com
Address: 79.170.100.42
> w3schools.com
Server: 127.0.0.53
Address: 127.0.0.53#53
Non-authoritative answer:
Name: w3schools.com
Address: 13.248.240.135
Name: w3schools.com
Address: 76.223.115.82
> _

Screenshot website visit via IP address:



Assignment 6.3: subnetting

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

Een /25 subnet betekent dat er 128 adressen in het netwerk zitten, want

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

Het bereik is **192.168.110.129 t/m 192.168.110.254**

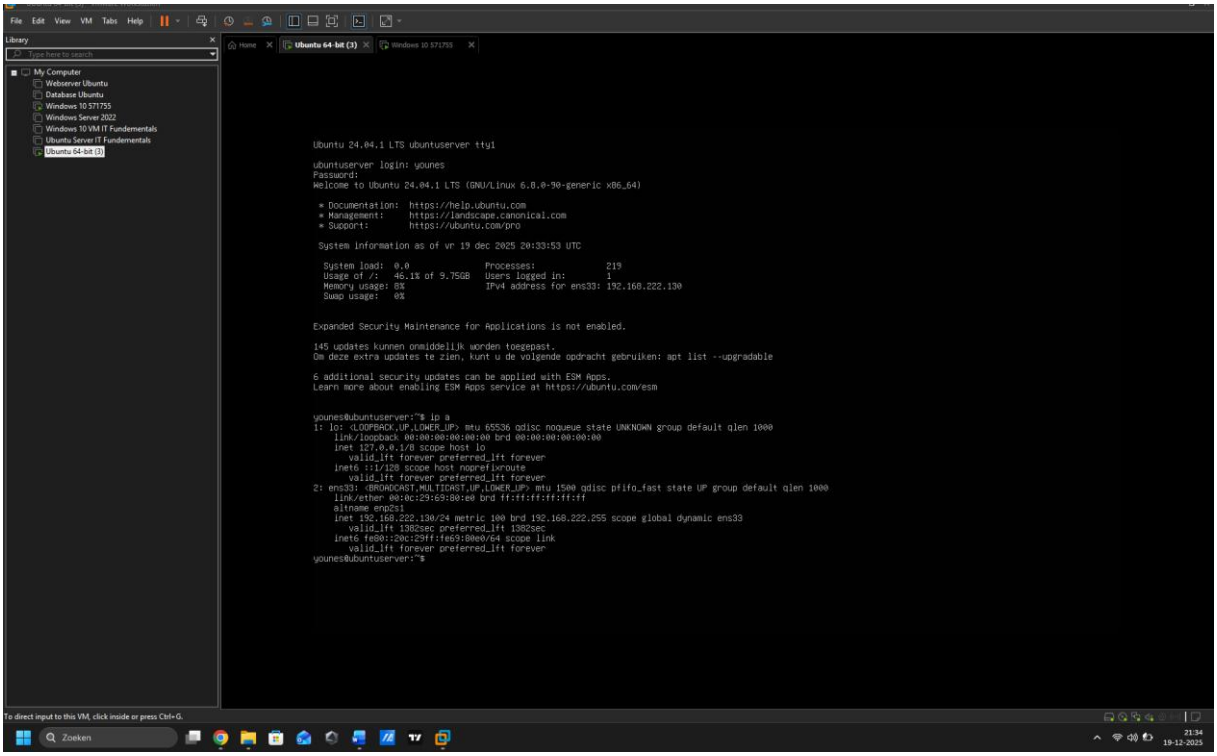
Check your two previous answers with this Linux command: ipcalc 192.168.110.128/25

Explain the above calculation in your own words.

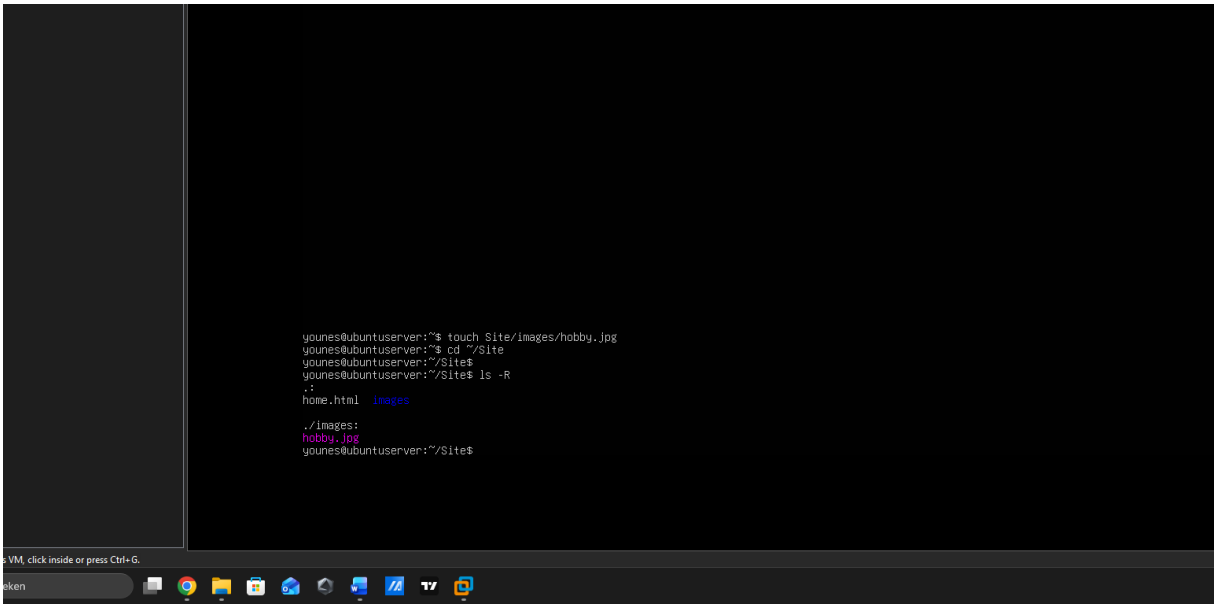
Het getal /25 geeft het subnetmasker aan. Dit betekent dat de eerste 25 bits van het 32-bits IP-adres gereserveerd zijn voor het netwerkgedeelte, waardoor er 7 bits overblijven voor de hosts ($2^7 = 128$ adressen) De laatste en de eerste zijn al gereserveerd voor de broadcast en netwerk zelf. Dus blijven er 126 over.

Assignment 6.4: HTML

Screenshot IP address Ubuntu VM:



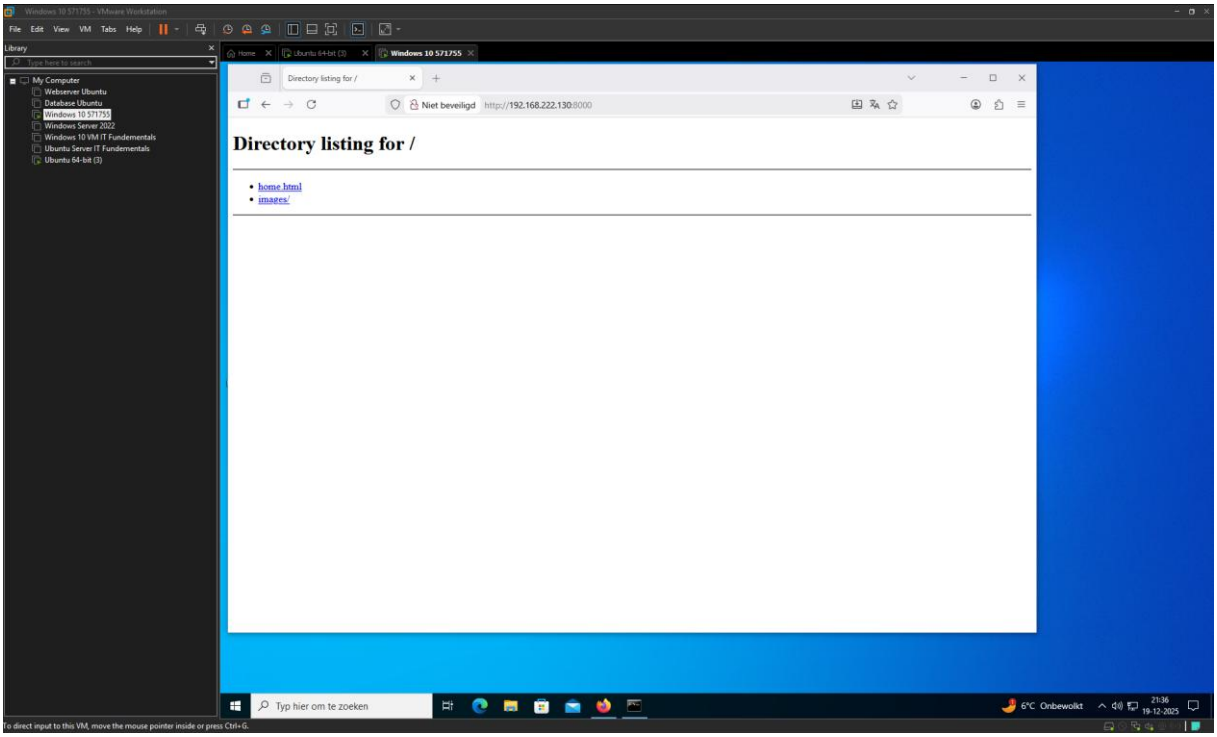
Screenshot of Site directory contents:



Screenshot python3 webserver command:

```
  System load:  0.0               Processes:             219
  Usage of /:   46.1% of 9.75GB   Users logged in:       1
  Memory usage: 8%                IPv4 address for ens33: 192.168.222.130
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

145 updates kunnen onmiddelijk worden toegepast.
Om deze extra updates te zien, kunt u de volgende opdracht gebruiken: apt list --upgradable

6 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm


younes@ubuntuserver:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    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 pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:69:80:e0 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.222.130/24 metric 100 brd 192.168.222.255 scope global dynamic ens33
       valid_lft 1382sec preferred_lft 1382sec
    inet6 fe80::20c:29ff:fe69:80e0/64 scope link
       valid_lft forever preferred_lft forever
younes@ubuntuserver:~$ cd ~/Site
younes@ubuntuserver:~/Site$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

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 (2⁵).
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.

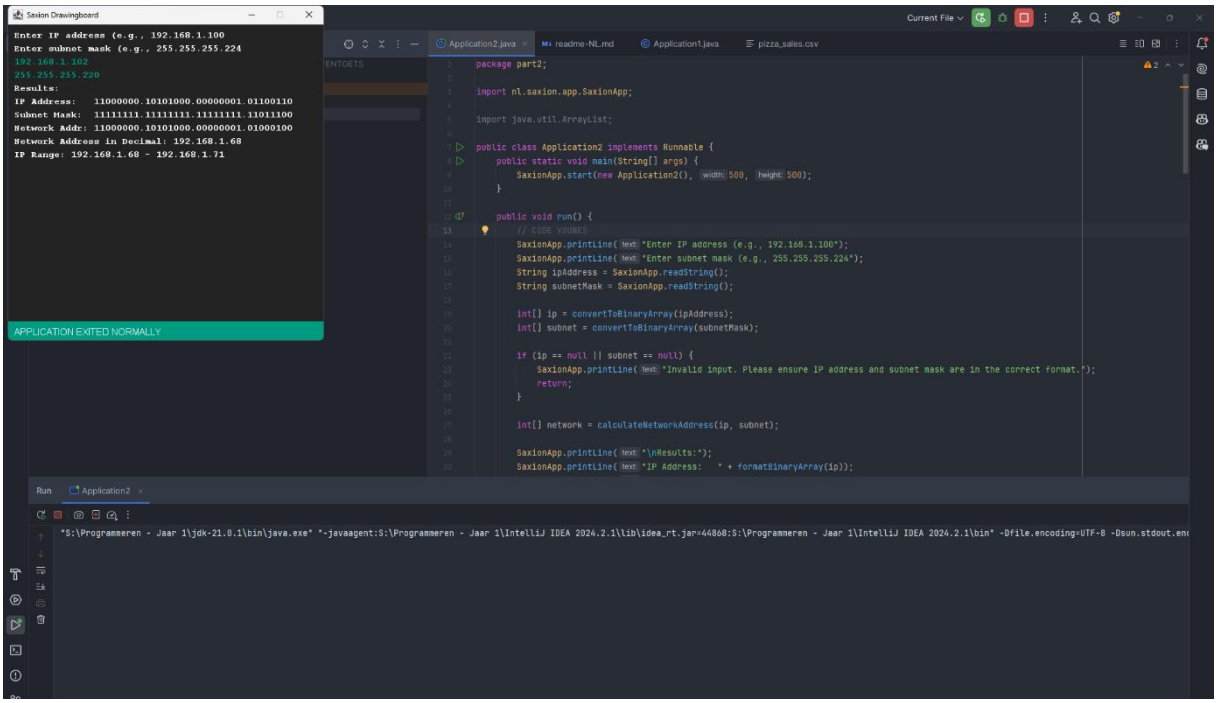
DE CODE :

```
package part2;

import nl.saxion.app.SaxionApp;

import java.util.ArrayList;

public class Application2 implements Runnable {
  public static void main(String[] args) {
    SaxionApp.start(new Application2(), 500, 500);
  }

  public void run() {
    // CODE YOUNES
    SaxionApp.printLine("Enter IP address (e.g., 192.168.1.100");
    SaxionApp.printLine("Enter subnet mask (e.g., 255.255.255.224");
    String ipAddress = SaxionApp.readString();
    String subnetMask = SaxionApp.readString();

    int[] ip = convertToBinaryArray(ipAddress);
    int[] subnet = convertToBinaryArray(subnetMask);

    if (ip == null || subnet == null) {
      SaxionApp.printLine("Invalid input. Please ensure IP address and subnet mask are in the
correct format.");
```

```
            return;
        }

        int[] network = calculateNetworkAddress(ip, subnet);

        SaxionApp.printLine("\nResults:");
        SaxionApp.printLine("IP Address:   " + formatBinaryArray(ip));
        SaxionApp.printLine("Subnet Mask:  " + formatBinaryArray(subnet));
        SaxionApp.printLine("Network Addr: " + formatBinaryArray(network));

        String networkAddressDecimal = convertToDecimal(network);
        SaxionApp.printLine("Network Address in Decimal: " + networkAddressDecimal);

        calculateAndDisplayRange(network, subnet);
    }

    private static int[] convertToBinaryArray(String dottedDecimal) {
        String[] parts = dottedDecimal.split("\\.");
        if (parts.length != 4) return null;

        int[] binaryArray = new int[32];
        for (int i = 0; i < 4; i++) {
            int octet;
            try {
                octet = Integer.parseInt(parts[i]);
            } catch (NumberFormatException e) {
                return null;
            }

            if (octet < 0 || octet > 255) return null;

            for (int j = 7; j >= 0; j--) {
                binaryArray[i * 8 + j] = (octet & 1);
                octet >>= 1;
            }
        }
        return binaryArray;
    }

    private static int[] calculateNetworkAddress(int[] ip, int[] subnet) {
        int[] network = new int[32];
        for (int i = 0; i < 32; i++) {
            network[i] = ip[i] & subnet[i];
        }
        return network;
    }

    private static String formatBinaryArray(int[] binaryArray) {
```

```
        StringBuilder formatted = new StringBuilder();
        for (int i = 0; i < binaryArray.length; i++) {
            formatted.append(binaryArray[i]);
            if ((i + 1) % 8 == 0 && i != binaryArray.length - 1) {
                formatted.append(".");
            }
        }
        return formatted.toString();
    }

    private static String convertToDecimal(int[] binaryArray) {
        StringBuilder decimal = new StringBuilder();
        for (int i = 0; i < 4; i++) {
            int value = 0;
            for (int j = 0; j < 8; j++) {
                value = (value << 1) | binaryArray[i * 8 + j];
            }
            decimal.append(value);
            if (i != 3) {
                decimal.append(".");
            }
        }
        return decimal.toString();
    }

    private static void calculateAndDisplayRange(int[] network, int[] subnet) {
        int hostBits = 0;
        for (int bit : subnet) {
            if (bit == 0) hostBits++;
        }

        int totalHosts = (int) Math.pow(2, hostBits);

        int[] broadcast = network.clone();
        for (int i = 31; i >= 32 - hostBits; i--) {
            broadcast[i] = 1;
        }

        String networkAddress = convertToDecimal(network);
        String broadcastAddress = convertToDecimal(broadcast);

        SaxionApp.printLine("IP Range: " + networkAddress + " - " + broadcastAddress);
    }
}
```

SCREENSHOT DAT HET WERKT :



Ready? Save this file and export it as a pdf file with the name: **week6.pdf**