**Database delen av nettsiden:**

Databasen er satt opp med to tabels: “users “og “messages”.

**users Tabell:**

user\_id: Auto-generert ID for å identifisere ulike brukere.

username: Brukerens brukernavn for nettstedet og meldingssystemet.

email: Brukerens e-postadresse for videre kommunikasjon.

password: Passordet til brukeren (lagret etter hashing av passordet).

is\_admin: En kolonne som indikerer om brukeren har administratorrettigheter (0 eller 1).

created\_at: Registrerer tidspunktet da kontoen ble opprettet.

**Messages Tabell:**

message\_id: Unik ID for hver melding.

sender\_id: ID-en til avsenderen av meldingen (det samme som user\_id).

recipient\_id: ID-en til mottakeren av meldingen.

message\_content: Inneholder teksten i meldingen.

timestamp: Angir tidspunktet for når meldingen ble sendt/laget.

**Melding delen av nettsiden:**

**Hovedsiden - index.php**

index.php fungerer som hovedsiden for prosjektet og informasjon “om meg” blir vist. Index.php har en “contact” side av nettsiden som har en “form” med en “label” der actionen av formen er send\_message.php.

**send\_message.php**

Denne filen er ansvarlig for å sende meldinger. Når en melding blir sendt fra “formen” fra index.php siden, blir “message\_content” lagt inn i message delen av databasen med “sender\_id” og “recipient\_id”.

**your\_messages.php**

Denne filen viser alle meldingene som er mottatt av den innloggede brukeren. Meldingene hentes fra databasen og vises i et tabellformat, inkludert meldingsinnhold og tidspunkt.

**delete\_message.php**

Denne filen sletter en spesifikk melding. Når brukeren sender en forespørsel om å slette en melding, blir den valgte meldingen slettet fra databasen.

**admin\_messages.php**

Denne filen er en admin versjon av “your\_messages.php” der alle meldingene fra alle de forskjellige brukerne blir vist.

**Brukerstøtte delen av nettsiden:**

**Account.php**

Denne filen er en samling av alle de forskjellige php filene som kjører filene gjennom from og “submit” knapper. Account.php inneholder alle de forskjellige endringer av informasjonen som er lag til i databasen. Her er alle de forskjellige filene som blir kjørt via accout.php:

**change\_username.php**

Denne filen håndterer endringen av brukernavn. Når brukeren sender inn formen med det nåværende og nye brukernavnet, verifiserer denne filen først det nåværende brukernavnet mot databasen. Hvis det nåværende brukernavnet er riktig, blir brukernavnet oppdatert i databasen.

**change\_email.php**

Denne filen håndterer endringen av email. Når brukeren sender inn formen med det nåværende og nye emailen, verifiserer denne filen først det nåværende emailen mot den som er registrert i databasen. Hvis det nåværende emailen er riktig, blir emailen oppdatert i databasen.

**change\_password.php**

Denne filen håndterer endringen av passord. Når brukeren sender inn formen med det nåværende og nye passordet, verifiserer denne filen først det nåværende passordet mot den som er registrert i databasen. Hvis det nåværende emailen er riktig, blir emailen oppdatert i databasen.

**delete\_account.php**

Ansvarlig for å slette brukerkontoer. Når brukeren bekrefter slettingen, blir all brukerrelatert informasjon fjernet fra databasen, inkludert meldinger som er sendt eller mottatt av brukeren.

**logout.php**

Tar seg av brukerens utlogging. Denne filen tømmer alle “session” variabler og omdirigerer brukeren til hovedsiden etter utlogging.

**Logg inn og registerings delen av nettsiden:**

**register.php**

Håndterer registreringsprosessen. Når en bruker sender inn registreringsskjemaet, sjekker denne filen om det valgte brukernavnet allerede eksisterer i databasen. Hvis ikke, blir passordet hashet og lagt til i databasen.

**login.php**

Behandler brukerens innlogging. Filen sjekker brukernavn og passord mot databasen. Hvis brukerens legitimasjon er riktig, blir bruker-ID-en lagret i en økt, og brukeren blir omdirigert til hovedsiden. I login blir også user\_id lagret i en session.

**Hvordan sette opp serveren:**

Til dette prosjektet så bruker vi “Ubuntu” for å hoste en server. Her er et steg for steg hvordan sette opp en server til protifolio nettsiden.

**Steg 1:**

**Skru på Hyper-V**

For å kunne kjøre ubuntu på en windows maskin så bruker vi noe som heter en “virtuell maskin”. For å kunne opprette en virtuell maskin så bruker vi et Microsoft-program som heter “Hyper-v”. For å skru på hyper-v må vi først åpne “CMD” som en administrator og skrive inn kommandoen:

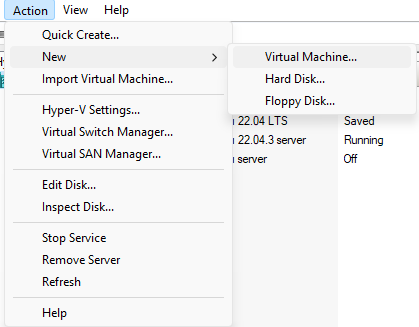
**DISM /Online /Enable-Feature /All /FeatureName:Microsoft-Hyper-V**

Etter kommandoen har blitt skrevet inn så må du restarte pc-en.

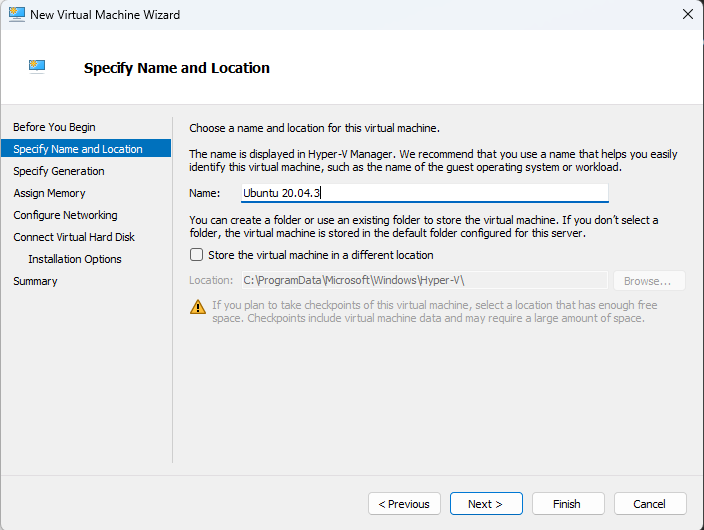
**Steg 2:**

**Sette opp ubuntu i Hyper-V**

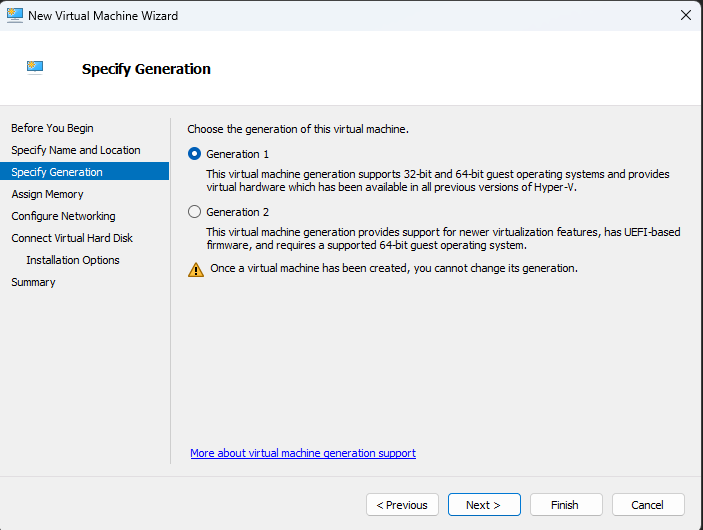
Etter du har restartet pc-en må du åpne programmet “Hyper-V Manager” og trykke på knappen der det står “action” deretter trykker du på “New” og “Virutal Machine”. Se skjermbildet under som referanse:



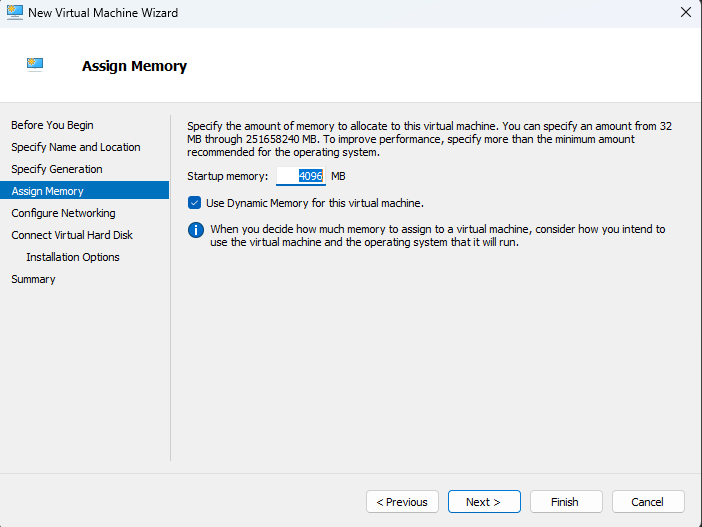
Etter du har lagd en virtuell maskin så trykker du på “next” knappen i det første vinduet. I det andre vinduet vil du få opp en textboks der du kan skrive navnet på serveren. Kall serveren for “Ubuntu 20.04.3 server”. Og deretter klikk “next” knappen.



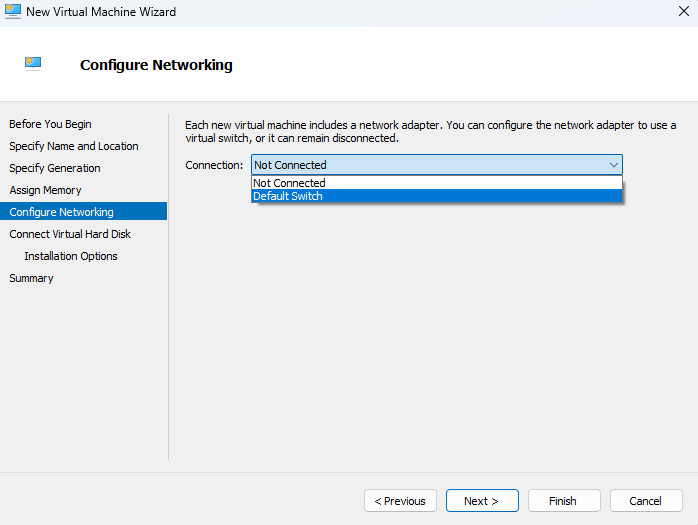
I det neste vinduet huker du av “Generation 1” og klikker “next” knappen.



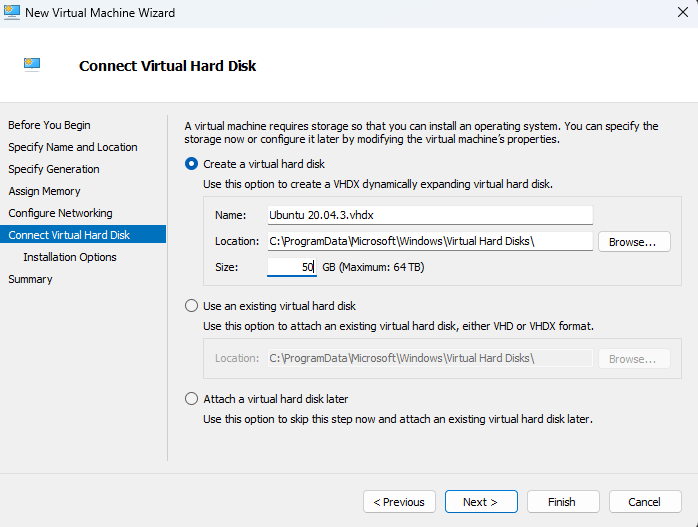
I det neste vinduet som dukker opp så skal du skrive hvor stor del av rammen din den virtuelle maskinen kan bruke. Ettersom at jeg setter opp serveren på en laptop med lite ram så velger jeg standard mengden med ram som den virtuelle maskinen har lov til å bruke. Hvis du setter opp på en maskin med mer ram så kan du øke mengden med ram som den virtuelle maskinen bruker. Etter du har valgt mengden med ram så klikker du “next” knappen.



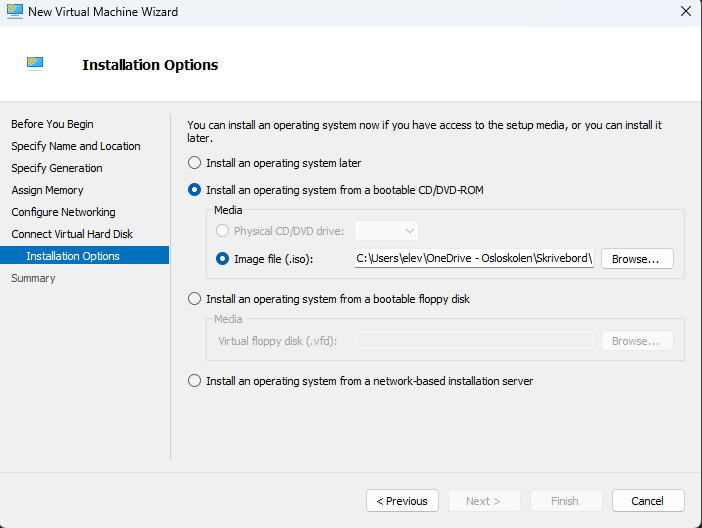
I det neste vinduet endrer du “connection” til “default Swtich”.



I det neste vinduet så endrer du “size” til 50GB. For å sette opp denne serveren så trenger du ikke 50 GB, men i mitt tilfelle så velger jeg å bruke 50GB for å ha en godt fungerende virtuell maskin. Etter du har endret “sizen” på den virtuelle maskinen din så klikker du på “next” knappen.



I det neste vinduet så huker du av “install an operating system from a bootable CD /DVD-ROM” og “Image file (.ios)”. Etter det så trenger vi å installere ubnuntu. Klikk på linken <https://ubuntu.com/download/desktop> og installer ubuntu versjon “20.04.3 deskstop”. Etter du har installert ubuntu så klikker du på “Browse” knappen og trykk på ubnuntu filen som vi nettopp installerte. Deretter klikk på “finish” knappen for å fullføre installasjonen av ubuntu.



Deretter for å åpne den virtuelle maskinen så dobbel klikker du på navnet til den virtuelle maskinen for å åpne den 

Og klikker på start knappen som dukker opp i vinduet til den virtuelle maskinen.

**Steg 3:**

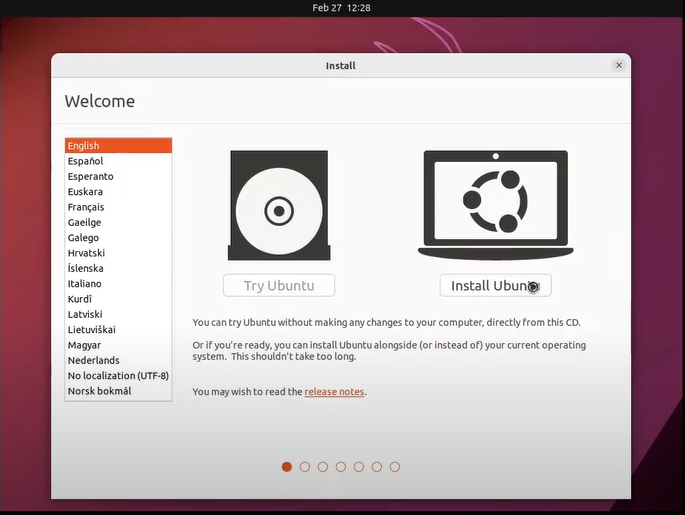
**Konfigurering av Ubuntu**

Etter at du har satt opp en virtuell maskin og starten den må vi sette opp Ubuntu.

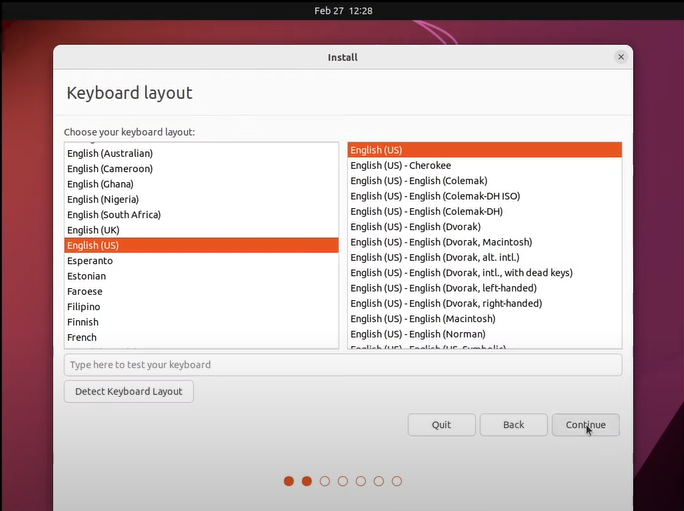
Etter du har klikket på start knappen vil du få opp et vindu med mange forskjellige valg. Klikk på det valget som heter “Try or install Ubuntu” og klikk enter.



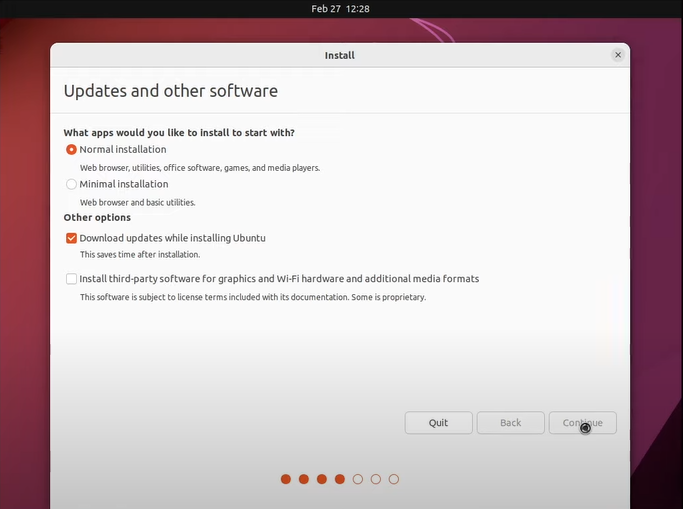
Etter du har klikket enter så vil du få opp et vindu med språk, og to forskjellige knapper. Velg engelsk som språket, og klikk på knappen som heter “Install Ubutu”.



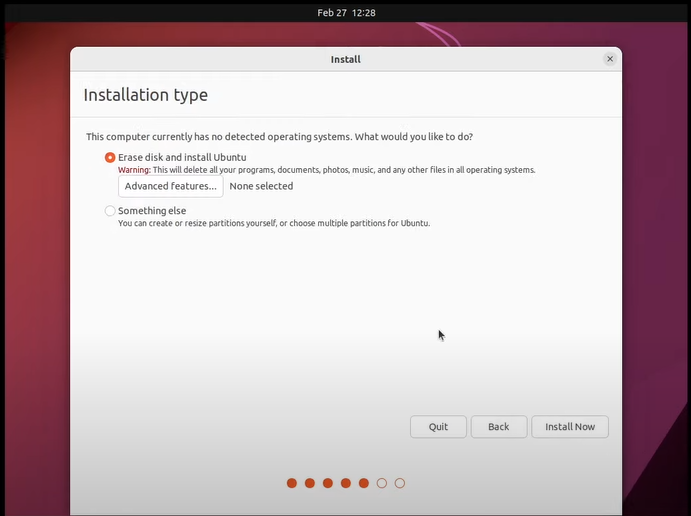
Etter du har klikket installert Ubuntu så vil du få opp ett vindu med valg om tastatur oppsett. Endre tastatur oppsette etter din personlige preferanse og klikk på “Continue” knappen.



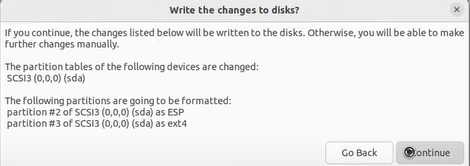
Etter du har klikket på “continue” knappen vil du få opp et vindu med forskjellige bokser du kan huke av. Ikke huk av noen av disse alternativene, men klikk på “continue” knappen for å fortsette installasjonen.



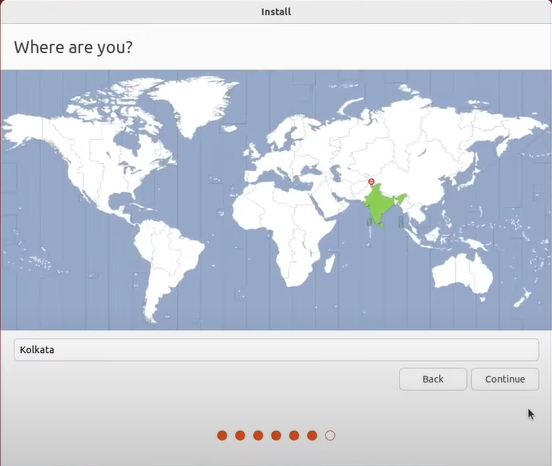
Etter du har klikket “continue” så vil du få opp et vindu med to bokser du kan huke av. Huk av alternativet som heter “Erase disk and install Ubuntu” og klikk “Install Now” knappen nederst på i vinduet.



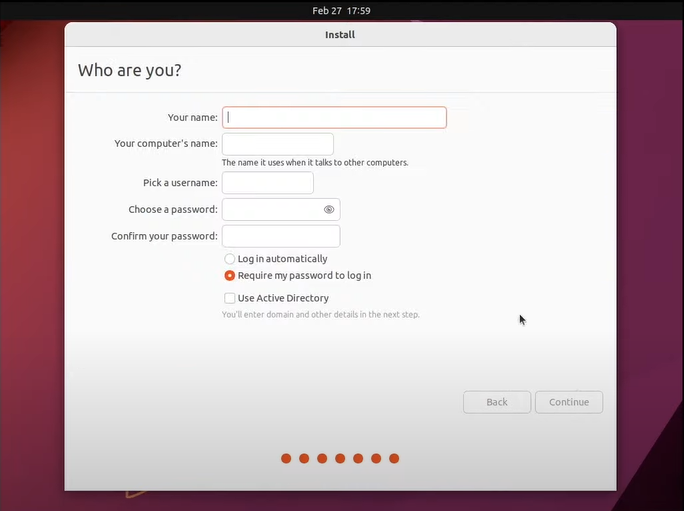
Etter du har klikket på “Install now” knappen vil du klikke “continue” på det vinduet som kommer opp.



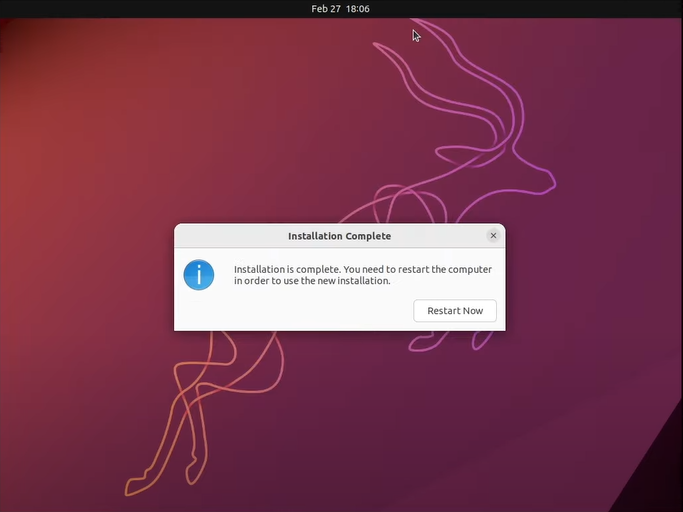
Etter du har klikket på “continue” knappen så vil du få opp et vindu der du kan sette posisjonen din. Sett posisjonen din ettersom hvor du befinner deg og klikk på “continue” knappen.



Etter at du har satt posisjonen din så vil du komme til et vindu der du kan skrive inn et brukernavn og et passord. Skriv inn brukernavnet ditt og skriv inn passordet ditt. Husk at det kan være greit i dette tilfelle å ha et kort passord til opprettingen av “apache” siden du kommer til å måtte skrive inn passordet ditt ofte når vi skal sette opp serveren. Etter du har fylt ut all informasjonen klikk på “continue” knappen for å ferdigstille installasjonen av Ubuntu.



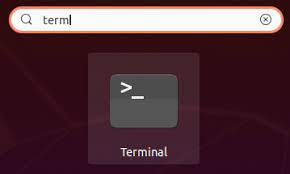
Etter installasjonen av Ubuntu er ferdig klikk på “Restart Now” knappen som kommer opp for å fullføre installasjonen av ubuntu.



**Steg 4:**

**Installere, og sette opp apache, mariadb og php:**

Det første vi gjør når vi skal sette opp apache er å åpne “terminalen” i ubuntu. Klikk på windows knappen på tastaturer ditt og søk opp “terminal”. Ikonet til terminalen ser slik ut:



Åpne terminalen og skriv inn følgene kommandoer:

**sudo apt update**

**sudo apt install apache2**

Disse kommandoene er for å sette opp apache2. Apache2 er det vi brukes for å hoste en server.

Etter vi har installert apache2 så må vi installere php sammen med apache. Skriv in følgene kommando:

**sudo apt install php libapache2-mod-php php-mysql**

Etter vi har installert php så trenger vi å installere mariadb for databasen av nettsiden. Skriv in følgende kommando:

**sudo apt install mariadb-server**

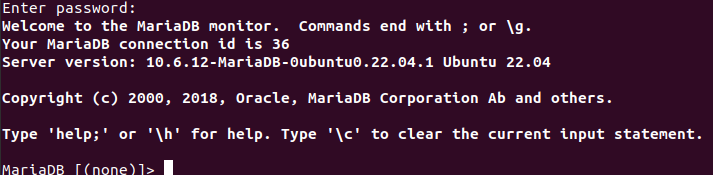
**sudo mysql\_secure\_installation**

Etter å ha skrevet inn de siste kommandoene vil det komme opp flere ja eller nei alternativer til hvordan installere mariadb. Det er 6 spørsmål totalt. Når du skal svare ja eller nei på spørsmålet bruker vi bokstavene “N” eller “Y”. Svar i følgende rekkefølge på de ulike spørsmålene: N, N, Y, N, Y, Y.

Etter å ha installert mariadb så skal vi sette inn databasen vår. For å sette inn databasen vår må vi først lage en database. Skriv inn følgende kommandoer:

**sudo mysql -u root –p**

Etter du har skrevet inn kommandoen så skriver du inn bruker passordet ditt og trykker du på enter. Siden vi ikke har lagt et root passord så trykker vi på enter en gang til. Når vi er inne i mariadb vil det se slikt ut:



Nå kan vi begynne å sette opp databasen til serveren. Først skriver vi inn følgende kommando for å opprette en database:

**CREATE DATABASE chat\_system;**

Etter at vi har opprettet en database som heter chat\_system så må vi opprette tabelsene til databasen. Skriv inn følgende kommandoer:

**USE DATABASE chat\_system;**

Deretter så må vi skrive inn database koden. Her er koden til databasen vår:

CREATE TABLE users (

user\_id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(50) NOT NULL,

email VARCHAR(100) NOT NULL,

password VARCHAR(255) NOT NULL,

is\_admin TINYINT(1) DEFAULT 0,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE messages (

message\_id INT AUTO\_INCREMENT PRIMARY KEY,

sender\_id INT NOT NULL,

recipient\_id INT NOT NULL,

message\_content TEXT NOT NULL,

timestamp TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (sender\_id) REFERENCES users(user\_id),

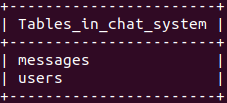
FOREIGN KEY (recipient\_id) REFERENCES users(user\_id)

);

Etter vi har skrevet inn database koden vår så dobbeltsjekker vi at databasen har blitt lagt til ved følgende kommando:

**SHOW TABLES;**

Hvis alt er riktig så burde vi få opp en boks med to tables som heter “messages” og “users”.



Dette betyr at mariadb er installert og databasen vår fungerer.

Etter at vi har satt opp mariadb må vi få in resten av koden til nettsiden og hoste det. Får å gjøre dette på enklest mulig måte bruker vi git. Først så trenger vi å installere git ved å bruke følgende kommandoer i databasen:

**sudo apt install git**

Etter at vi har installert git så skriver vi inn følgende kommando for å klone prosjektet vårt:

**git clone https://github.com/Ortega-juuhl/PortifolioWebsite.git**

Etter at vi har klonet prosjektet vårt så må vi kopiere alle filene som er inni prosjekt mappen til apache sin server mappe. Vi gjør dette med følgende kommando:

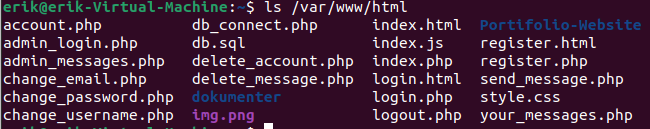
**cd PortifolioWebsite/Portifolio-Website**

**sudo cp -r \* /var/www/html/**

Etter at vi har kopiert alle filene så dobbeltsjekker vi at alle filene ble kopiert ved følgende kommando:

**ls /var/www/html**

Da burde noe lignende komme opp:



Etter vi har kopiert alle filene fra prosjekt mappen til apache sin server mappe så trenger vi å endre hvilken rekkefølge apache prioriterer filer slik at index.php blir prioritert over index.html. Vi gjør dette med følgende kommandoer:

**sudo nano /etc/apache2/sites-available/000-default.conf**

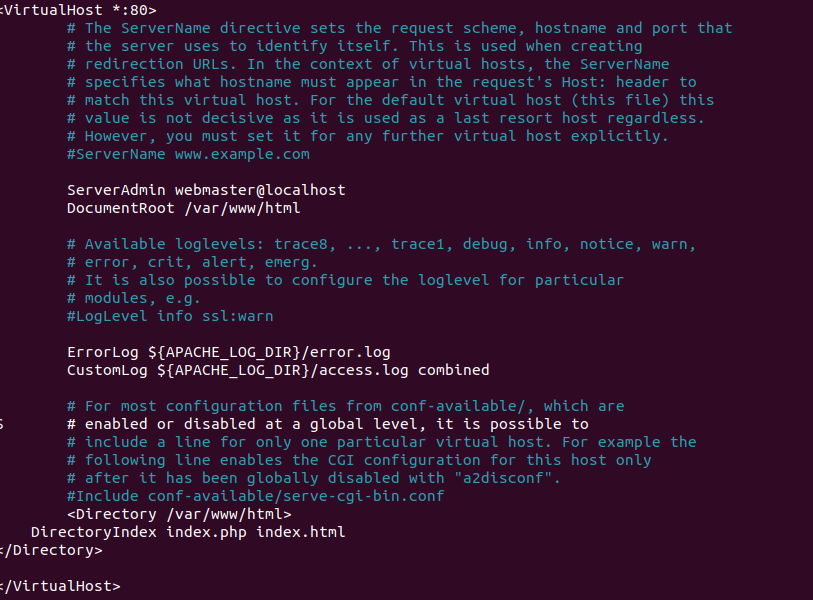
Når du skriver inn følgende kode så vil du åpne en fil med alle apache instillingene. I denne filen så skal vi legge til følgende kode:

**<Directory /var/www/html>**

**DirectoryIndex index.php index.html**

**</Directory>**

Når du har lagt til koden burde filen se slik ut:



Etter at vi har lagt til kodebiten så trykker vi på ctrl + O for å lagre alle endringer og trykker på enter etter.

Til slutt så trenger vi å sette opp en statisk ip-adresse til serveren. Vi gjør dette ved følgende kommandoer:

Først så trenger vi å finne ut gateway adressen som vi gjør med følgende kommando:

**ip route | grep default**

Når du skriver inn denne koden burde du få opp dette:



Numrene etter “via” er de numrene som er gateway adressen din. Etter vi har funnet ut gateway adresses vår så skriver vi inn følgende kommando:

**sudo nano /etc/netplan/your-config-file.yaml**

Dette er en fil der vi skal kopiere inn følgende skript:

**network:**

**version: 2**

**renderer: networkd**

**ethernets:**

**eth0:**

**addresses:**

**- 10.100.1.112/24**

**routes:**

**- to: 0.0.0.0/0**

**via: 172.29.112.1**

**on-link: true**

**nameservers:**

**addresses: [8.8.8.8, 8.8.4.4]**

Endre der det står “via” i koden til gateway adressen din og trykk på ctrl + o for å lagre endringene, deretter enter og til slutt trykker du på ctrl + x for å gå ut av filen.

Helt til slutt så må vi starte apache og mariadb. Vi gjør dette ved følgende kommandoer:

**sudo systemctl start mariadb**

**sudo systemctl start apache2**

Og også restart vi både apache, og mariadb for at alle endringene skal ha blitt lagret. Vi gjør dette med følgende kommandoer:

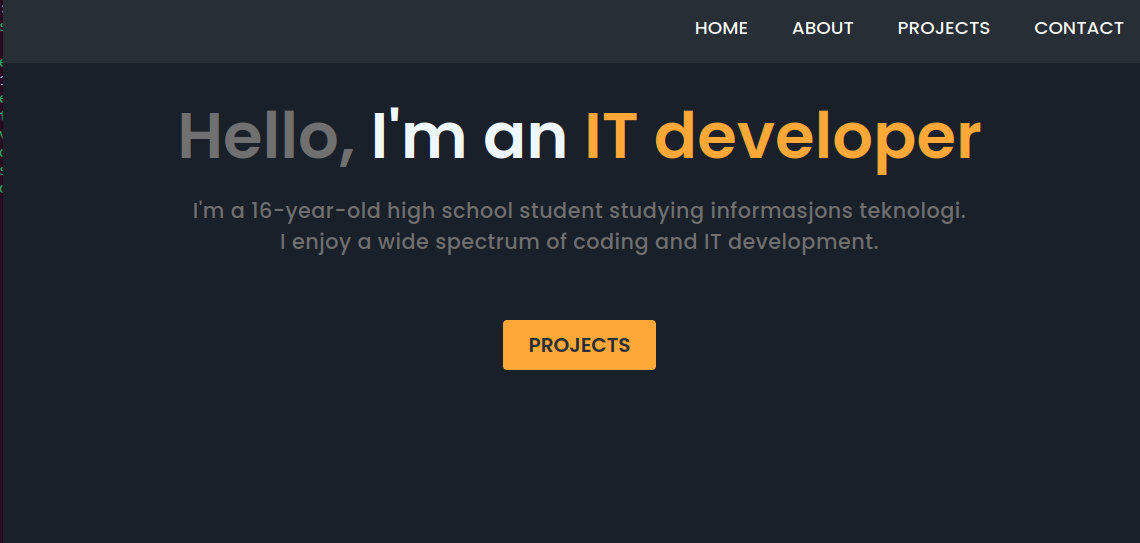
**sudo systemctl restart mariadb**

**sudo systemctl restart apache2**

For å sjekke om serveren fungerer så skriver vi inn ip-adressen:

**10.100.1.112**

Når du har skrevet ip-adressen inn i nettleseren din så burde du få opp noe som dette her:



Dette indikerer på at serveren er oppe og går, og at alt fungerer.