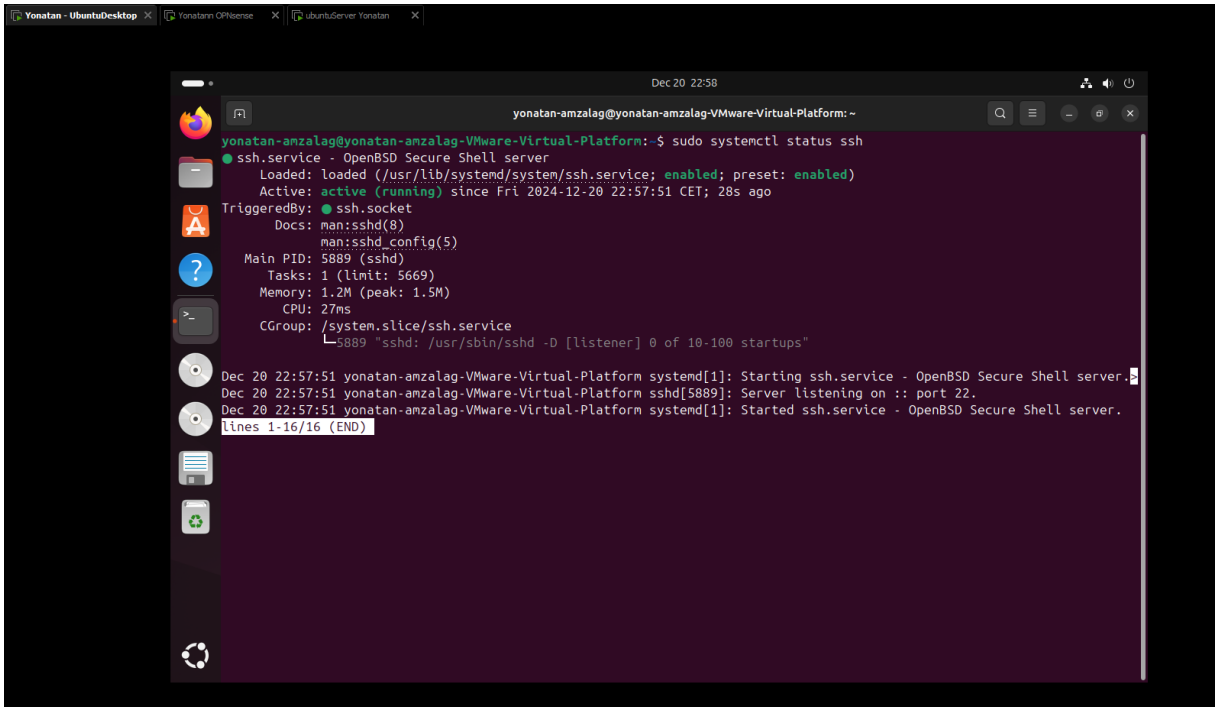
Template Week 6 – Networking

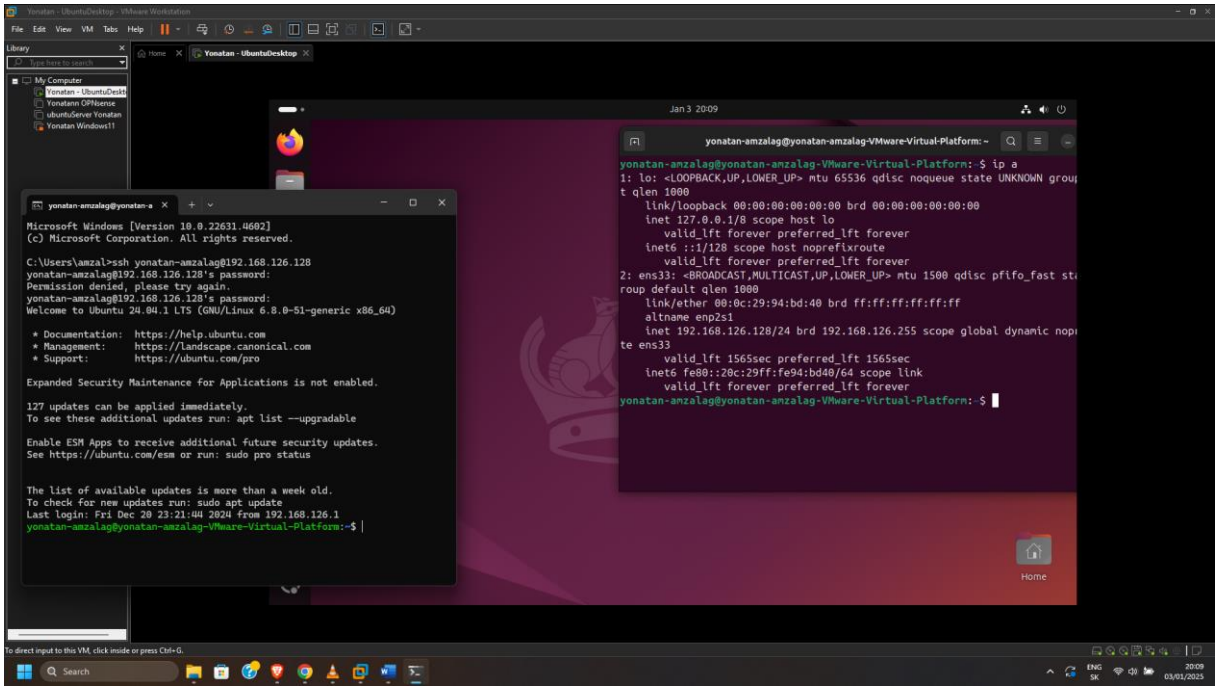
Student number: 563634

Assignment 6.1: Working from home

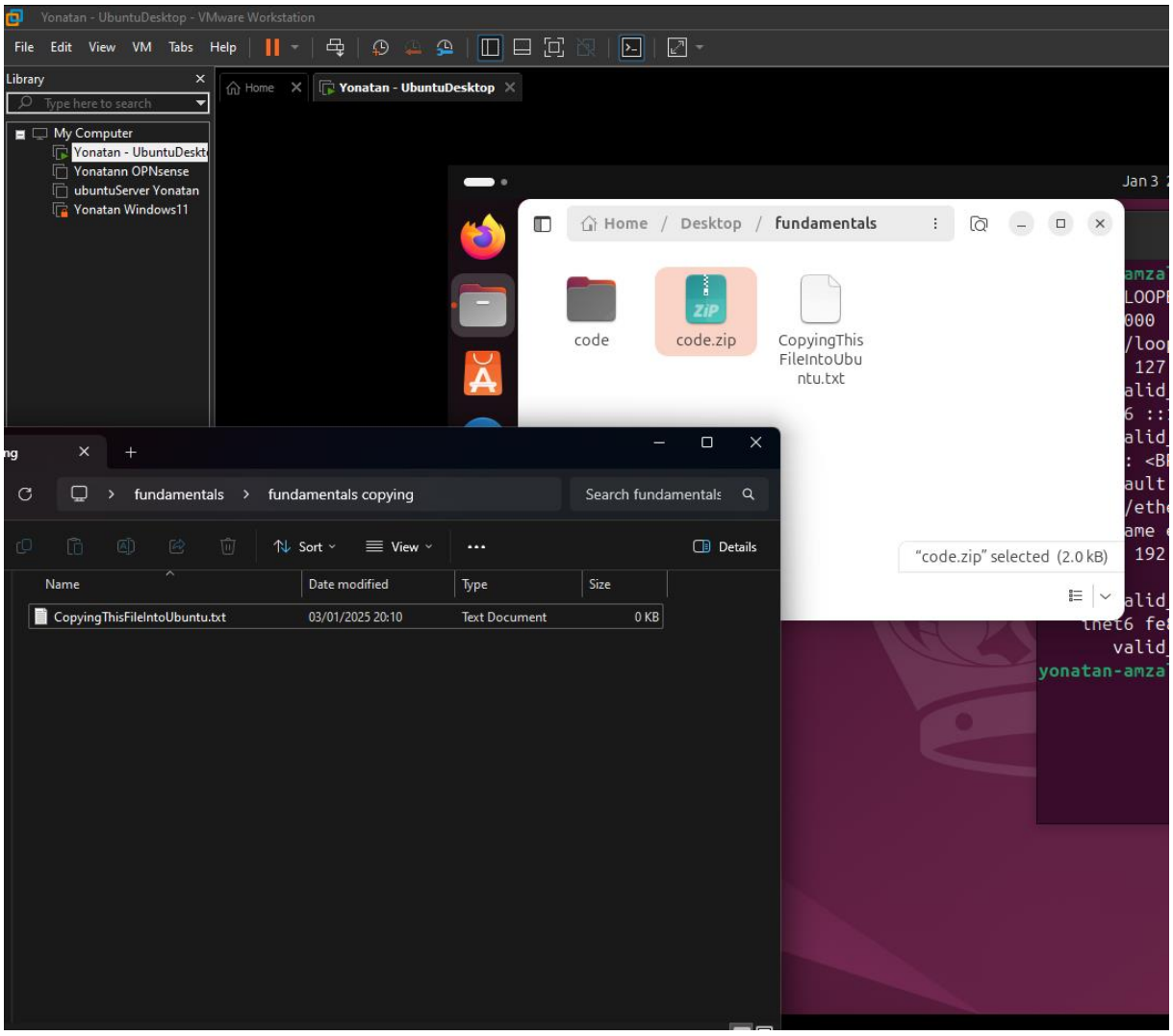
Screenshot installation openssh-server:

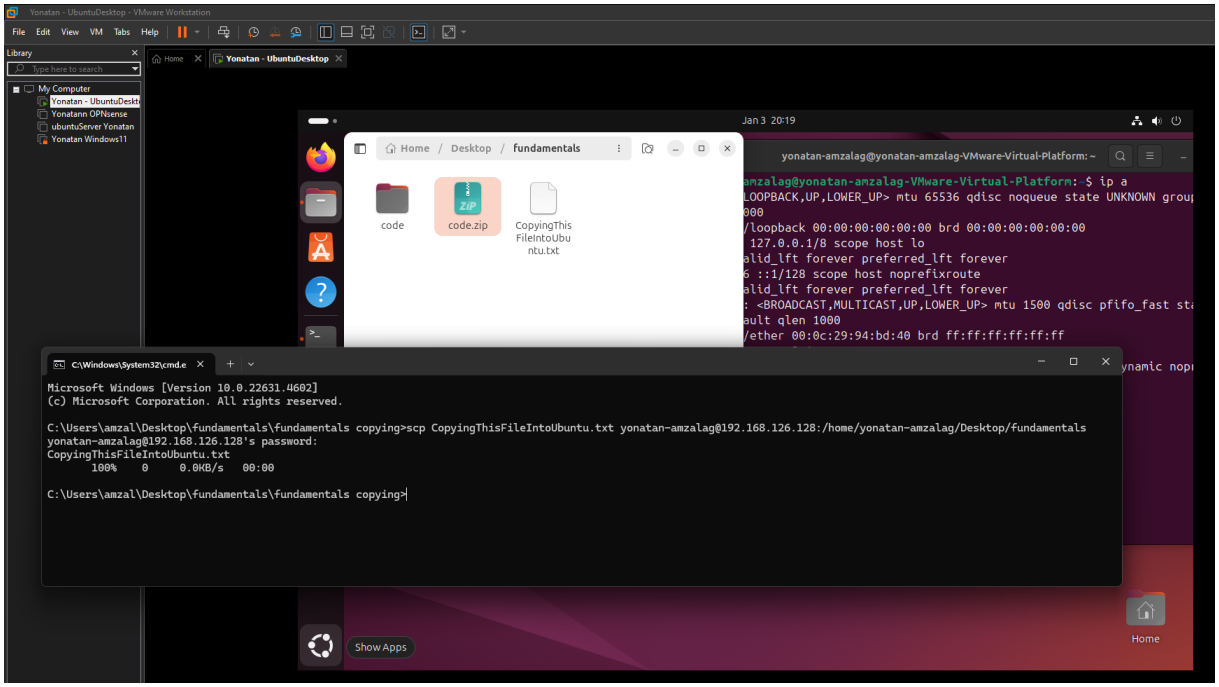


Screenshot successful SSH command execution:

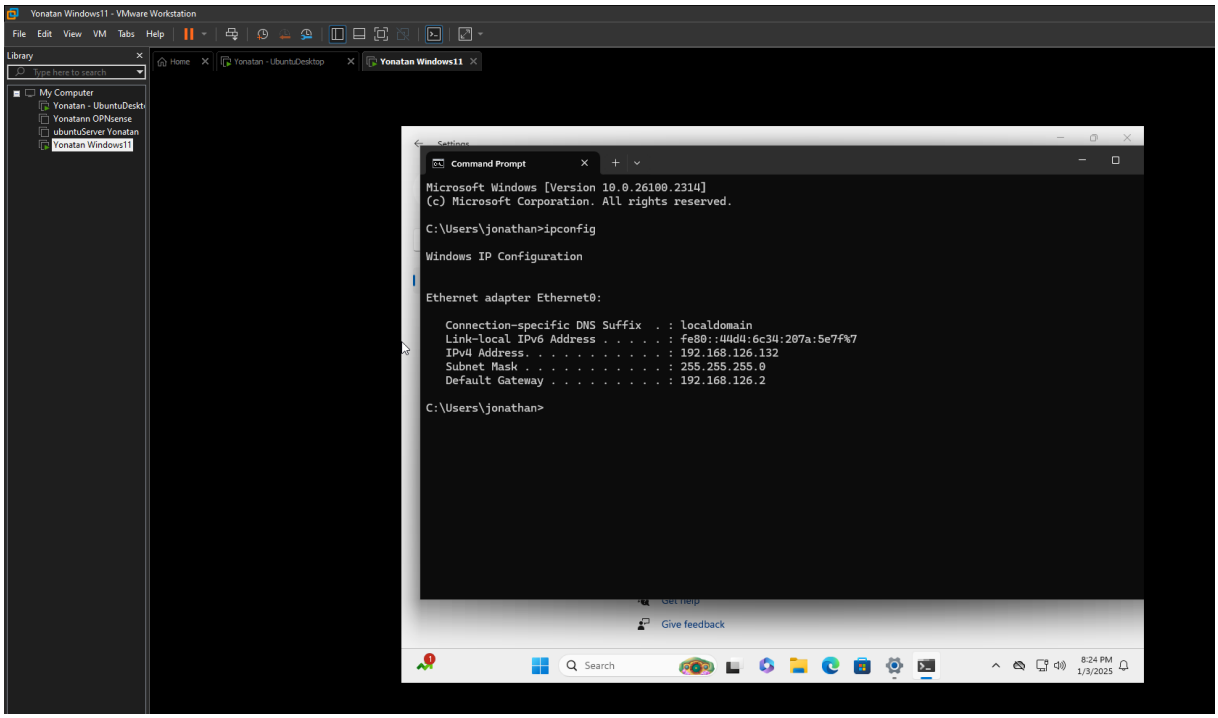


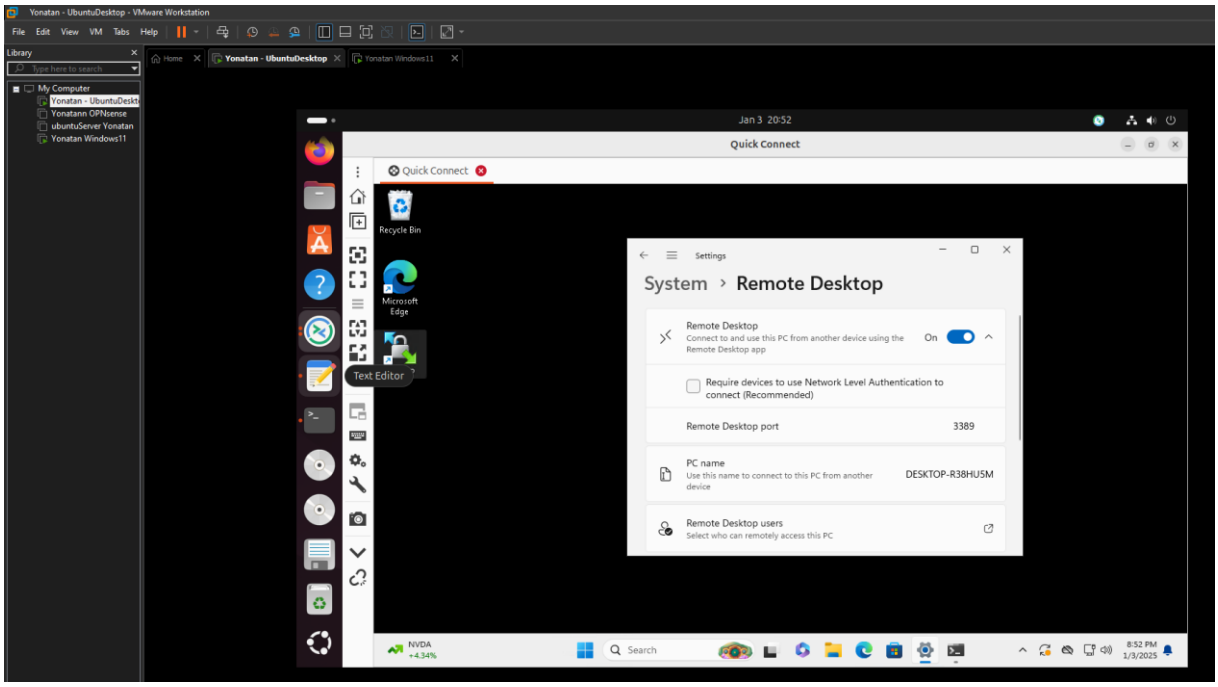
Screenshot successful execution SCP command:

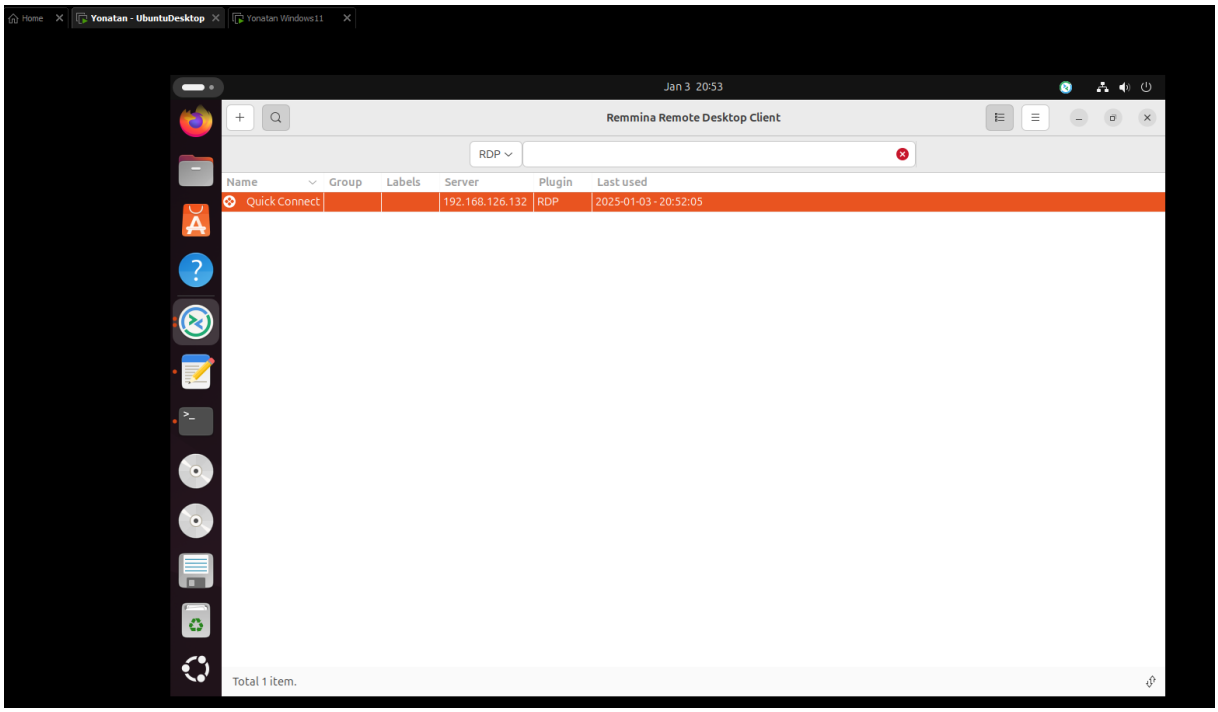




Screenshot remmina:

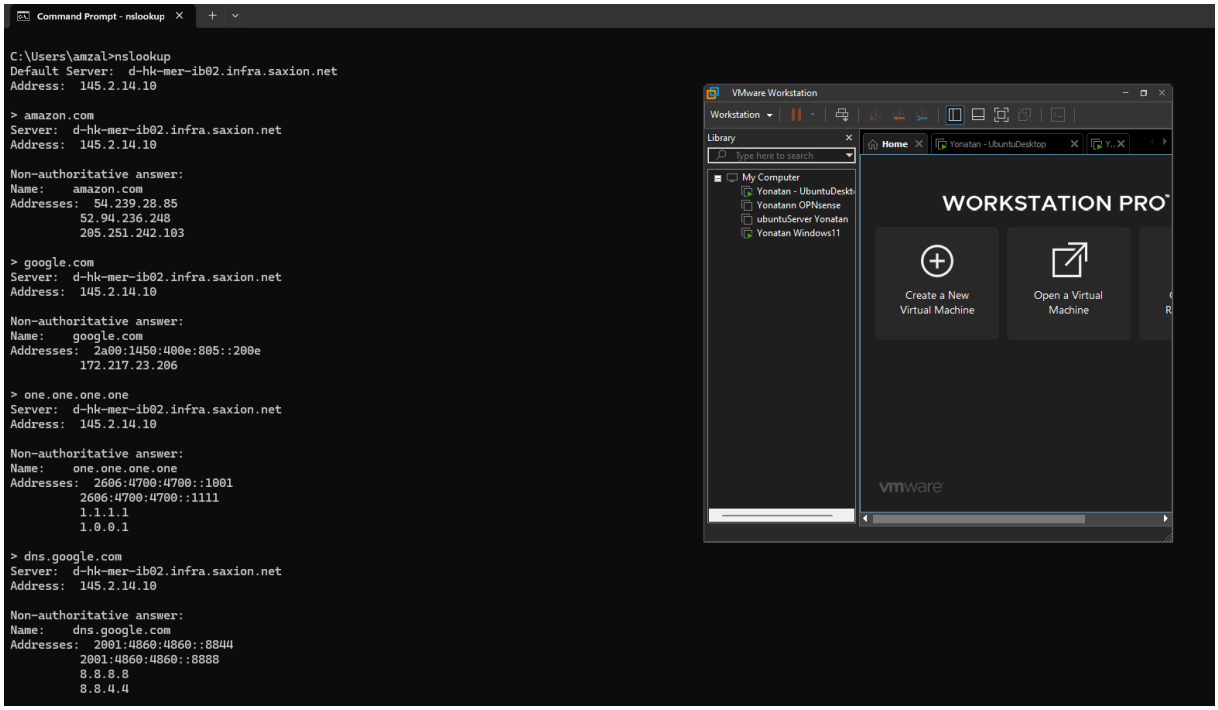


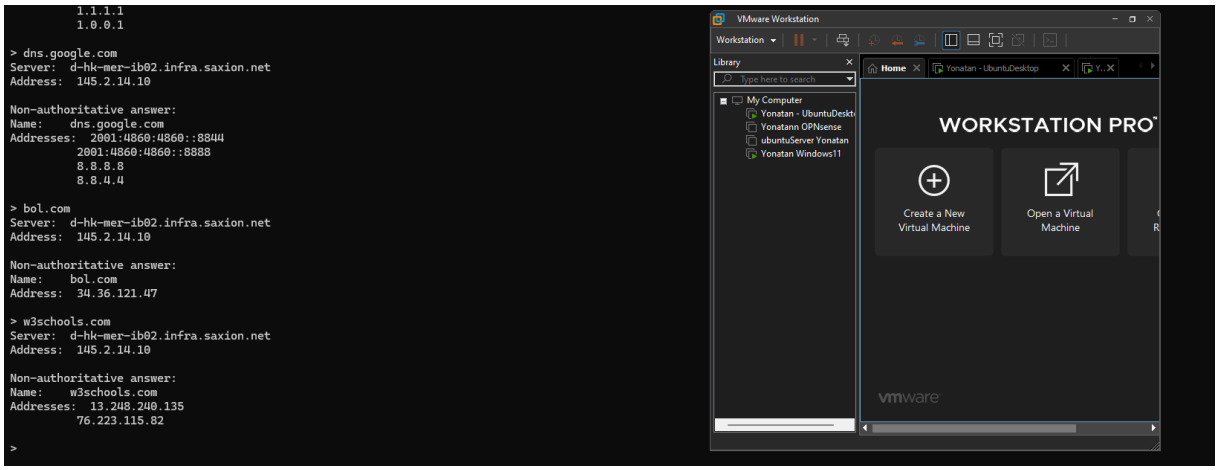




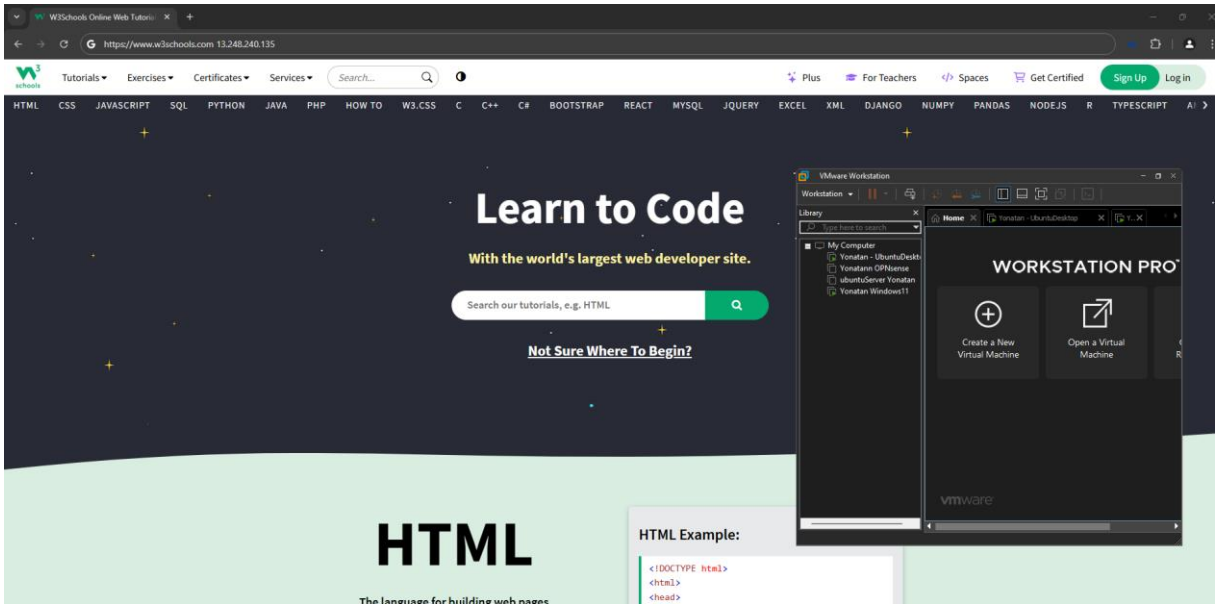
Assignment 6.2: IP addresses websites

Relevant screenshots nslookup command:





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 IP addresses.

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

We cannot use the first and last ip address, so the answer is between

Check your two previous answers with this calculator:
https://www.calculator.net/ip-subnet-calculator.html

home / other / ip subnet calculator

IP Subnet Calculator

This calculator returns a variety of information regarding Internet Protocol version 4 (IPv4) and IPv6 subnets including possible network addresses, usable host ranges, subnet mask, and IP class, among others.

IPv4 Subnet Calculator

Result

IP Address:	192.168.110.128
Network Address:	192.168.110.128
Usable Host IP Range:	192.168.110.129 - 192.168.110.254
Broadcast Address:	192.168.110.255
Total Number of Hosts:	128
Number of Usable Hosts:	126
Subnet Mask:	255.255.255.128
Wildcard Mask:	0.0.0.127
Binary Subnet Mask:	11111111.11111111.11111111.10000000
IP Class:	C
CIDR Notation:	/25
IP Type:	Private
Short:	192.168.110.128 /25
Binary ID:	11000000101010000110111010000000
Integer ID:	3232263808
Hex ID:	0xc0a86e80
in-addr.arpa:	128.110.168.192.in-addr.arpa
IPv4 Mapped Address:	::ffff:c0a8.6e80
6to4 Prefix:	2002:c0a8.6e80::/48

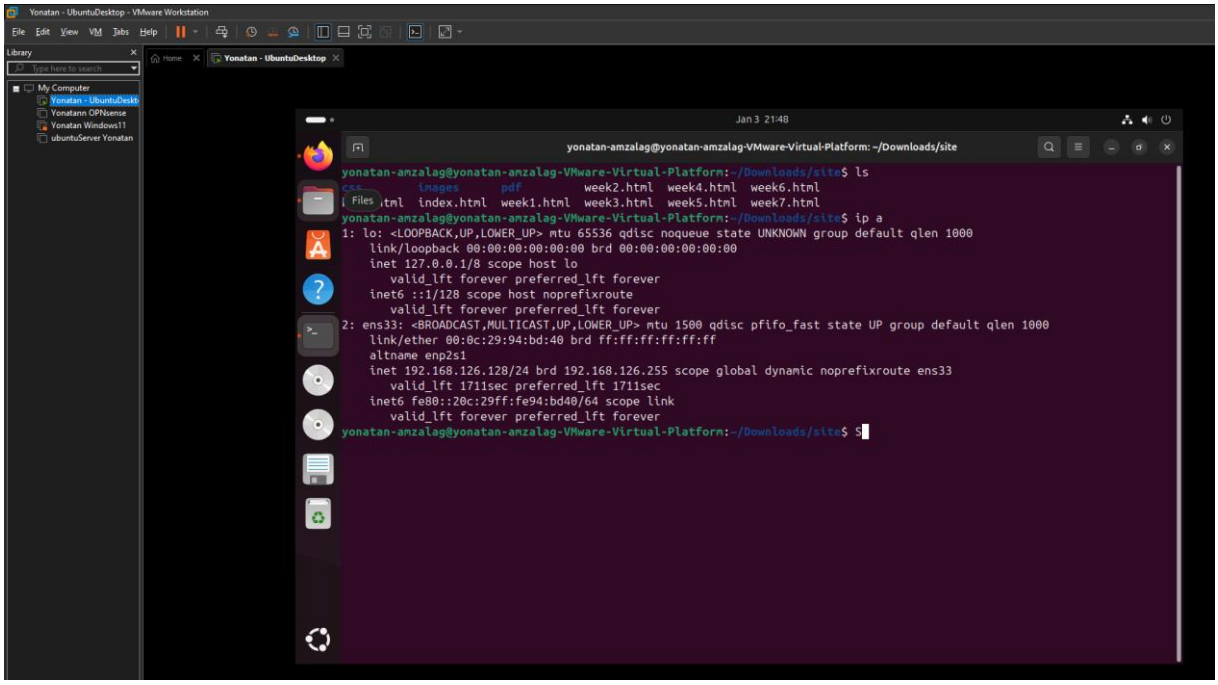
All 2 of the Possible /25 Networks for 192.168.110.*

Network Address	Usable Host Range	Broadcast Address:
192.168.110.0	192.168.110.1 - 192.168.110.126	192.168.110.127
192.168.110.128	192.168.110.129 - 192.168.110.254	192.168.110.255

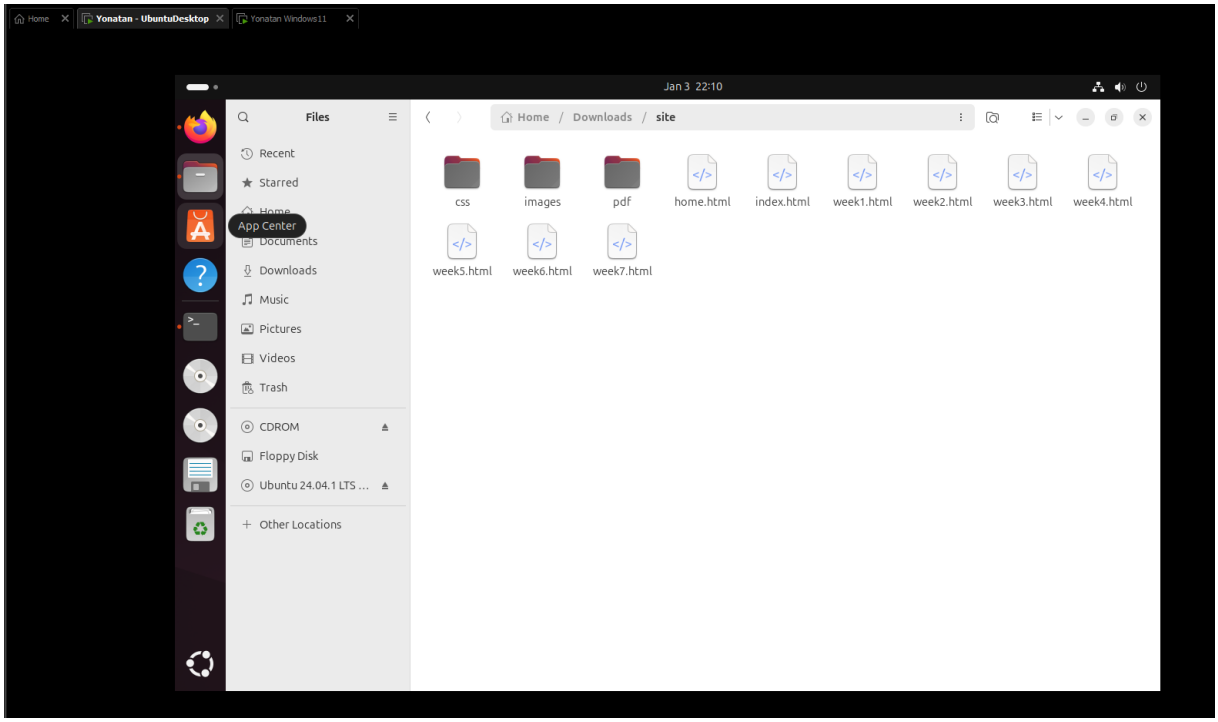
Explain the above calculation in your own words.

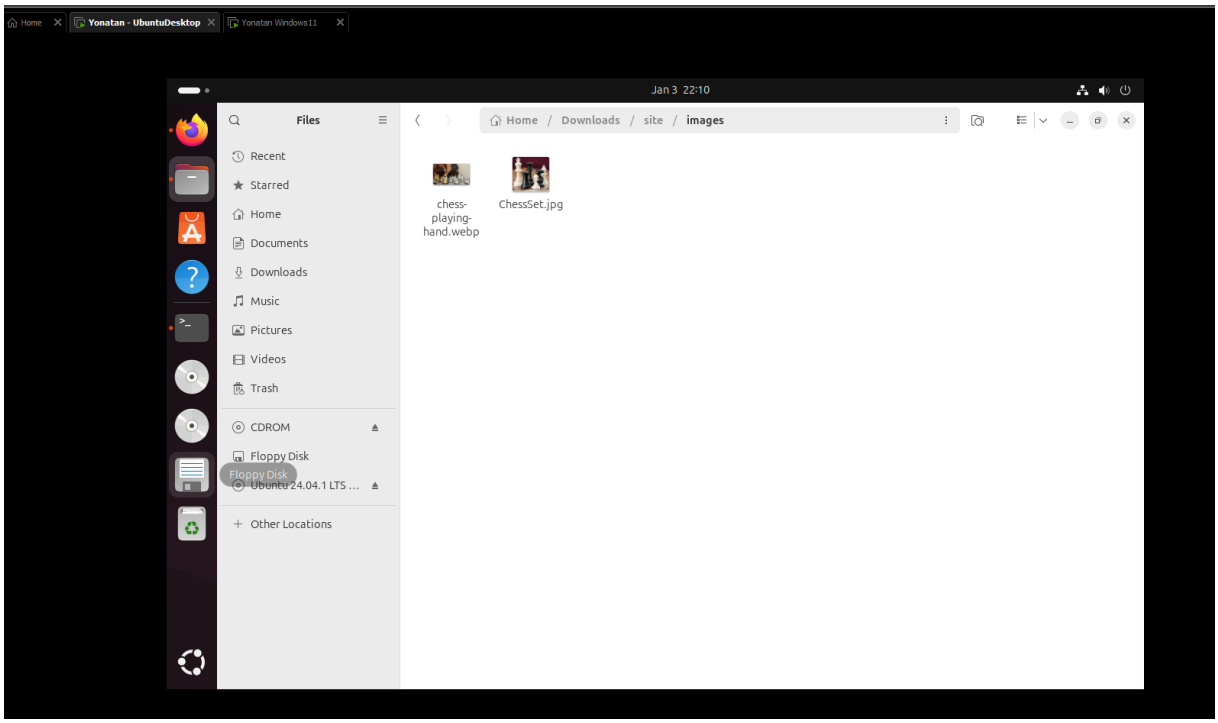
Assignment 6.4: HTML

Screenshot IP address Ubuntu VM:

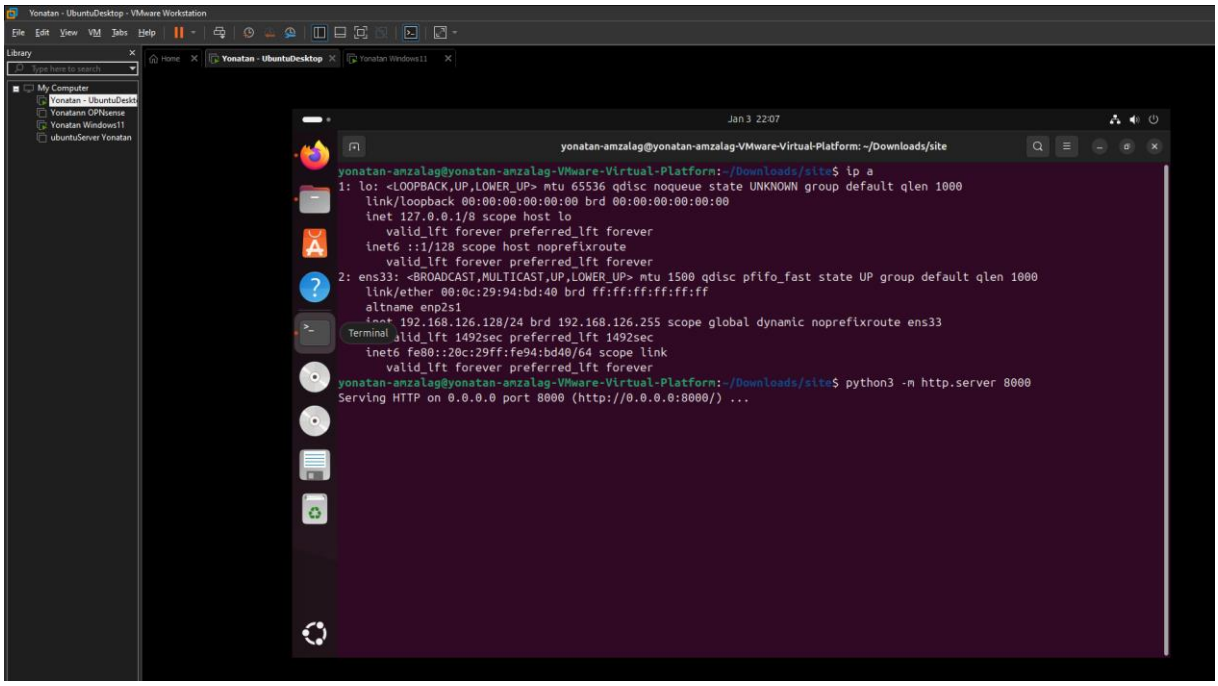


Screenshot of Site directory contents:

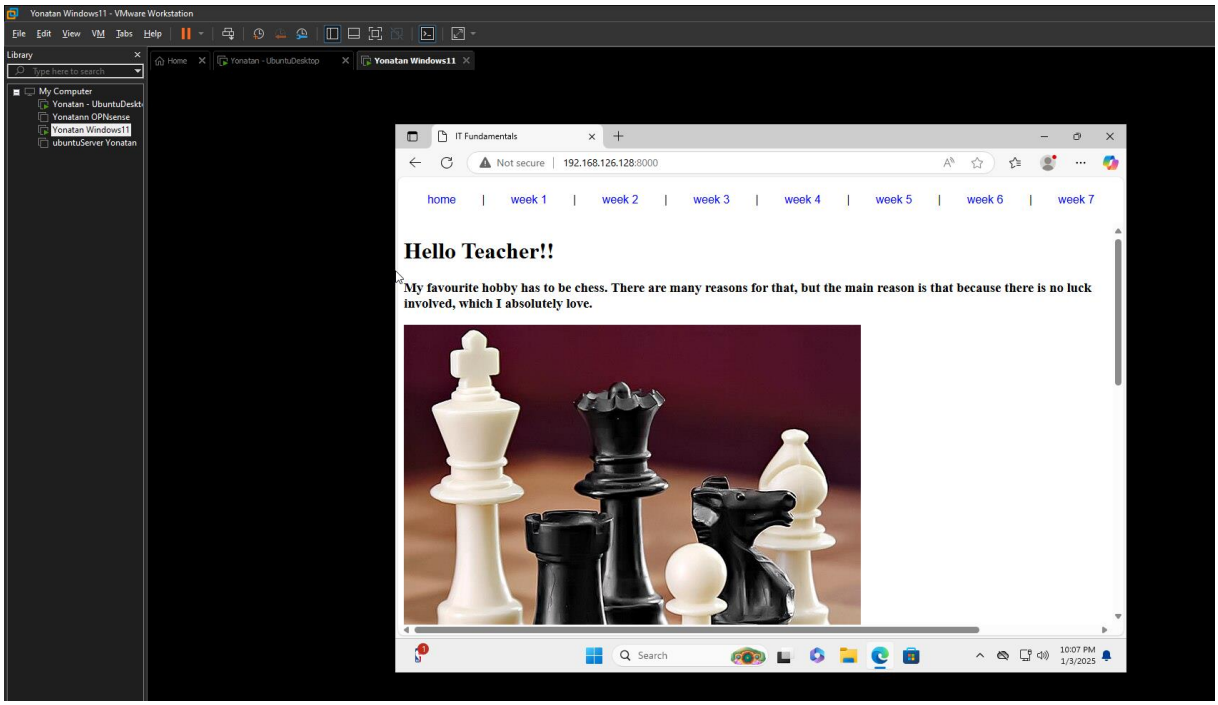




Screenshot python3 webserver command:



Screenshot web browser visits your site



Bonus point assignment – week 6

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.


```
import nl.saxion.app.SaxionApp;

import java.awt.*;

public class Application implements Runnable {

    public static void main(String[] args) {
        SaxionApp.start(new Application());
    }

    public void run() {

        while (true) {
            SaxionApp.printLine("Choose an option:");
```

```
        SaxionApp.printLine("1. Is number odd?");
        SaxionApp.printLine("2. Is number a power of 2?");
        SaxionApp.printLine("3. Two's complement of number?");
        SaxionApp.printLine("4. Calculate network segment");
        SaxionApp.printLine("5. Exit");
        SaxionApp.printLine("Student number: 563634, Yonatan Amzalag");

        int choice = SaxionApp.readInt();

        if (choice == 5) {
            SaxionApp.printLine("BYE BYE!");
            break;
        }


        if (choice == 1 || choice == 2 || choice == 3) {
            SaxionApp.print("Enter a number: ");
            int number = SaxionApp.readInt();


            if (choice == 1) {
                if (isOdd(number)) {
                    SaxionApp.printLine("Number " + number + " is odd.", Color.green);
                } else {
                    SaxionApp.printLine("Number " + number + " is even.", Color.green);
                }
            } else if (choice == 2) {
                if (powerOfTwo(number)) {
                    SaxionApp.printLine("Number " + number + " is a power of 2.", Color.green);
                } else {
                    SaxionApp.printLine("Number " + number + " is not a power of 2.", Color.green);
                }
            } else if (choice == 3) {
                int complement = twosComplement(number);
                SaxionApp.printLine("Two's complement of " + number + " is " + complement,
Color.green);
            }
        } else if (choice == 4) {
            SaxionApp.printLine("Enter IP address");
            String ip = SaxionApp.readString();
            SaxionApp.printLine("Enter subnet mask");
            String subnet = SaxionApp.readString();

            String networkAddress = calculateNetworkAddress(ip, subnet);
            SaxionApp.printLine("Network Address: " + networkAddress);
        } else {
            SaxionApp.printLine("Bad choice, please try again", Color.green);
        }
```

```
    }
  }


  public static boolean isOdd(int number) {
    return (number & 1) == 1;
  }

  public static boolean powerOfTwo(int number) {
    return number > 0 && (number & (number - 1)) == 0;
  }

  public static int twosComplement(int number) {
    return ~number + 1;
  }

  public static String calculateNetworkAddress(String ip, String subnet) {
    String[] ipParts = ip.split("\\.");
    String[] subnetParts = subnet.split("\\.");
    int[] networkParts = new int[4];

    for (int i = 0; i < 4; i++) {
      int ipPart = Integer.parseInt(ipParts[i]);
      int subnetPart = Integer.parseInt(subnetParts[i]);
      networkParts[i] = ipPart & subnetPart;
    }

    return String.format("%d.%d.%d.%d", networkParts[0], networkParts[1], networkParts[2],
networkParts[3]);
  }
}
```

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