OPPDRAG 1

# Innhold

* **Netverk** og **tilkobling**
* **Server** og **tjenster**
* **Python** og **GitHub**

## Nettverk og tilkobling

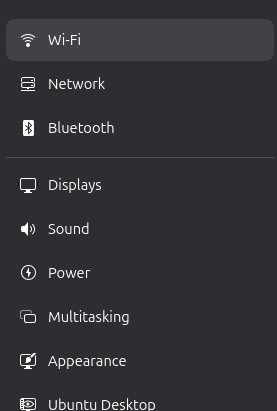
* Åpnet nettverksinnstillinger på Kuben.it og konfigurerte **IPv4 manuelt**.
* Satte inn spesifikk **IP-adresse, nettverksmaske, gateway og DNS** (som oppgaveteksten beskrev).
* Brukte kommandoen ip a på Raspberry Pi for å finne Pi-en sin IP-adresse.
* Brukte ping [IP-adresse] for å teste kommunikasjonen:
* Fra PC à Raspberry Pi.
* Fra Raspberry Pi à PC (fant PC sin IP-adresse med ipconfig).

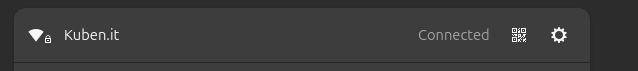
Hvorfor?:

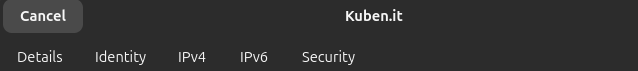
Jeg satt opp **statisk IP-adresse** for å sikre at **rasberry pi** og **laptop** kunne **kommunisere** på samme nettverk **uten at adressen skal endre seg**. Ved å bruke **ip a** og **ipconfig** kunne jeg finne fram **IP-adresen** til **raspberry** **pi** og **laptop**. **Ping** ble brukt som et enkelt verktøy for **å verifisere at nettverkforbindelsen** funker faktisk begge veier.

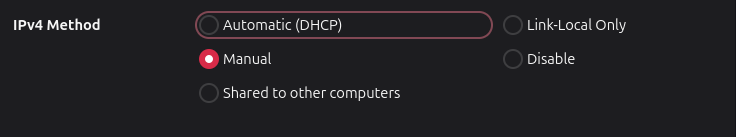
Et bilde som inneholder tekst, skjermbilde, Font

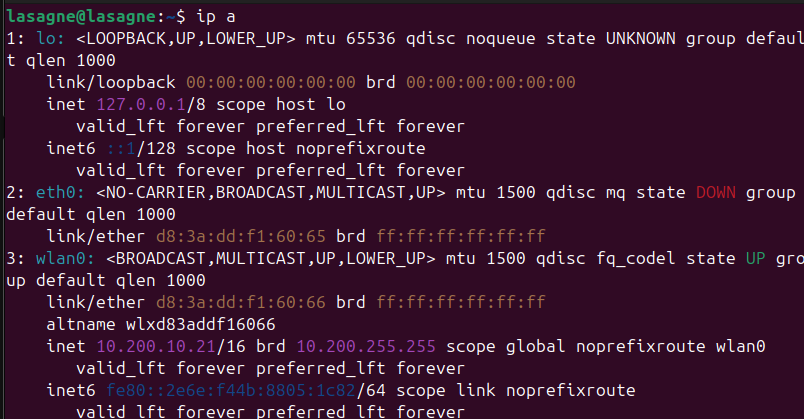
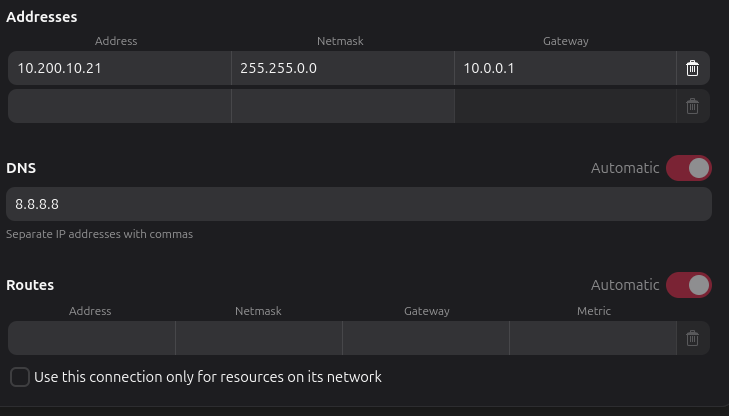
KI-generert innhold kan være feil.

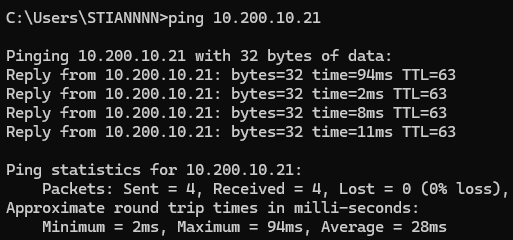


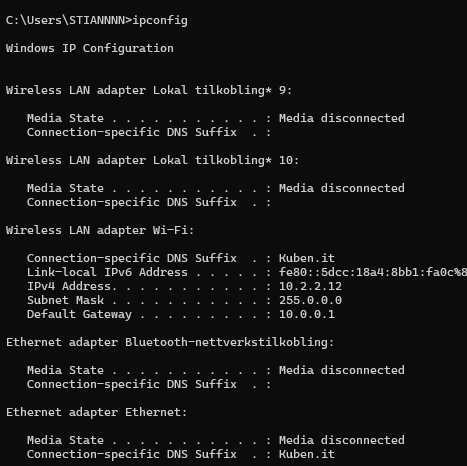


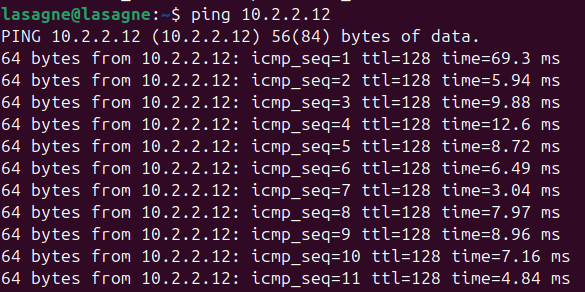










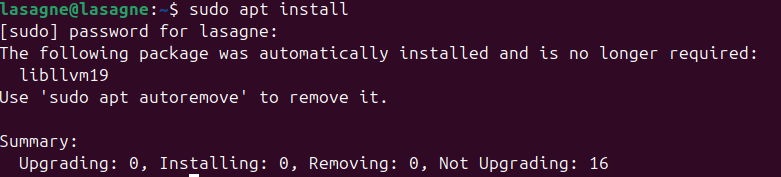


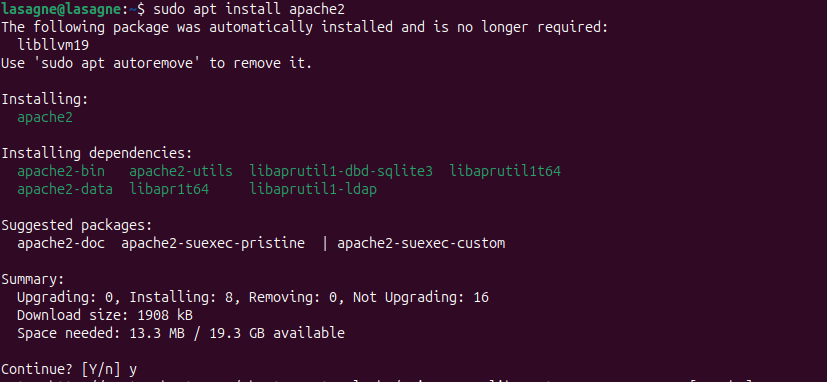
Server og tjenster

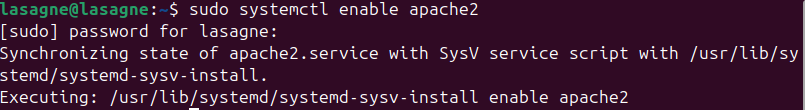
# Apache server

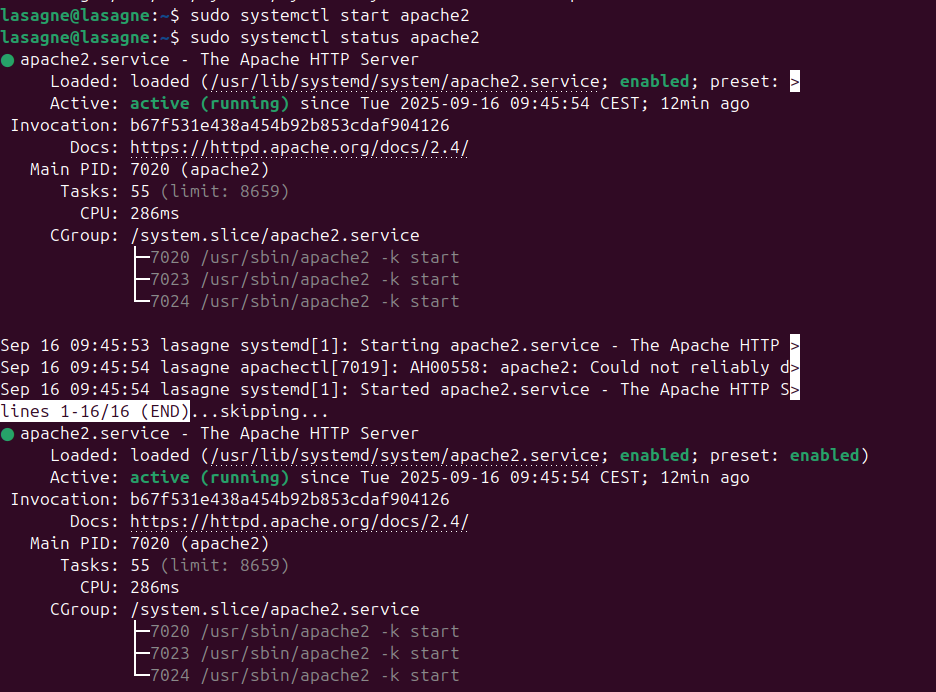
* Oppdaterte **pakkelisten** på **Raspberry Pi** med **sudo apt update**.
* Installerte **Apache2** med **sudo apt install apache2**.
* Aktiverte **Apache2** til å starte automatisk:
* **sudo systemctl enable apache2**
* **sudo systemctl start apache2**
* Kontrollerte **status** med **sudo systemctl status apache2**.
* Konfigurerte **brannmur** slik at **HTTP og HTTPS**-**trafikk** er tillatt.
* Testet **webserveren** ved å åpne **http://[Raspberry Pi IP]** i nettleser.
* Endret innhold i **index.html** under **/var/www/html/** og lagret med **Ctrl+O** i **nano**.

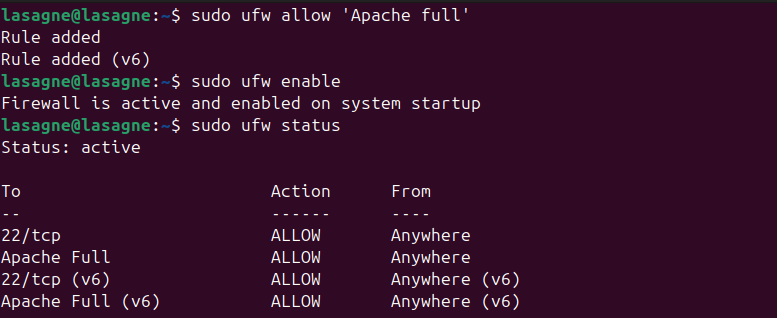
Hvorfor?:  
Jeg installerte **Apache 2** for å gjøre **Rasberry pi** til en **webserver** som kan levere nettsider. **Brannmur-reglene** ble lagt til for å åpne for **webtrafikk** uten blokkering. Ved å gå inn på Pi-en sin IP-adresse kunne jeg teste at serveren svarte. Endring i **index.html** viste at jeg kunne kontrollere **innholdet** på nettsiden, som er viktig for å kunne publisere egne sider.



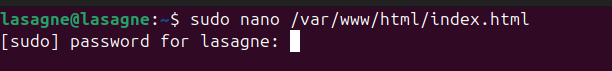


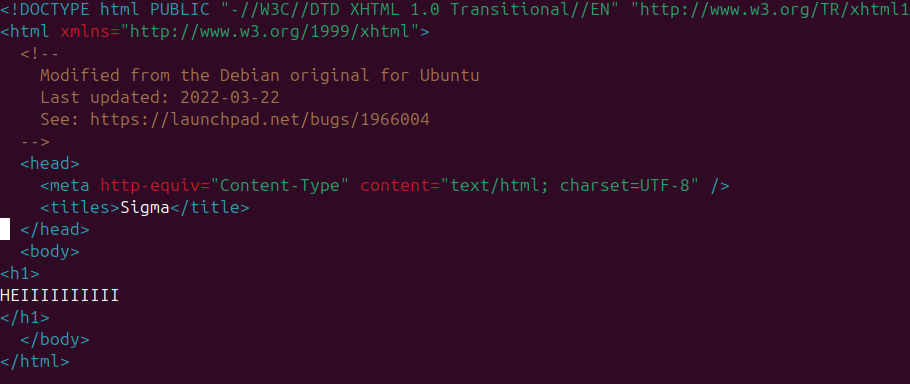












## Minecraft server

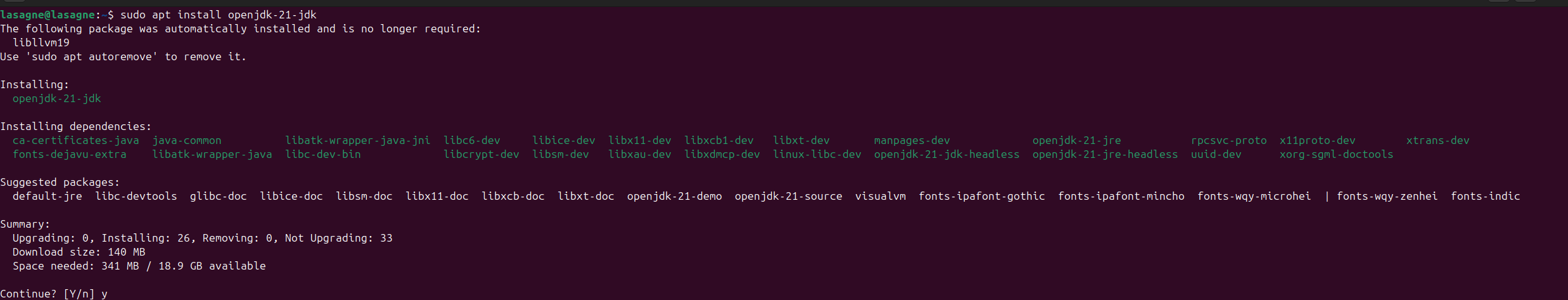
* Installerte Java med sudo apt install openjdk-21-jdk.
* Sjekket at Java-versjonen var korrekt (java -version).
* Opprettet en mappe for Minecraft-serveren og flyttet meg dit:
* mkdir -p ~/minecraft/server
* cd ~/minecraft/server
* Lastet ned server JAR-fil fra Minecraft sin offisielle nettside: [Minecraft server download](https://www.minecraft.net/en-us/download/server) og la filen i servermappen.
* Prøvde å starte serveren med:
* java -Xmx1G -Xms1G -jar server.jar nogui
* 

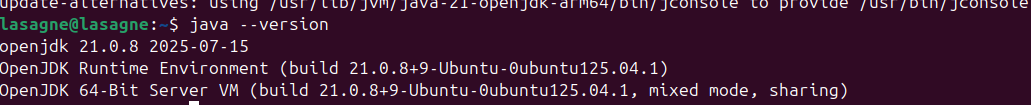
Fikk en feilmelding fordi filen eula.txt ble opprettet automatisk.

* Aksepterte **EULA** ved å åpne eula.txt med nano, endre eula=false til eula=true og lagre (Ctrl+O).
* Startet serveren på nytt med samme kommando.
* Kunne ikke logge inn via Minecraft-klienten på grunn av brannmurbegrensning.
* Lagde en **brannmurregel** for å tillate trafikk til Minecraft-serveren.
* Økte RAM-innstillingene til 2G for bedre ytelse:
* java -Xmx2G -Xms2G -jar server.jar nogui
* Etter dette kunne serveren startes og klienten logget på uten problemet.

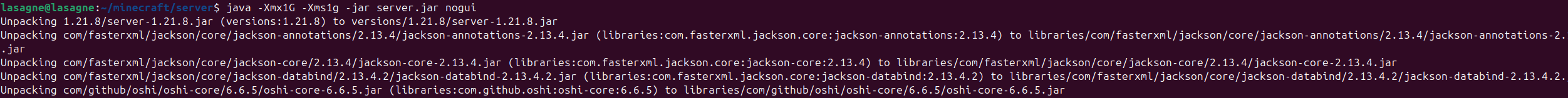
Hvorfor?:

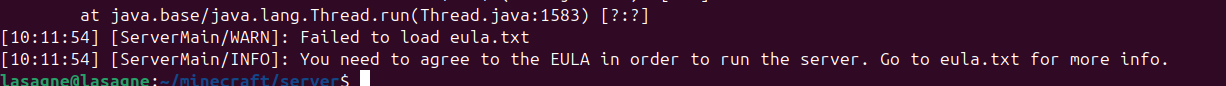
**Java** er **nødvendig** fordi **Minecraft server** kjører på **Java platformen**. **Server mappen** og **JAR-filen** organiserer filene slik at **serveren** kjører **isolert**. **Eula.txt** må **godtas** for å overholde **Mojangs brukervikår**, ellers starter ikke serveren. **Brannmurregelen** er nødvendig for å **tillate innkommende trafikk** på **serverporten**, ellers kan ikke **klienter** koble seg. Ved å øke **ram** sikrer vi at serveren kan håndtere flere spillere og prosessere uten å krasje.

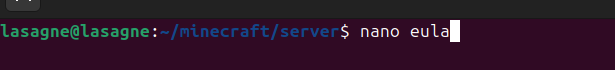


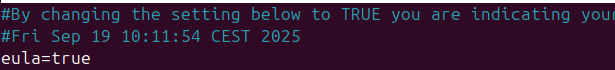


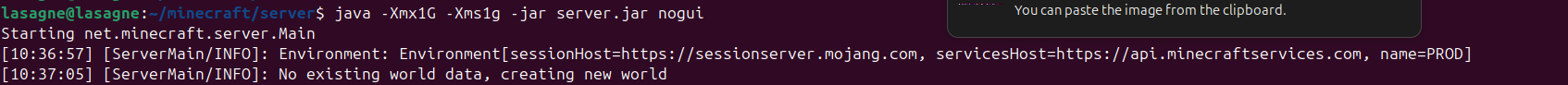


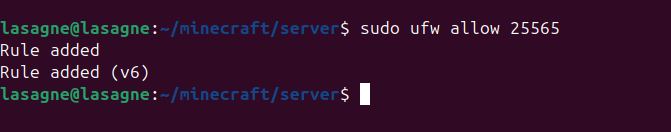












Et bilde som inneholder skjermbilde, PC-spill, Programvare for videospill, Digital komposisjon

KI-generert innhold kan være feil.

## Python og GitHub

* Opprettet en **mappe** og la inn **Python-skriptet** (hei.py).
* **Initialiserte Git** og la til filene (git init, git add ., git commit -m "First intial commit").
* Knyttet til GitHub-repo (git remote add origin [URL]).
* Kjørte **Python-skriptet** med python3 hei.py.
* Pushet koden til **GitHub** med **git push origin main**.
* Testet **git clone** for å laste ned repo til en ny mappe.
* **Verifiserte** at filene ble lastet ned med **ls**.
* Kjørte skriptet igjen fra den klonede repoen.
* Brukte **git pull** for å sjekke etter oppdateringer (ingen endringer denne gangen).

Hvorfor?:

Jeg brukte **Git** for å **versjonskontrollere** koden og **GitHub** for å lagre den **eksternt**. På denne måten kan jeg enkelt **dele kode**, **samarbeide** og **hente den tilbake** på **andre enheter**. **Git clone** og **git pull** demonstrerer hvordan man kan jobbe med samme prosjekt på flere maskiner. Å kjøre **Python-skriptet** både før og etter push/pull viser at filene fungerer og er synkronisert mellom Raspberry Pi og GitHub.





