Niina Sormunen

MASTODON PALVELIMEN PERUSTAMINEN JA FEDEROINTI

MASTODON PALVELIMEN PERUSTAMINEN JA FEDEROINTI

Niina Sormunen
Opinnäytetyö
Syksy 2024
Tietotekniikan tutkinto-ohjelma,
ohjelmistokehitys
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Tietotekniikan tutkinto-ohjelma ohjelmistokehitys

Tekijä: Niina Sormunen
Opinnäytetyön nimi: Mastodon Linux-palvelimen perustaminen ja federointi
Työn ohjaajat: Teemu Korpela, Tuula Hopeavuori
Työn valmistumislukukausi ja -vuosi: Syksy 2024 Sivumäärä: 54

Lopputyön aihe on Mastodon Linux-palvelimen perustaminen ja federointi. Tavoitteena oli oppia palvelimien toimintaan, Linux-käyttöjärjestelmän, komentorivin ja Mastodonin tekniikkaa käyttöliittymän takana.

Työssä käytettiin Hetznerin pilvlpalvelua, johon ensin asennettiin Ubuntu, ja Mastodonin omia asennusohjeita. Pääosa työstä tehtiin MacOS-käyttöjärjestelmän terminaalin kautta. Työ itsessään jakautui kolmeen osaan: vaaditun tekniikan selvittämiseen ja muuhun taustatyöhön, asennus-pohjatyöhön ja itse asennukseen. Tekstissä käsitellään myös lyhyesti Mastodonin taustalla olevaan ActivityPub protokollaa.

Tavoitteena ollut Mastodon-palvelimen perustaminen toteutui, vaikka siinä olikin haasteita ja vaati paljon ongelmanratkaisukykyä. Projekti jatkuu opintojen jälkeen luodun palvelimen parissa ja siitä toivotaan kehittyvän ainakin muutaman sadan ihmisen yhteisö.

Mastodon, Linux, palvelin, federointi, sosiaalinen media, ActivityPub

ABSTRACT

Oulu University of Applied Sciences
Degree Programme in Infromation Technology, Option of Software Development

Author: Niina Sormunen
Title of thesis: Founding a Linux Mastodon server and federating it
Supervisors: Teemu Korpela, Tuula Hopeavuori
Term and year when the thesis was submitted: Autumn 2024
Number of pages: 54

The subject of the thesis is Founding of a Linux Mastodon server and federating it. The goal was to learn how servers work, how Linux operating system work and what the technical processes are behind the Mastodon user face.

The project used Hetzner cloud service, where the first install was Ubuntu OS and then Mastodon's own installation guides. The work was done mainly via MacOS Terminal and it was broadly divided into three parts: finding out the required technology and other background research, installation prerequisites and the actual installation. The thesis also discusses briefly ActivityPub, the protocol behind Mastodon.

The main goal, founding the server succeeded, even though there were plenty of challenges and it required a lot of problem solving skills. The project will continue after the studies and hopefully will develop into a community of few hundred users.

Keywords:

Mastodon, Linux, ActivityPub, server, federation, social media

SISÄLLYS

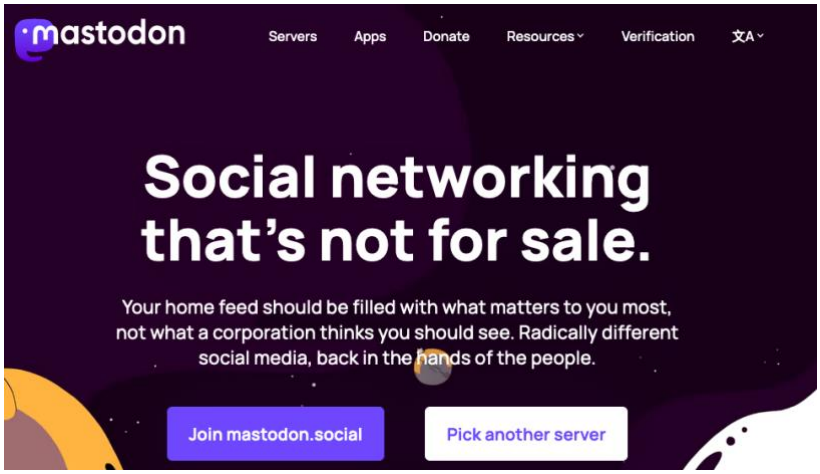
1 JOHDANTO

Alkuvuodesta 2024 aloin miettiä sopivia lopputyön aiheita. Olin tässä vaiheessa ollut jo vuoden Mastodon-nimisen sosiaalisen median vakiokäyttäjä ja olin vaihtanutkin palvelinta, missä profillini oli, jo useamman kerran. Huomasin, että useampi seuraajistani oli perustanut jo oman palvelimen, sillä he kokivat, että vain se varmistaa, että palvelin pysyy toiminnassa niin kauan kuin itse haluaa. Sen lisäksi digitoimisto Duden perustaja Roni Laukkarinen on mastodonilaisittain menestynyt palvelimen perustaja ja ylläpitäjä, ja hänen ja monien muiden esimerkistä innostuneena, päätin tehdä lopputyöni Mastodon-palvelimen perustamisesta ja federoinnista.

Opinnäytetyössä käsitellään suunnittelu- ja toteutusvaihe sekä käydään läpi olennaisimmat asiat Mastodon-palvelimen perustamisesta. Jatkossa palvelimesta käytetään sanaa instanssi. Lähdeluettelossa näkyviä lähteitä on hyödynnetty asiaan perehtymisessä.

2 MIKÄ ON MASTODON

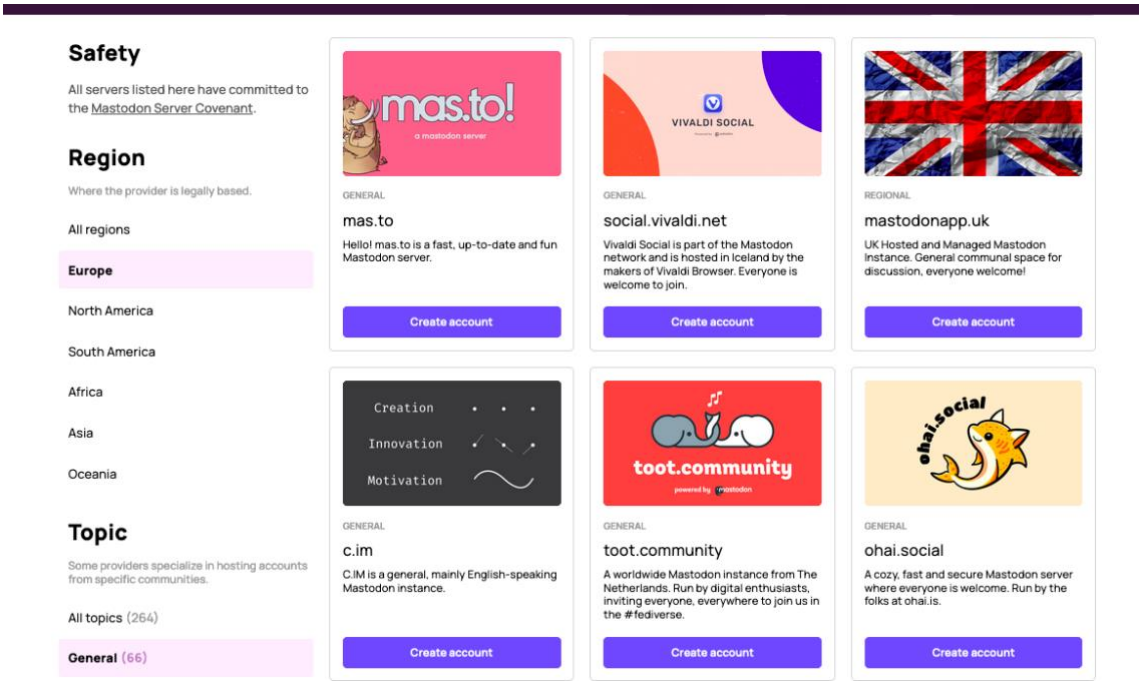
Mastodon (kuva 1) on saksalaisen Eugen Rochkon, vuonna 2016 aloittama projekti Twitterin korvaajaksi. Hän uskoi, että nopean kommunikoinnin alustat ovat liian tärkeitä kuuluakseen yhdelle kaupalliselle palvelulle, joten hän otti tavoitteekseen rakentaa käyttäjäystävällisen mikrobloggaus-tuotteen, joka ei kuuluisi kenellekään ja olisi käytännöllinen jokapäiväiseen käyttöön (Mastodon 2024.)



Kuva 1 Join Mastodonin etusivu. joinmastodon.org

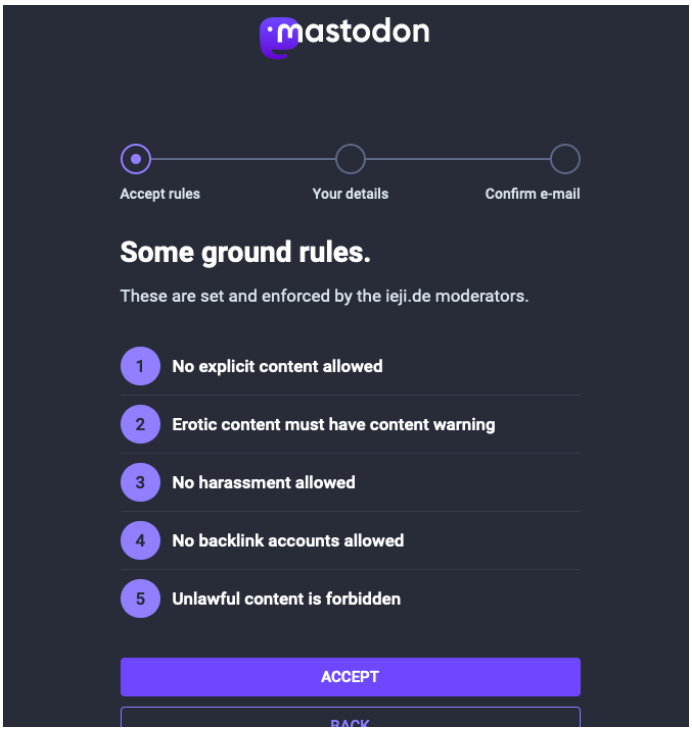
Alusta on siis ohjelmisto, jota käytetään instansseissa. Niitä on eri teemoilla, jotkut voivat olla TV-aiheisia, jotkut keskittyvät teknologiaan ja löytyy myös instansseja, jotka ovat tiettyyn poliittiseen suuntaukseen painottuvia. Instanssin pystyy perustamaan kuka tahansa, kunhan on jonkinlaiset tietotekniikkataidot.

Alusta toimii niin, että käyttäjä menee joinmastodon.org sivulle, joka tarjoaa instanssivaihtoehdoiksi mastodon.socialiin, joka on alkuperäisen perustajan perustama ja kaikista suurin kymmenillä tuhansilla käyttäjillään tai 'valitse toinen instanssi', jota painamalla avautuu näkymä (kuva2), jossa voi etsiä itselleen sopivan instanssin (Mastodon 2024.)



Kuva 2 Saatavilla olevien instanssien kuvallinen sisällysluettelo. joinmastodon.org

Kun instanssi on valittu, ohjaa se sivustolle, jossa instanssi toimii. Siellä täytyy ensin hyväksyä kyseisen instanssin perustajan määrittelemät säännöt (kuva3) (ieji.de 2024.)



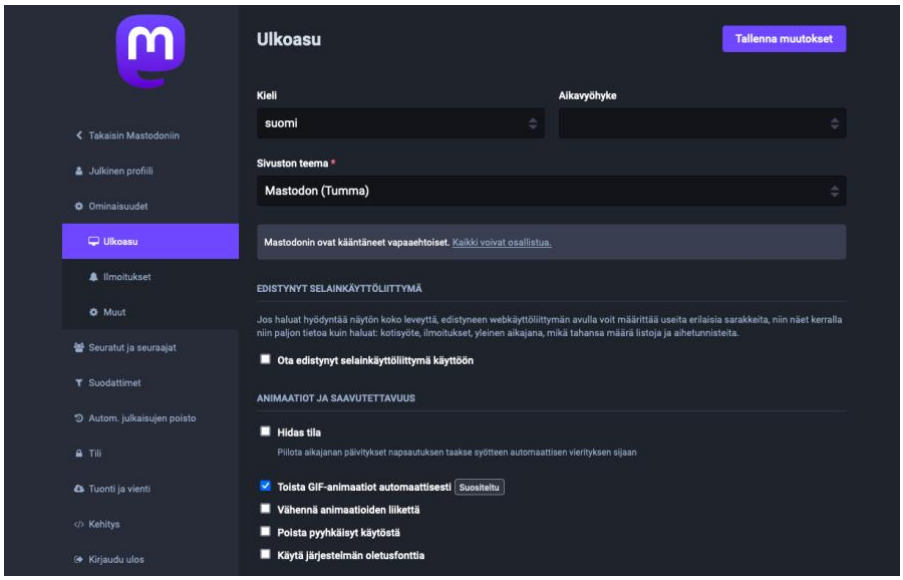
Kuva 3 Iedi nimisen instanssin säännöt. iedi.de

Hyväksymisen jälkeen luodaan käyttäjätunnus ja salasana sekä annetaan sähköpostiosoite vahvistusta varten. Kun tili on vahvistettu, pääsee muokkaamaan omaa profiilia ja lisäämään sinne tietoa itsestään, sekä profiilikuvan.

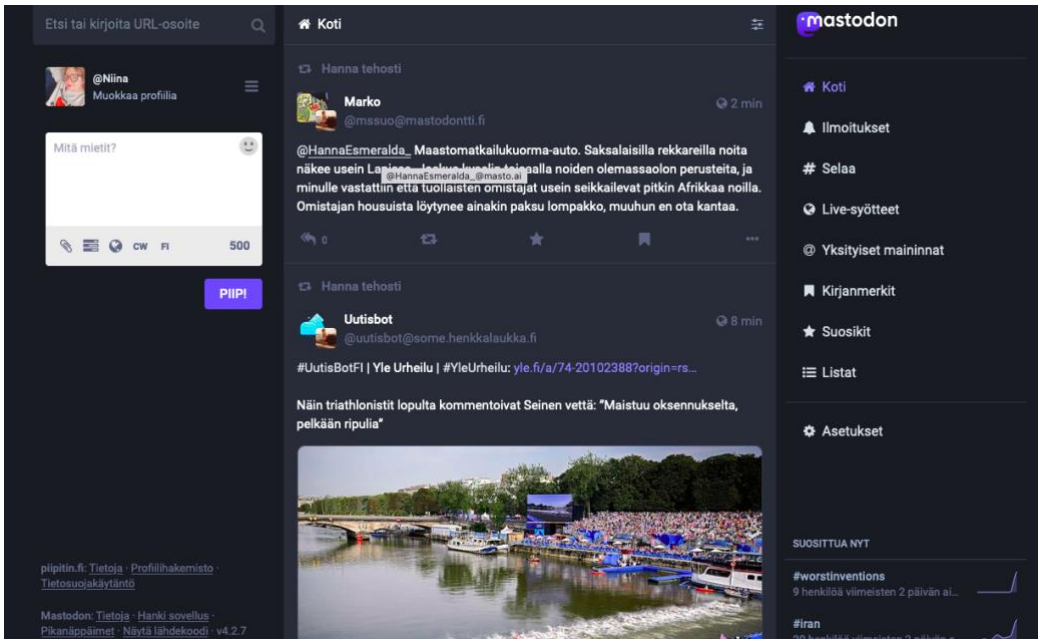
Käyttäjätilin voi luoda niin moneen instanssiin kuin haluaa, mutta jokainen instanssi toimii omalla verkkosivullaan ja jokaiseen pitää kirjautua erikseen.

Instanssin valinnalla ei ole niin väliä, sillä ne kommunikoivat keskenään samoin kuin eri sähköpostit kommunikoivat – Gmailista voi lähettää Outlookiin ja niin edelleen. Mastodonissa onnistuu tilin siirtäminen instanssilta toiseen niin, että seuraajat säilyvät.

Omaa käyttäjäkokemustaan voi hallita hyvin pitkälle (Mastodon 2024.) ja esimerkiksi omaa toottivirtaa voi muokata erilaisilla suodattimilla (kuva 4). Toottivirta koostuu mikroviesteistä eli tooteista, samoin kuin Twitter-virta twiiteistä (kuva 5).



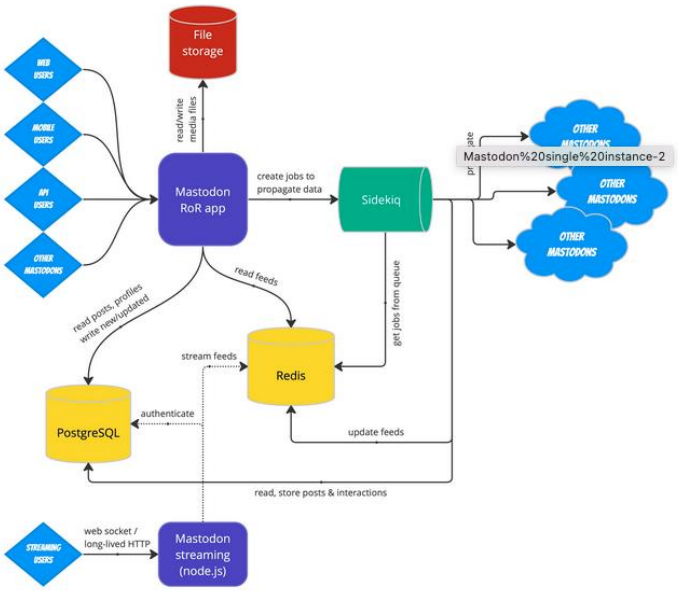
Kuva 4 Piipitin-instanssin käyttäjän asetusten muokkausnäkymä. piipitin.fi



Kuva 5 Piipitin-instanssin käyttäjän pääviestivirta. piipitin.fi

3 MASTODON ARKKITEHTUURI

Mastodonin rakenne ja toiminta on seuraavanlainen:
Käyttäjällä on oma instanssi, joka kommunikoi muiden instanssien kanssa samaan tapaan kuin eri sähköpostit kommunikoivat toisten sähköpostien kanssa. Kuva 6 mukainen arkkitehtuuri:



Kuva 6. Kaavio Mastodonin arkkitehtuurista. https://softwaremill.com

3.1 Mastodon-instanssin perustamiseen tarvittavat taustatyökalut

Kun instanssin haluaa perustaa, tarvitaan ensimmäiseksi web-serveri, jossa voi pyörittää Ubuntu palvelinta. Vertailtuani eri palveluita, päädyin saksalaisen Hetznerin jaettuun CPU (x86) palvelimeen, jossa on RAM-muistia 4 GB, levytilaa 40 GB ja kuukausittainen liikenne saa olla 70 teraa. Tässä vaiheessa käyttäjiä instanssilla tulee olemaan vain yksi, mutta haluan pitää option auki siinä, että avaan mahdollisuuden muillekin liittyä instanssin käyttäjiksi.

Seuraavaksi tarvitaan domain-osoite, joka ohjaa palvelimelle. Tämä tarkoittaa, että domainhost referoi käyttäjät Ubuntu-palvelimen IP-osoitteeseen.

Kolmanneksi vaaditaan transaktionaalinen sähköpostirajapinta, kuten Mailgun, käyttäjien validointia ja kirjautumisia varten.

3.2 Muut vaaditut elementit

Jotta itse varsinaiseen instanssin luomiseen päästään, vaaditaan muitakin elementtejä. Ne ovat:

- Mastodon: itse sosiaalinen media
- Nginx. Monikäyttöinen web-serveriohjelmisto, jota käytetään tässä tapauksessa edustapalvelimena

Näiden lisäksi on tarpeellista olla SSH-yhteys.

4 TAUSTATYÖ ELI PILVIPALVELU

Pilvipalvelu on tietotekniikkapalvelu, jossa eri tietotekniset resurssit, esimerkiksi tallennustila, laskentatehot ja tietoliikenneyhteydet hallinnoidaan ja toimitetaan internetin välityksellä. Pilvessä käytettävä palvelu ei tällöin sijaitse käyttäjän koneella tai organisaation palvelimilla vaan palveluntarjoajan palvelimilla. Pilvipalveluiden käyttö on nykyään keskeinen strategia yrityksille, jotka haluavat pysyä teknologisen kehityksen kärjessä. Pilvi tarjoaa monia etuja, kuten kustannussäästöjä ja skaalautuvuutta. Ne mahdollistavat nopeamman innovoinnin ja käyttöönoton, sillä uusia palveluita ja sovelluksia voi kehittää nopeammin ja matalammin kustannuksin.

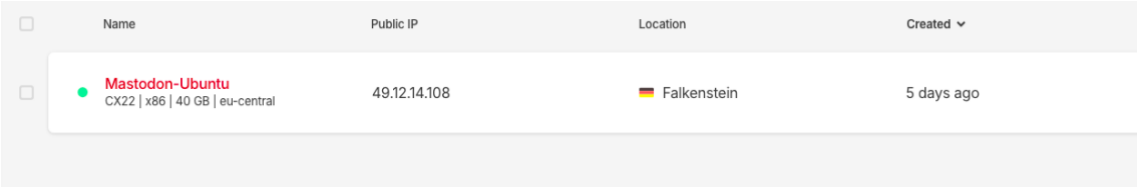
4.1 Pilvipalveluiden päätyypit

Pilvipalveluja on erilaisia, riippuen asiakkaan tarpeista. Näitä on neljää eri päätyyppiä

- PaaS eli Platform-as-a-Service: käyttäjälle näkymätön, mutta mahdollistaa kehittäjille rakentamisen ja testauksen, sekä ajaa sovelluksia ilman, että kehittäjien tarvitsee hallita taustalla olevaa infrastruktuuria. PaaS-palvelut tarjoavat työkaluja, tietokantoja ja palvelinympäristöjä helpottamaan sovelluskehitystä
- BaaS eli Backend-as-a-Service: palvelumalli, jossa valmiiksi rakennetut taustapalvelut toimitetaan ja hallinnoidaan pilvipalveluna. Esimerkkinä toiminnoille on tietokantojen hallinta, käyttäjien todennuksen ja push-ilmoitukset. Muun muassa Firebase on BaaS-palvelu.
- SaaS eli Software-as-a-Service: tyypillisesti selaimella käytettävä sovellus. SaaS-palvelu tarjoaa käyttäjälleen koko sovelluksen käyttöliittymineen, toimintalogiikoineen ja tietokantoineen täysin pilvessä. Esimerkiksi Googlen Gmail on tällainen palvelu.
- IaaS eli Infrastucture-as-a-Service: alimman tason pilvipalvelumalli, jossa asiakkaat voivat vuokrata IT-infrastruktuuria, kuten virtuaalikoneita, tallennustilaa ja verkkoresursseja. Esimerkkinä tälläisestä palvelusta on Amazon AWS.

4.2 Pilvipalvelu Mastodon-serverille

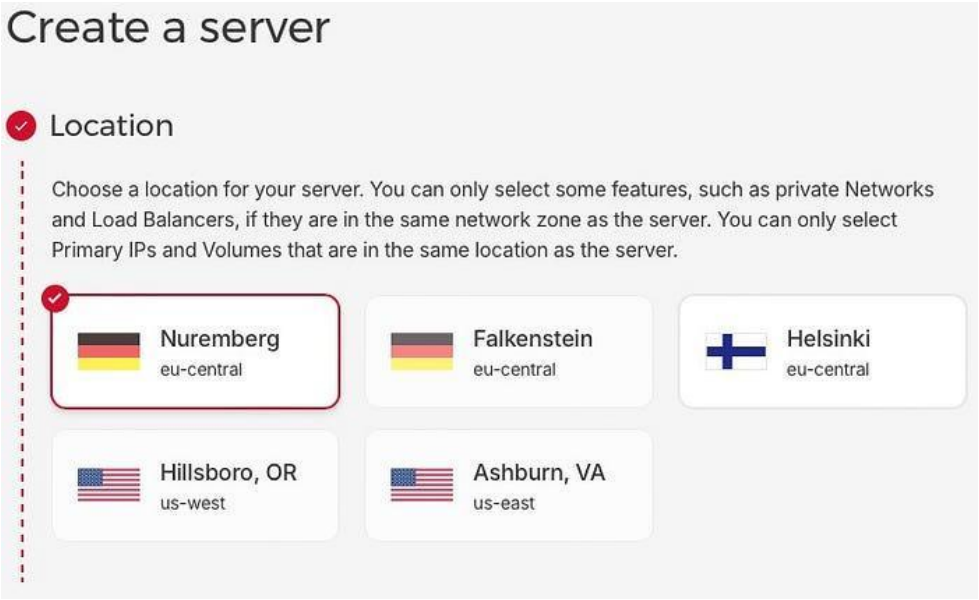
Ensimmäiseksi ennen Ubuntun pilvipalveluun asentamista, täytyi valita palvelu. Palveluksi päätyi Hetznerin IaaS (kuva 7), jossa on Ubuntun vaatima 64-bittinen arkkitehtuuri, tässä tapauksessa amd64/Intel., jossa on RAM-muistia 4 GB ja levytilaa 40 GB. Tämä ylittää Ubuntun minimisysteemivaatimukset, jotka ovat RAM: 2 GB ja levytila: 5 GB.



Kuva 7. Toiminnassa oleva serveri ja domainosoite Hetzner.com

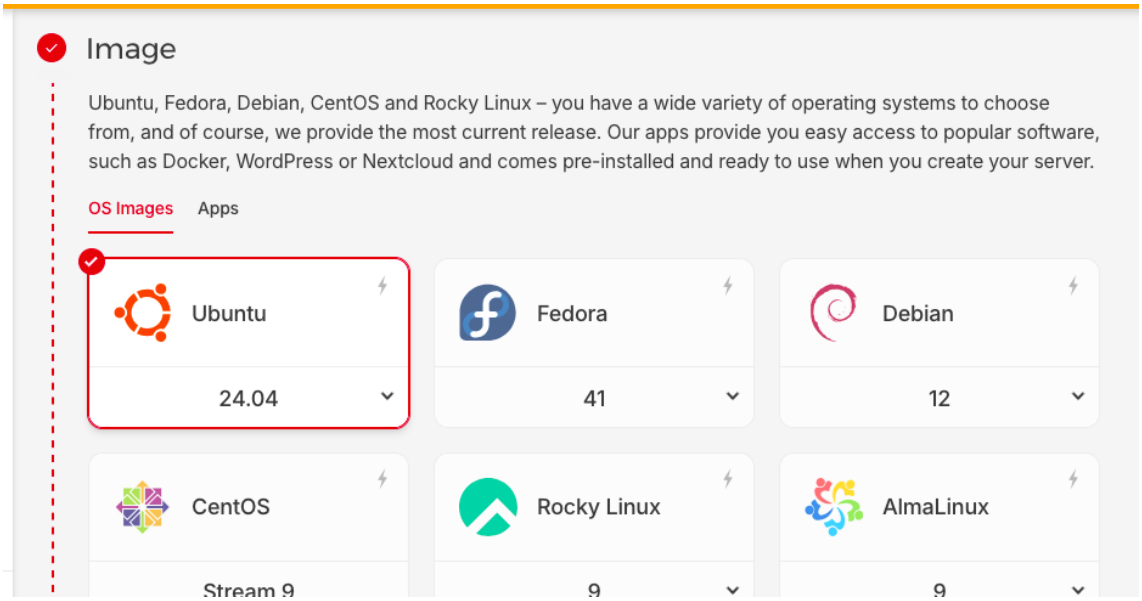
4.3 Serverin asentaminen

Hetzner pilvipalvelun asentaminen on hyvin yksinkertaista ja sitä hankittaessa palvelu käy läpi osiot askel askeleelta. Ensin valitaan serverin lokaatio (kuva 8):



Kuva 8.Luo serveri näkymä. https://medium.com/@benjaminstorm

Tämän jälkeen valitaan käyttöjärjestelmä (kuva 9), joka on aiemmin mainittu Ubuntu. Tämänhetkinen versio on 24.04.



Kuva 9. Pilivpalvelussa käyttöön tulevan käyttöjärjestelmä- valikko. Hetzner.com.

Toisena on valittava serverin tyyppi. Koska luotavassa Mastodon-instanssissa tulee olemaan vain yksi käyttäjä, joka on myös admin, riittää tähän jaettu CPU ja x86 Intel/AMD- arkkitehtuuri. Jos Mastodon-instanssi avattaisiin isommalle käyttäjäjoukolle olisi perusteltua valita x86 AMD.

5 TOTEUTUS

5.1 Serveri

Serveri asennettiin tässä projektissa useamman kerran. Serverin luonnin ja itse Mastodon-asennuksen välillä kului aikaa. Ensimmäisen serverin asennuksen yhteydessä oli luotu SSH-avain, jonka salasana oli sittemmin unohtunut. Kun tässä vaiheessa yritettiin kirjautua serverille, se pyysi root-salasanaa, jonka uudelleen luontia selvitin Hetznerin teknisen tuen kautta. He vastasivat seuraavanlaisesti:

”Root- salasana lähetetään vain, jos alussa ei valita SSH-avainta, kun serveria luodaan. Jos SSH-avain on valittu, sillä autentikoidaan yhteys serveriin. Ota huomioon, että servereillä on eri SSH asetukset, riippuen luotiinko serveri SSH avaimilla vai root-salasanalla. Serveri , joka on luotu julkisella avaimella, ei anna kirjautua root-salasanalla. Ainoa tapa, jolla tämä onnistuu näillä koneilla on VNC-konsolin kautta. Jos root-salasanalla haluaa kirjautua, täytyy asetukset muuttaa jotta sallitaan root-käyttäjän kirjautua sisään salasanalla.

Jos salasanalla kirjautuminen SSH:n kautta on välttämätöntä, ja jos cpU on suhteellisen uusi ja siinä on vähän tai ollenkaan konfigurointia, on mahdollista luoda uusi serveri ilman SSH-avainta. Vaihtoehtoisesti, SSH-konfiguraatio olemassa olevassa serverissa voidaan muuttaa käyttämällä VNC-konsolia ja luotua salasanaa.

Jos salasana halutaan sallia kirjautuessa pilvipalvelussa sijaitsevaan serveriin, jolle alunperin oli luotu SSH-avain, rivi “#PermitRootLogin prohibit-password” täytyy muuttaa “PermitRootLogin yes”.

Ota huomioon, että pelkän salasanan käyttäminen autentikointiin tuo omat riskinsä. Suosittelen käyttämäään SSH-avaimia.”

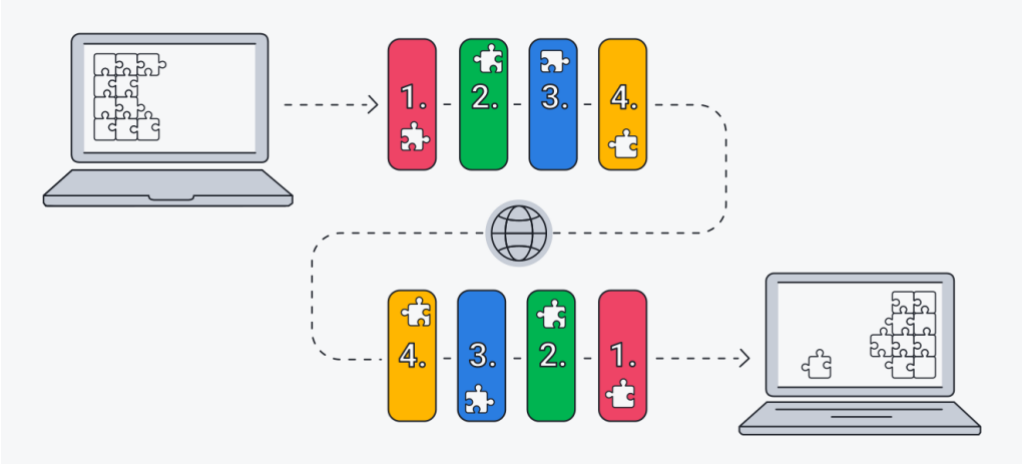
Selvisi, ettei tässä tapauksessa root-salasanaa onnistu vaihtamaan sudo passwd root, joten Ubuntu-serverin luonti aloitettiin alusta.

5.2 SSH-yhteys

Mastodon-asennus alkaa SSH-yhteyden luomisella. SSH on suojattu yhteys, jolla saa etäyhteyden Linux, Windows ja MacOS palvelimiin. Tämän lisäksi yhteys sallii avainpohjaisen tunnistautumisen, tiedostojen siirron ja liikennetunnelin eli portin eteenpäin viennin. Portin eteenpäin vienti tarkoittaa sitä, että datapaketit voivat siirtyä verkkojen yli, joita ne eivät normaalisti voisi ylittää. Syitä käyttää SSH-yhteyttä on komentojen antaminen serveriin etäisyydeltä, infrastruktuurin ylläpito ja tiedostojen siirtäminen.

5.2.1 SSH:n toiminta

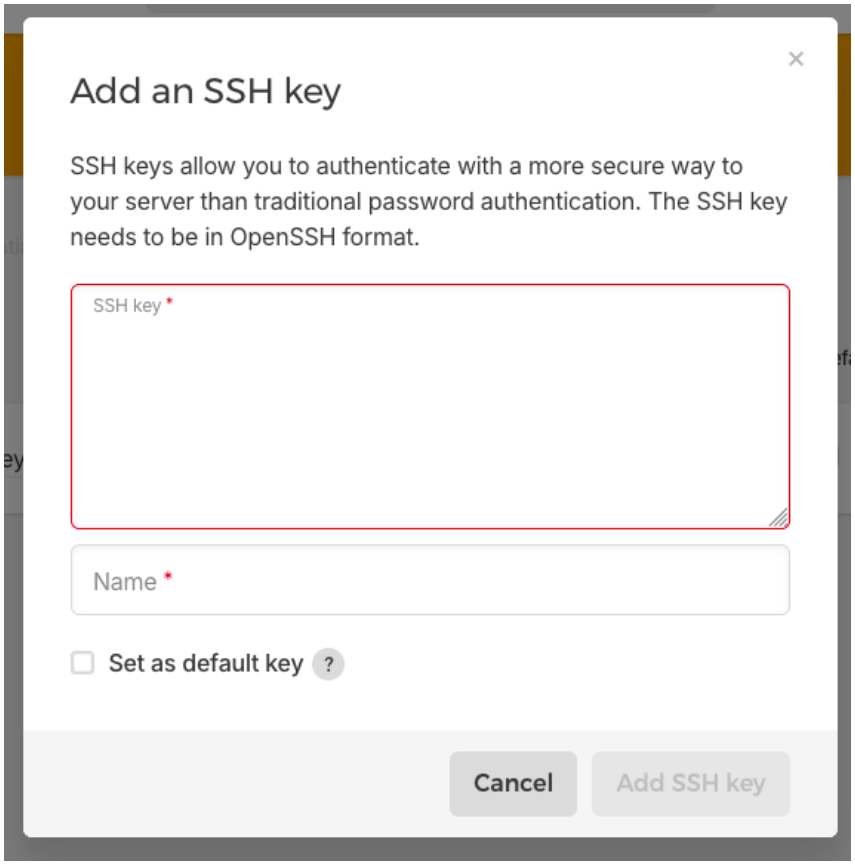
SSH toimii TCP/IP-protokollassa, jotka ovat erillisiä protokollia, joihin moderni internet suurimmalta osin perustuu. Muita protokollia ovat HTTP, FTP ja SMTP.TCP (Transmission Control Proto), joka määrittää mitä paketteja lähetetään, kun IP (Internet Protocol) on löytynyt. TCP jakaa siirrettävän datan osiin, jotka lähetetään neljän kerroksen läpi. Tämä data kootaan kasaan vastaanottavassa päässä päinvastaisessa järjestyksessä (kuva 10).



Kuva 10. Kaavio TCP/IP:n toiminnasta. avg.com.

5.3 Hetzner- yhteys

Hetznerin palvelussa pitää ensin luoda SSH-avain ennen serverin luomista. Kun tili on hankittu, löytyy avaimen luomisosasto Security-osiosta. Sieltä löytyy painike Add SSH KEY, joka avaa näkymän (kuva 11):



Kuva 11. SSH-avaimen lisäysnäkymä. Hetzner.com.

Tähän vaaditaan SSH-avainpari OpenSSH:lla, ja se luodaan komennolla

```
Ssh-keygen -t ed25519 -C “jokin kommentti”
```

Tämä luo sekä yksityisen että julkisen id-ed25519 tiedoston. Julkisessa tiedostossa on .pubp-pääte. Terminalissa näkyy pub-tiedoston sisältävä numerosarja, joka alkaa kirjaimilla SSH226 ja tämä sarja liitetään kuvassa 11 näkyvään SSH-key -osioon. Tämän lisäksi tiedostolle on annettava nimi.

Kun avain on luotu, voi serverin luoda luvussa 4.3 esitetyllä tavalla. Serverin luomisen jälkeen voidaan ottaa SSH-yhteys serveriin, tässä tapauksessa Ubuntu-serveriin. Yhteys luodaan komentoikkunassa. Tässä tapauksessa, kun käytössä on MacOs, on komentoikkuna Terminal-niminen.

Jos käytössä on IPv4, yhteys luodaan komennolla:

```
SSH (käyttäjänimi) root@(ip-osoite) xx.xx.xx.xxx
```

Jos käytössä on puolestaan IPv6, on komento:

```
SSH (käyttäjänimi) root@(ip-osoite) xxxx:xxx:xxxx::x
```

Nämä komennot luovat salatun yhteyden kahden hostin välillä. Salaus murentaa siirrettävän datan osiin, jotka siirtyvät yhteyden välityksellä validoituun kohteeseen. Kun serverille kirjaudutaan alkuperäisellä salasanalla, serveri pyytää muuttamaan salasanan. Tämän jälkeen voi tehdä configurointi-tiedoston (config file). Tässä tapauksessa config-tiedostoa ei muodostunut automaattisesti ja se piti luoda .ssh- kansioon manuaalisesti. Tiedosto editoitiin nanolla, mutta ensin katkaistiin yhteys serveriin. Sitten nano avattiin komennolla:

```
nano ~/.ssh/config
```

Tiedostoon kirjoitettiin seuraavat asiat:

```
HostName: <oma ip-osoite>
User: root
IdentityFile ~/.ssh/id_ed25519
```

Jos tiedostoon olisi lisätty ensimmäiseksi riviksi Host: <uniikki nimi>, voisi serveriin ottaa yhteyden pelkällä SSH <uniikki nimi> komennolla. IdentityFile määrittää sen tiedoston, mistä autorisointi-identiteetti luetaan. Se on niin kutsuttu Private key, joka sallii SSH-yhteyden serverille. Toisin sanoen se on avain, joka sallii Public key -autorisoinnin. Koska rootin käyttäminen pääkäyttäjänä on riskialtista, luotiin uusi käyttäjä admin-oikeuksilla.

```
usermod -aG sudo <newuser>
```

Tämä luo uuden käyttäjän ja antaa hänelle superkäyttäjän oikeudet.

6 MASTODON-ASENNUS

Mastodonin asennukseen on ohjeet joinmastodon.org sivustolla. Siellä käydään läpi kaikki asentamiseen kuuluvat vaiheet tutoriaalin muodossa, jota on käytetty tässä osiossa.

6.1 Pohjatyö

Asentaminen alkaa ennakkoehdoilla, jotka ovat Ubuntu 24.04 serveri, domain-nimi, joka on tässä tapauksessa hankittu Zoner.fi -palvelusta ja on nimeltään fanttila.fi, sekä email, joka on hankittu Mailjet- palvelusta.

Kaikki pohjatyössä annettavat komennot tehdään root-käyttäjänä.

Ensin pitää asentaa curl, wget, gnupg, apt-transport-https, lsb-release ja ca-serfikaatit..

```
Apt install -y curl wget gnupg apt-transport-https lsb-release ca-certificates
```

Tässä kohtaa tuli vastaan ensimmäinen virheilmotus, joka oli ”package gnupg not available”. Tämä ratkaistiin komennolla:

```
sudo apt-get update
```

Tämän jälkeen ensimmäinen komento meni läpi.

6.1.1 Node.js

Seuraavana asennuksena oli node.js. Siihen käytettii seuraavaa komentoa:

```
curl -fsSL https://deb.nodesource.com/gpgkey/nodesource-repo.gpg.key | gpg --dearmor -o /etc/apt/keyrings/nodesource.gpg
```

Tässäkin kohtaa tuli virheilmoitus, joka johtui siitä, että komento oli kopioitu virheellisesti tutoriaalisivulta. Komento ajettiin uudestaan ja se meni läpi.

Seuraava komento node.js -asennusprosessissa oli:

```
echo "deb [signed-by=/etc/apt/keyrings/nodesource.gpg] https://deb.nodesource.com/node_20.x nodistro main" | tee /etc/apt/sources.list.d/nodesource.list
```

Tästä tuli myös virheilmoitus, joka sanoi, että deb-tiedostoa ei löydy. Ratkaisu saatiin askubuntu.com forumilta, jossa ehdotettiin, että komennon eteen laitetaan sudo su -c ja koko alkuperäinen komento aloitetaan ja päätetään hipsuilla. Tämä sitten toimi ja päästiin eteenpäin ennakkoehtojen täyttämisessä.

6.1.2 PostreSQL

Asennus alkoi komennolla:

```
Wget -/usr/share/keyrings/postgresql.asc
https://www.postgresql.org/media/keys/ACCC4CF8.asc
```

Virheilmoitus: “permission denied”. Selvisi, että kun SSH-yhteys oli välillä suljettu ja sitten uudelleen kirjauduttu sisään, käyttäjänä oli toinen serverin käyttäjä. Kun tämä vaihdettiin sudo -I kommenolla rootiksi, komennon ajo onnistui.

Seuraava komento oli:

```
echo "deb [signed-by=/usr/share/keyrings/postgresql.asc] http://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg main" > /etc/apt/sources.list.d/postgresql.list
```

Jälleen tuli virheilmoitus, tällä kertaa puuttuvasta sources.list.d- kansiosta ja postgresql.list tiedostosta. Tähän kokeiltiin ensin sudo su -C, mutta error tuli samasta asiasta. Tätä lähdettiin ratkomaan selvittämällä mistä löytyy sources.list.d. Kun ymmärrettävää selitystä ei löytynyt, tehtiin päätös, että on parempi poistaa kaikki Postgresqllään liittyvä ja aloittaa asennus tästä kohtaa alusta. Tähän löytyi ratkaisu Githubista.

Ensin listattiin kaikki PostgreSQL-paketit:

```
dpkg -l | grep postgres
```

Toisessa vaiheessa poistetiin kaikki tämänhetkiset Postgreql-liitännäiset paketit:

```
sudo apt –purge remove postrsql*
```

Kolmantena poistettiin kaikki postresql data ja hakemistot:

```
sudo rm -rf /var/lib/postgresql/
sudo rm -rf /var/log/postresql/
sudo rm -rf /etc/postgressql
```

Neljäntenä eli viimeisenä vaiheena poistettiin postgres käyttäjä:

```
sudo deluser postgres
```

Tämä mahdollisti Postgresql-asennuksen aloittamisen alusta. Seuraavana oli vuorossa apt-päivitys:

```
apt update

apt install -y \

  imagemagick ffmpeg libvips-tools libpq-dev libxml2-
dev libxslt1-dev file git-core \

  g++ libprotobuf-dev protobuf-compiler pkg-config gcc
autoconf \

  bison build-essential libssl-dev libyaml-dev
libreadline6-dev \
```

```
  zlib1g-dev libncurses5-dev libffi-dev libgdbm-dev \
  nginx nodejs redis-server redis-tools postgresql
postgresql-contrib \
  certbot python3-certbot-nginx libidn11-dev libicu-
dev libjemalloc-dev
```

Seuraavaksi tarvittiin Yarn, joka on avoimen lähdekoodin pakettien hallinnoija, jota käytetään Javascript- projektien riippuvuuksien hallinnointiin. Yarn sallii muiden kehittäjien ratkaisujen käytön omien ohjelmistojen kehityksessä. Jotta oikea versio voidaan automaattisesti asentaa, se toteutetaan corepackin käyttöönotolla. Tähän on komento:

```
corepack enable
```

Tämän jälkeen päästään itse Mastodonin asentamiseen. Aluksi täytyy root-käyttäjänä luoda uusi käyttäjä 'Mastodon', joka saadaan komennolla

```
adduser  --disabled-password Mastodon
```

Disabled-password ei toiminut, vaan järjestelmä kehotti luomaan salasanan.

6.2 Setting up Mastodon

Kun päästään Mastodonin asentamiseen, pitää käyttäjäksi muuttaa aiemmin luotu mastodon. Ensimmäiseksi pitää ladata viimeisin stabiili versio Mastodonista komennoilla:

```
git clone https://github.com/mastodon/mastodon.git live &&
cd live
```

```
git checkout $(git tag -l | grep '^v[0-9.]*$' | sort -V |
tail -n 1)
```

Git clone-komentorivistä tuli virheilmoitus:

fatal: could not create work tree dir 'live': Permission denied

Tätä lähdettiin ratkaisemaan StackOverflow-sivustolta. Vastaus ongelmaan oli, että kyseessä on käyttäjän luvasta asioiden asentamiseen serverille, ja että tämän voisi ratkaista komennolla

```
sudo chown -R $USER /var/www
```

USER tilalle vaihdettiin mastodon, mutta tuli virheilmoitus ’ chown: missing operand after ‘/var/www’’

Ratkaisu tähän löytyi Unix:StackExchange-sivustolta:

```
chown -R root:mastodon /var/www
```

Tämäkään ei lopulta ratkaissut ongelmaa, vaikka näyttikin menevän läpi. Kyse oli sittenkin vain siitä, että git clone-komennon edessä ei ollut sudo. Tämän jälkeen komento

```
git checkout $(git tag -l | grep '^v[0-9.]*$' | sort -
V | tail -n 1)
```

ei mennyt läpi ja terminaali antoi virheilmoitukseksi:

```
not a git repository (or any of the parent directories):
.git
```

Ongelma ratkaistiin ensin tarkastamalla eri käyttäjien luvat komennolla:

```
ls -lha
```

Jonka perään tarkastettiin, millä käyttäjällä on luvat kansiossa live oleviin tiedostoihin:

```
ls- la live
```

Käyttäjänä kaikissa oli mastodon, joten seuraavaksi siirryttiin kansioon live cd live- komennolla ja jonka jälkeen ajettiin uudelleen:

```
git checkout $(git tag -l | grep '^v[0-9.]*$' | sort -V | tail -n 1)
```

Tässä vaiheessa komento meni läpi ja päästiin seuraavan vaiheeseen, joka on Rubyn asentaminen.

6.3. Rubyn asentaminen

Mastodonin asennusohjeiden mukaan tässä käytetään rbenvtä Rubyn version hallintaan.

```
git clone https://github.com/rbenv/rbenv.git ~/.rbenv
```

Koska rbenv oli asennettu vahingossa aiemmin sekaannuksen vuoksi, antoi terminaali virheeksi:

```
fatal: destination path '/home/mastodon/.rbenv' already exists and is not an empty directory.
```

Oletettiin, että tämän voi jättää huomiotta ja siirryttiin seuraaviin komentoihin, jotka menivät kaikki läpi:

```
echo 'export PATH="$HOME/.rbenv/bin:$PATH"' >> ~/.bashrc
echo 'eval "$(rbenv init -)"' >> ~/.bashrc
exec bash
```

Tässä vaiheessa piti kloonata repo Githubista, ruby-build. Kun git clone komento

```
git clone https://github.com/rbenv/ruby-build.git "$(rbenv root)"/plugins/ruby-build
```

tuli jälleen varoitus, että path on jo olemassa. Tässäkin todennäköisesti oli kyse siitä, että tämä vaihe oli vahingossa tehty aiemmin ja siispä kokeiltiin seuraavaa komentoa:

```
RUBY_CONFIGURE_OPTS=--with-jemalloc rbenv install
```

joka meni läpi ja asensi Rubyn viimeisimmän version. Viimeiseksi asennettiin loput Ruby. ja Javascript-riippuvuudet. Komentoja oli tässä vaiheessa neljä, joista kaksi ensimmäistä meni läpi:

```
bundle config deployment 'true'
bundle config without 'development test'
```

Kolmannen komennon kanssa oli ongelmia.

```
bundle install -j$(getconf _NPROCESSORS_ONLN)
```

Virheilmoituksessa sanottiin seuraavaa:

```
The deployment setting requires a lockfile. Please make sure
you have checked
your Gemfile.lock into version control before deploying.
```

Tämän selvittämiseen meni aikaa. Yritettiin ensin löytää Gemfile olettaen, että tämä auttaisi asiaa.

Ratkaisua etsiessä selvisi, että sama ongelma on ollut muillakin ja usea vastaus kertoi, että tämä Gemfile.lock on luotu automaattisesti komennolla bundle install. Gemfile.lock on tärkeä, koska siinä sijaitsevat kaikki gemit, jotka kulloisenakin hetkenä on asennettu. Kokeiltiin päivittää bundle käyttämällä:

```
bundle update
```

Tästä tuli ilmoitus, että

```
Your bundle only supports platforms [] but your local
platform is x86_64-linux.
Add the current platform to the lockfile with
bundle lock --add-platform x86_64-linux` and try again.
```

Tämän ilmoituksen mallin mukaan ajettiin komento:

```
bundle lock --add-platform x86_64-linux
```

Tästä saatiin ilmoitus, että lockfile on luotu kansioon /home/mastodon/Gemfile.lock. Tämän jälkeen päästiin vaiheeseen neljä riippuvuuksien asentamisessa, joka oli:

```
bundle install -j$(getconf _NPROCESSORS_ONLN)
```

Viimeinen komento `yarn install` antoi viestin:

```
Corepack is about to download
https://registry.yarnpkg.com/yarn/-/yarn-1.22.22
```

Ohjelma kysyi tähän perään, että saadaanko lupa asentaa, johon annettiin hyväksyntä kirjoittamalla Yes.

6.3.1 Konfiguraation generointi

Ensin piti ajaa komento:

```
RAILS_ENV=production bin/rails mastodon:setup
```

Tästä tuli virheilmoitus bash: bin/rails No Such Directory. Jonkin aikaa asiaa tutkiessa kävi selville, että Ruby ei ollut asentunut aiemmin kunnolla. Asennusta yritettiin uudestaan git clone-komennolla, mutta se ei onnistunut, vaan tuli lista asioita, joita voisi lisätä komennon perään. Huomattiin, että oltiin poistuttu live-kansiosta ja kun palattiin sinne, ajettiin:

```
RUBY_CONFIGURE_OPTS=--with-jemalloc rbenv install
```

Tämä meni tällä kertaa kunnolla läpi ja kokeiltiin seuraavaa komentoa:

```
RAILS_ENV=production bin/rails mastodon:setup
```

Järjestelmä pyysi ajamaan `bundle install` uudelleen ja kun tämä oli tehty, onnistui edellisen komennon ajaminen. Tämä loi konfiguraatio-tiedoston, ajoi esikompilaation ja loi tietokantarakenteen. Konfiguraatiotiedosto tallentui .env.production nimellä. Sitä voi tarkastella ja editoida omien tarpeiden mukaan. Tämän osion päätteeksi voidaan vaihtaa käyttäjä mastodonista rootiksi komennolla exit.

6.3.2. SSL-sertifikaatin hankinta

Mastodonin asennusohjeet kehottavat käyttämään Let's Encryptin ilmaista SSL-sertikaattia. Let's Encrypt on ilmainen, automatisoitu ja avoin sertifikaatti, jota hallinnoi ISRG (Internet Security Research Group. Tämä ajetaan komennolla:

```
certbot certonly --nginx -d example.com
```

Tämä ei toiminut vaan saatiin virheilmoitus `'some challenges have failed'.` Asiaa alettiin selvittää ja mahdollisia syitä virheilmoitukselle löytyi lukuisia eri sivustoilta. Ensimmäinen selitys oli, että A ja AAAA merkinnät puuttuivat. Asia tarkistettiin letsdebug.net sivulta ja se käytti http-01 testiä ja tämä varmisti asian. Huomaamatta jäi, että portti 80 olisi pitänyt olla auki. Tässä vaiheessa kuitenkin luotiin A ja AAAA DNS asetuksiin ja yritettiin uudestaan. Tällä kertaa virheilmoituksena tuli timeout during connect. Selityksenä oli ongelma palomuurissa, yksityiskohta joka jäi huomaamatta. Zoner.fi:stä tarkistettiin A ja AAAA asetuksiin liittyen, että pitääkö nämä luoda heidän palveluunsa. Asiakaspalvelusta vastattiin, että koska liikenne halutaan ohjata fanttila.fi:hin Hetzner palvelimessa, niin ne pitää luoda sinne. Koska tämä oli tehty, ajettiin komento uudestaan, mutta tuloksena oli sama virheilmoitusi. Löytyi ohje, että todennäköisesti Nginx ei ole asentunut kunnolla ja se kannattaa poistaa ja asentaa uudelleen. Ensin poisto:

```
sudo apt-get purge nginx nginx-common
```

Haluttiin varmistaa, että se asentuu oikein, seurattiin ohjeita Hetznerin sivuilta. Ensimmäinen askel oli:

```
sudo apt install nginx
```

Tämä asensi Nginxin ja seuraavaksi piti päivitää palomuuri. Se aloitettiin listaamalla kaikki saatavilla olevat applikaatiot:

```
sudo ufw app list
```

Komento tulosti listan:

```
Nginx Full
Nginx HTTP
Nginx HTTPS
OpenSSH
```

Nginx HTTP sallii ainoastaan HTTP liikenteen ja avaa tähän portin. Nginx HTTPS sallii https-liikenteen ja sekin avaa itselleen sopivan portin. Nginx full sallii molemmat liikenteet ja avaa näitä varten portit 80 ja 443. Ohjeet sanoivat, että kannattaa sallia vain kaikkein tiukimman version tuotannnossa, mutta koska tässä vaiheessa vielä testataan asioita eikä SSL:ää ole säädetty, valittiin vaihtoehdoista Nginx http. Jotta tämä saatiin käyttöön, ajettiin komento:

```
sudo ufw allow 'Nginx HTTP'
```

Muutokset varmistettiin komennolla:

```
sudo ufw status
```

Joka tulosti:

```
Status: active
To                         Action      From
```

```
OpenSSH                    ALLOW          Anywhere
Nginx HTTP                 ALLOW          Anywhere
Open SSH (v6)              ALLOW          Anywhere
Nginx HTTP (v6)            ALLOW          Anywhere
```

Nyt palomuurin pitäisi sallia HTTP-yhteydet Nginxään ja se tarkistettiin komennolla:

```
systemctl status nginx
```

Tämä toimi ja näkyviin tuli seuraava:

```
Nginx.service - A high performance web server and a reverse
proxy server
     Loaded:   loaded   (/lib/systemd/system/nginx.service;
enabled; vendor preset: enabled)
     Active: active (running) since Sat 2021-08-21 17:54:37
CEST; 21min ago
       Docs: man:nginx(8)
   Main PID: 6370 (nginx)
      Tasks: 2 (limit: 2280)
     Memory: 4.1M
     CGroup: /system.slice/nginx.service
             ├─6370 nginx: master process /usr/sbin/nginx -
g daemon on; master_process on;
             └─6371 nginx: worker process
```

Statuksen pystyi myös verfioimaan menemällä osoitteeseen http://49.12.14.108 , joka on tulevan mastodon-serverin osoite ja näkyviin olisi pitänyt tulla (kuva 12) seuraava teksti:

Kuva 12. Welcome to nginx! Hetzner.com 2024.

Koska näin ei tapahtunut, käytiin tarkistamassa palomuuri, ja varmistamassa että portti 80 on auki. Koska se ei ollut, se lisättiin sinne ja saatiin kuvan näkymä esiin.

Seuraava vaihe oli sisältökansion luominen, joka tehtiin komennolla:

```
sudo mkdir -p /var/www/fanttila.fi/html
```

Jotta tästä vaiheesta pääsi eteenpäin, piti tarkistaa, että kansiossa on oikeat sallinnat. Tähän piti käyttää komentoa:

```
sudo chown -R $USER:$USER /var/www/example.com/html
```

Komento ymmärrettiin väärin ja USERin paikalle vaihdettiin mastodon joka myöhemmin ilmeni ongelmana, sillä tässä vaiheessa komentorivi ei hälyttänyt, että jotain olisi väärin. Siirryttiin siis seuraavaan kohtaan tutoriaalissa.

Nyt piti luoda yksinkertainen index.html tiedosto ja se luotiin nano-komennolla:

```
sudo nano /var/www/example.com/html/index.html
```

Tämä avasi editorin ja sinne lisättiin html-koodi mallin mukaan. Koska kyseessä oli testisivu, koettiin että tässä vaiheessa ei tarvitse mitään omaa koodia.

```
<!doctype html>
```

```
<html>
  <head>
    <title>This is our test website</title>
  </head>
  <body>
    <p>Hello, mastodon!</p>
  </body>
</html>
```

Seuravaksi piti kertoaa Nginxlle, että mihin osoittaa pyynnöt ja se luotiin jälleen kerran nanolla.

```
sudo nano /etc/nginx/sites-available/fanttila.fi
```

Tähän tiedostoon tehtiin serveri-konfigurointi, joka oli:

```
server {
    listen 80;
    server_name fanttila.fi;
    root /var/www/fanttila.fi/html;
    index index.html;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

Jotta tämä serveriblokki saatiin aktivoitua, piti luoda linkki kansioon /etx/nginx/enabled/, joka osoitti tiedostoon /etx/nginx/sites-available/fanttila.fi

Tämän jälkeen piti muokata nginx konfiguraatio, joka tehtiin nanolla:

```
sudo nano /etc/nginx/nginx.conf
```

Siellä piti aktivoida lauseke `server_names_hash_bucket_size  64;` Nyt oli mahdollista testata oliko Nginx konfiguraatio oikein ja se varmistui, kun näkyviin tuli:

```
the configuration file /etc/nginx/nginx.conf syntax is
ok
nginx: configuration file /etc/nginx/nginx.conf test is
successful
```

Viimeisenä ladattiin Nginx uudestaan komennolla `sudo systemctl reload nginx` jonka olisi pitänyt tuoda näkyviin aiemmin luotu index.html osoitteessa fanttila.fi. Näin ei käynyt vaan palattiin takaisin alkuun ja käytiin ohjeet läpi kohta kohdalta. Siinä huomattiin virhe komennossa

```
sudo chown -R $USER:$USER /var/www/example.com/html
```

Se korjattiin ja selaimeen tuli näkyviin Hello, mastodon.

6.4 Viimeiset vaiheet ja ongelmat

Kun asentamista jatkettiin seuraavana päivänä, oli tapahtunut jotain serverin päässä eikä enää pystynyt luomaan SSH-yhteyttä, koska salasana ei toiminut. Kokeiltiin varmuudeksi salasanan eri muunnelmia, mutta mikään niistä ei toiminut. Koska root-salasanakaan ei toiminut, piti lähteä selvittämään miten uuden salasanan saisi luotua.

Ensin kokeiltiin Hetzner-konsolin rescue-osiosta Reset Root Password-painiketta ja se loi uuden salasanan, joka ei kuitenkaan toiminut SSH-yhteyttä luotaessa. Salasanan generoimista yritettiin uudelleen, mutta nyt konsoli antoi virheilmoituksen, ettei generointi onnistunut. Oltiin siis jumissa. Aloitettiin metsästämään miten root-salasanan saisi luotua uudelleen ja lopulta otettiin yhteys Hetzerin tukeen. Sieltä tuli seuraavanlainen vastaus:

”…On mahdollista säätää SSH-avaimia ’Käynnistä rescue & power cycle’ sen sijaan, että käyttäisi reset rootpasswordia. Jotta tämä onnistuu, täytyy tehdä näin:

1. Mene serverin Rescue-välilehdelle ja paina ’Enable rescue & power cycle’. Tämän jälkeen serveri boottaa itsensä rescue-systeemiin

Tällä sai konsolissa näkymään väliaikaiset tunnukset ja salasanan SSH-yhteyttä varten

2. Muodosta SSH-yhteys serveriin käyttämällä rescue-tunnuksia
3. Aja seuraavat komennot:

- Mount /dev/sda1 mnt
- Chroot-prepare /mnt
- Chroot /mnt ”

Nämä eivät toimineet, joten tarkistettiin lsblk-komennolla root@rescue -käyttäjänä mitä osia siellä on. Hetznerin dokumentit sanoivat, että jos käytössä ei ole RAID-elementtejä, näkymän (kuva 13) pitäisi olla tälläinen:

```
root@rescue ~ # lsblk
NAME    MAJ:MIN RM   SIZE RO TYPE  MOUNTPOINT
loop0     7:0    0     4G  1 loop
sda       8:0    0 447.1G  0 disk
├─sda1    8:1    0     4G  0 part
├─sda2    8:2    0   512M  0 part
└─sda3    8:3    0 442.6G  0 part
sdb       8:16   0 447.1G  0 disk
└─sdb1    8:17   0   446G  0 part
```

Kuva 13. Käytössä olevat SDA:t. Hetzner.com.

Eräät ohjeet kertoivat, että aiemmin mainitusta komentolistassa käytetään sda:ta, joka on viimeinen tai toiseksi viimeinen, käytettiin komentoa:

```
Mount /dev/sda15 mnt
```

Tästä seurasi virheilmoitus:

```
’bin/bash’: No such file or directory
```

Tästä pääteltiin, että kyseinen sda15 oli väärä ja kokeiltiin sitten sda14:lla ja kun sekään ei toiminut, yritettiin vielä sda1:llä, mikä sitten toimi. Tämän jälkeen pystyttiin ajamaan chroot-komennot.

Vimeisenä komentona ajettiin psswd, jolloin vaihdettiin root-käyttäjän salasana ja poistuttiin tästä käyttäjästä.

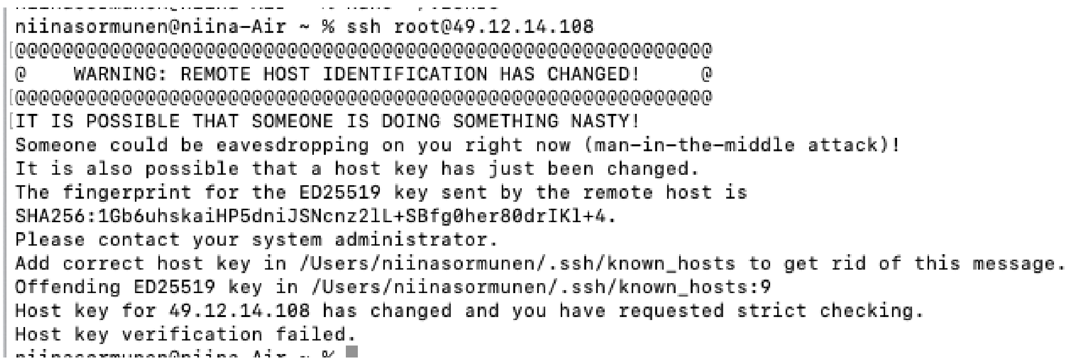
Nyt pystyttiin uudelleen käynnistämään serveri komennolla:

```
reboot
```

Terminaali palasi tietokoneen pääkäyttäjäksi ja nyt päästiin siis yrittämään tavallisena root-käyttäjänä ssh-yhteyden luomista.

6.5 Hostkey-ongelma

Kun yhteyttä lähdettiin yrittämään root-käyttäjänä uudella salasanalla, tuli (kuva 14) seuraavanlainen varoitus:



```
niinasormunen@niina-Air ~ % ssh root@49.12.14.108
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the ED25519 key sent by the remote host is
SHA256:1Gb6uhskaiHP5dniJSNcnz2lL+SBfg0her80drIKl+4.
Please contact your system administrator.
Add correct host key in /Users/niinasormunen/.ssh/known_hosts to get rid of this message.
Offending ED25519 key in /Users/niinasormunen/.ssh/known_hosts:9
Host key for 49.12.14.108 has changed and you have requested strict checking.
Host key verification failed.
```

Kuva 14. Warning: Remote host identification has changed!

Lähdin selvittämään asiaa antamalla komennon:

```
Ssh-keygen -f “/Users/niinasormunen/.ssh/known_hosts” -R “49.12.14.108”
```

Tällä sai näkyviin virheelliset rivit, joita oli loppujen lopuksi viisi riviä. Ensin kokeiltiin näiden rivien kommentoimista, mutta kun virheilmoitukset jatkuivat, poistettiin known_hosts-tiedostosta kaikki hostkeyt. Tämän jälkeen yllättävänä seurauksena alkoivat alkuperäiset salasanat toimia.

6.6 502 bad gateway

Tässä vaiheessa oli lähes kaikki asennusvaiheet käyty läpi. Oli enää muutama komento jäljellä ja fanttila.fi olisi toiminnassa. Ensin ajettiin komennot:

```
cp /home/mastodon/live/dist/nginx.conf /etc/nginx/sites-available/mastodon
ln -s /etc/nginx/sites-available/mastodon /etc/nginx/sites-enabled/mastodon
rm /etc/nginx/sites-enabled/default
```

Tässä vaiheessa ei tullut mitään virheilmoitusta, joten jatkettiin eteenpäin. Sitten editoitiin tiedostoa: /etc/nginx/sites-available/mastodon
Sieltä piti vaihtaa example.com kohdille fanttila.fi ja sen lisäksi kommentoida takaisin rivit

```
ssl_certificate
/etc/letsencrypt/live/example.com/fullchain.pem;
ssl_certificate_key
/etc/letsencrypt/live/example.com/privkey.pem;;
```

Kaikki näytti menevän hyvin, joten seuraavaksi annettiin mastodon-käyttäjille lupa liikkua omassa kotihakemistossaan:

```
chmod o+x /home/mastodon
```

Jotta kaikki muutokset tulisivat voimaan, ajettiin komento:

```
systemctl restart nginx
```

Nyt näkyvissä olisi pitänyt olla Mastodonin oma error-sivu, mutta näkymä oli ihan tyhjä. Tämä jätettiin huomioimatta ja jatkettiin eteenpäin. Ensin piti kopioida systemd palvelumallit Mastodon hakemistosta:

```
cp/home/mastodon/live/dist/mastodon-*.service
/etc/systemd/system/
```

Viimeisimpänä ajettiin:

```
systemctl daemon-reload
systemctl enable --now mastodon-web mastodon-sidekiq
mastodon-streaming
```

Nämä näyttivät menevän läpi, mutta kun avattiin fanttila.fi näkyviin tuli 502 Bad Gateway Error.

6.6.1 502 bad gateway-ongelman ratkaisu

Kun fanttial.f i-mastodoninstanssi ei käynnistynytkään, lähdettiin metsästämään ongelmaa ja ensimmäinen idea oli, että virhe ehkä löytyy niginxin error tsi access-logista. Käytettiin komentoa:

```
tail /var/log/nginx/error.log
```

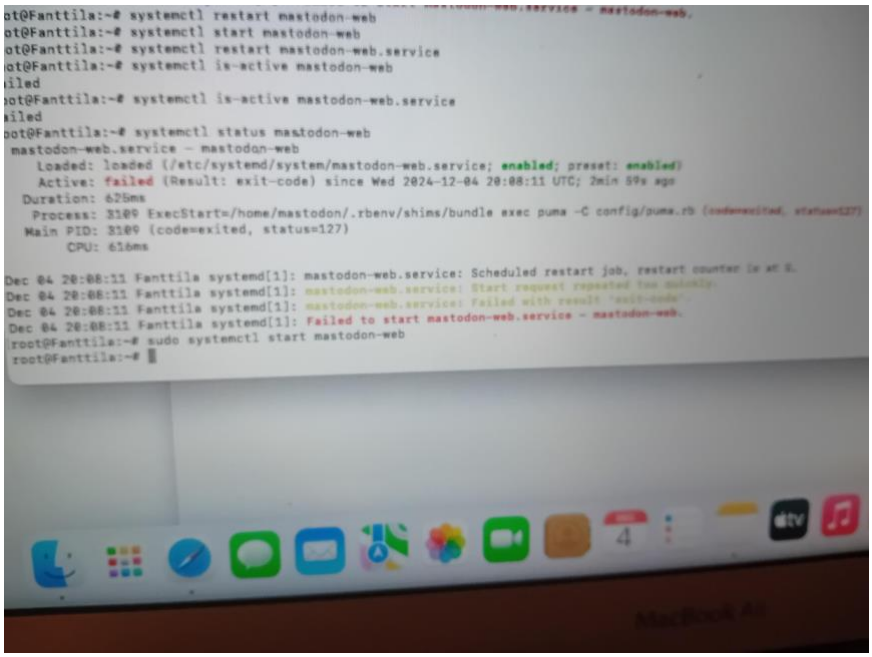
Saatiin virheilmoitus, että error.log connection refused while connecting to upstream. Katsottiin mastodon-web status komennolla:

```
systemctl status mastodon-web
```

Virheilmoituksena tuli failed to start ja yritettiin komentoa:

```
systemctl restart mastodon-web
```

Fanttila.fi valitti edelleen 502-virheestä ja error.logissa toistuivat samat virheet kuin aiemminkin. Tarkastettiin uudelleen mastodon-web status (kuva 15) ja näkymä oli tällainen:

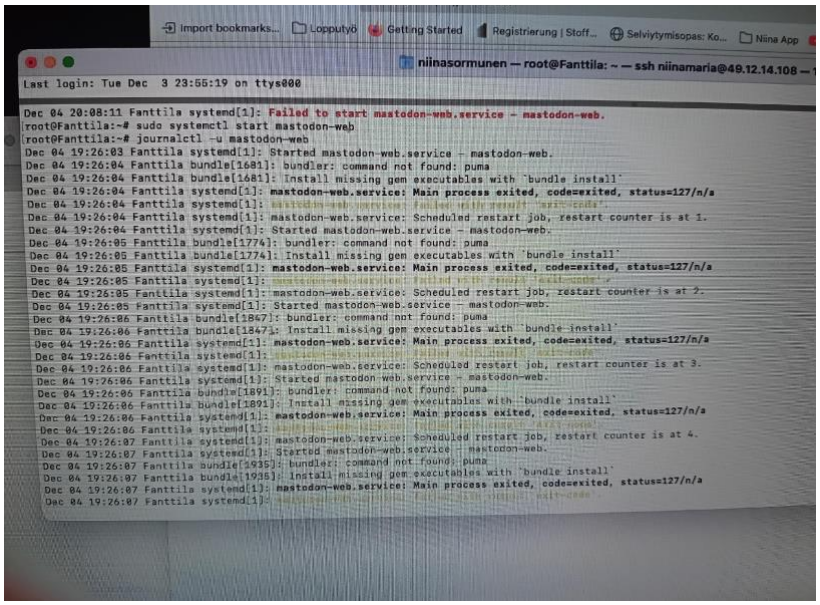


Kuva 15. Mastodon-web status

Tarkasteltiin mastodon-webiä hieman tarkemmin komennolla:

```
Journalctl -u mastodon-web
```

Näkuviin tuli (kuva 16) logi:



Kuva 16. Mastodon-web log.

Tässä vaiheessa tehtiin bundle install ja terminaali antoi varoituksen puuttuvasta Gem. -tiedostosta. Kokeiltiin sitten bundle installia käyttäjänä mastodon ja se näytti toimivan. Palattiin takaisin root-

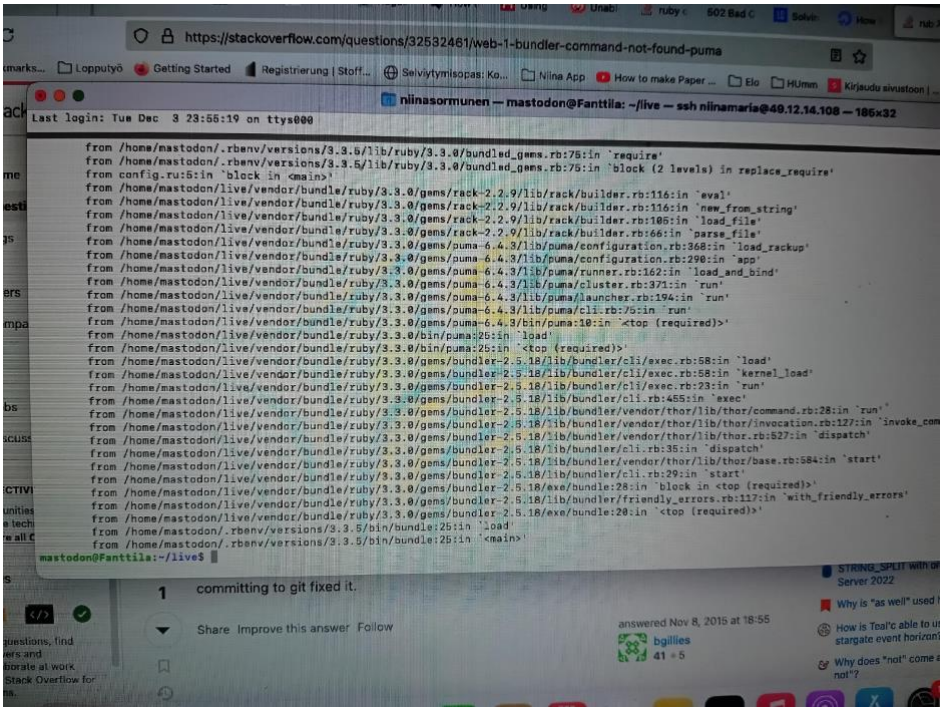
käyttäjäksi ja ajettiin systemctl start mastodon-web. Kun tuli jälleen kerran 502-virhe, katsottiin journalctl:llä mistä on kyse ja ongelma oli puma-gem. Vaihdettiin takaisin mastodon-käyttäjäksi, siirryttiin live-kansioon ja ajettiin komento:

```
RAILS_ENV=production/home/mastodob/.rbenv/shims/bundle
install
```

Sen perään komento:

```
RAILS_ENV=production/home/mastodon/.rbenv/shims/bundle
exec puma -C config/puma.rb
```

Seuraavaksi tarkistettiin mitä bundle/puma-prosesseja oli käynnissä (kuva 17) ja terminaali näytti tällaista:



Kuva 17. Bundle/puma prosessit

Tässä logissa näkyi virhe “key not found ‘otp_secret’”, mikä tarkoitti sitä, että asennuksessa oli jäänyt yksi vaihe tekemättä mastodon-käyttäjänä:

```
RAILS_ENV=pruduction bin/rails mastodon:setup
```

Tämän ajaminen käynnisti asetusten säätämisen, jossa vastattiin kysymyksiin näin:

Are you using Docker to run Mastodon? No

Postresql host: /var/run/postgreqsl? Enter

Port 5432? Enter

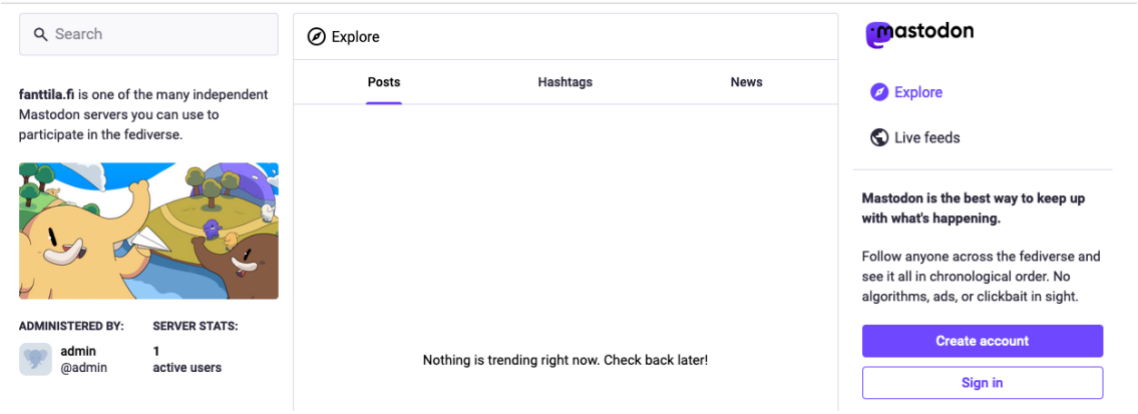
Do you want to store uploaded files in cloud? No (koska käytössä jo Hetznerin pilvipalvelu)

Do you want to send emails from localhost? Yes (asetettiin notifications@fanttila.fi)

Will be written in .en.production. Save configuration? Yes

Database schema must be loaded. If the database already exists, this will erase its contents. Prepare database now=? Tässä vastattiin “No”, mutta se piti myöhemmin korjata ja käydä tämä vaihe uudestaan ja sanoa “Yes”.

Tämän jälkeen ajettiin mastodon-käyttäjän live-kansiossa Yarn install ja fanttila.fi -instanssi (kuva 18) alkoi toimia.



Kuva 18. Fanttila-instanssin etusivu. Fanttila.fi

7 FANTTILA-INSTANSSI

7.1 Instanssin säätäminen

Kun instanssi on luotu, se federoituu eli yhdistyy muiden instanssien kanssa automaattisesti. On kuitenkin monia toimintoja taustalla, jotka kannattaa säätää käytettävyyden ja tietoturvallisuuden takia. Tämän projektin raameissa ne jätetään myöhäisempään vaiheeseen, koska ne eivät sinällään ole heti välttämättömiä.

Asennusohjeissa on kohta Setting up your new instance. Siinä neuvotaan, miten luodaan instanssille pääkäyttäjä. Se tehdään SSH-yhteydessä mastodon-käyttäjänä ja siinä käytetään seuraavaa komentoa:

```
RAILS_ENV=production bin/tootctl accounts create \
  alice \
  --email alice@fanttila.fi \
  --confirmed \
  --role Owner
```

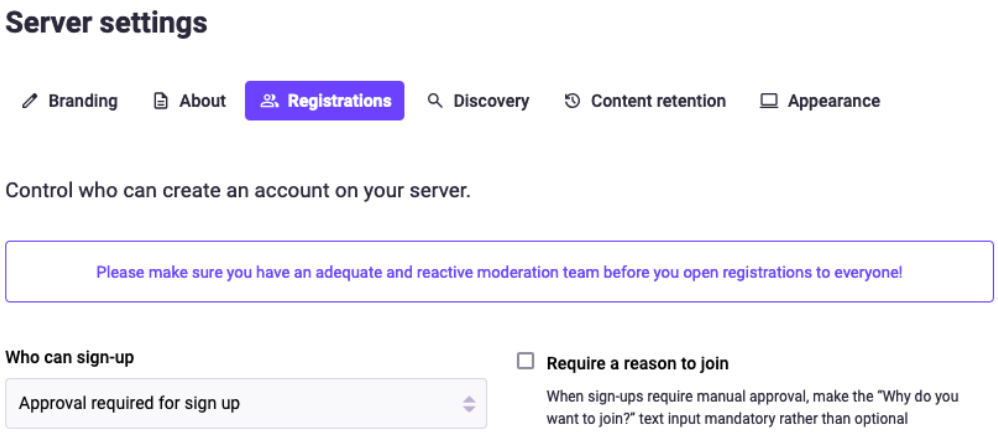
Tämä loi terminaaliin näkyviin kertakäyttöisen salasanan, jolla pystyi kirjautua instanssiin. Siellä kuitenkin luki, että käyttäjä pitää hyväksyä, ennen kuin pääsee muokkaamaan instanssiasetuksia. Käyttäjän hyväksyntä tehtiin komennolla:

```
RAILS_ENV=production bin/tootctl accounts modify alice –approve
```

Tämän komennon jälkeen avautuivat täydet pääkäyttäjäoikeudet ja sen avulla muokattiin hieman asetuksia admin-näkymässä.

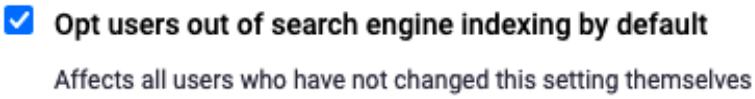
Hallinta-osiossa on serveriasetukset, joita on viisi. Brändäys, jossa voi antaa instanssille oman nimen, kontaktinimen ja sähköpostin, sekä kuvailun instanssista. About-osiossa voi antaa lisäinformaatiota instanssin toiminnasta esimerkiksi yksityisyysasetuksista. Rekisteröinnissä voi ja

pitää muuttaa asetukset sellaisiksi, että muutkin pääsevät instanssin käyttäjäksi. Reunaehdoksi asetettiin (kuva 19), että jokainen rekisteröinti pitää manuaalisesti hyväksyä.



Kuva 19. Instassin admi-asetukset. Fanttila.fi.

Discovery-osion, jossa voi esimerkiksi määritellä, mitä fanttilan ”etusivulla” on näkyvillä, määritellään myös miten instanssin käyttäjien tootit näkyvät hakukoneissa. Tähän valittiin opt-out (kuva 20) niille, jotka eivät itse ole muuttaneet tässä asiassa asetuksiaan.



Kuva 20. Indeksointi-asetusvalikko. Fanttila.fi.

Toiseksi viimeinen osio on cache ja käyttäjien tuottaman sisällön säilyttäminen. Ensimmäisenä kysytään miten kauan halutaan säilyttää mediasisältöä instanssin ulkopuolisilta käyttäjiltä. Kun tämä luku on positiivinen, sisältö poistetaan määritellyn ajan kuluttua, tässä tapauksessa 30 päivän jälkeen. Jos joku hakee sisällön sen jälkeen kun se on poistettu, se ladataan serverille uudestaan, jos se vain on saatavilla alkuperäisessä lähteessä. Toinen asetus on se, että kauanko säilytetään instanssin käyttäjien sisältöä. Käyttäjillä on mahdollisuus arkistoida omaa sisältöä myöhempää latausta varten. Jos tämän asetuksen luku on positiivinen, nämä arkistot poistetaan automaattisetsti serveriltä tietyn ajan kuluttua. Tässä valittiin luvuksi 30.

Viimeisimpänä on instanssin ulkonäön muokkaus, mihin voi lisätä omaa CSS-muotoilua.

7.2 Ensimmäisten käyttäjien liittyminen

Kun ilmoitettiin julkisesti, että fanttila.fi on toiminnassa ja että sinne voi luoda profiilin, niin pääkäyttäjän toisen instanssin profiiliin tuli viesti, ettei vahvistussähköpostia fanttila.fi:stä ole tullut. Tässä vaiheessa ei tämän henkilön pyyntöä ollut vielä hyväksytty fanttila.fin hallintapaneelissa. Kun pyyntö hyväksyttiin, eikä vahvistussähköpostia siitäkään huolimatta tullut, niin lähdettiin selvittämään asiaa Sidekiqin kautta. Sieltä löytyi virheilmoitus ’connection refused localhost port:25’.

Ongelmaksi huomattiin, että ensinnäkin SMTP-palveluksi valitussa Mailjetissä, ei ollut SPF ja DKIM kirjaukset aktivoituna. Tämä tehtiin tekemällä kaksi TXT-tietuetta Hetznerin DNS-osiossa. Ensimmäinen, SPFää varten tehty TXT täytettiin host-nimellä fanttila.fi. ja arvolla, joka oli ” v=spf1 include:spf.mailjet.com ?all”. Kun tätä tietuetta yritettiin tehdä ensimmäisen kerran, jäi huomaamatta, että fanttila.fi perään piti lisätä piste ja arvo laittaa hipsujen sisään. DKIM kirjauksessa host-nimeksi piti asettaa mailjet_domainkey.fanttila.fi ja value, joka alkoi k-rsalla.

Toinen ongelma oli, että active_maileria ei oltu vielä konfiguroitu. Tähän löytyi ohjeet Ruby on Rails oppaasta. Ensin piti generoida maileri. Tämä tehtiin mastodon-käyttäjän live-kansiossa komennolla:

```
$ bin/rails generate mailer User
```

Tämä loi Maileriin liittyvät luokat:

```
create  app/mailers/user_mailer.rb
invoke  erb
create    app/views/user_mailer
invoke  test_unit
create    test/mailers/user_mailer_test.rb
create    test/mailers/previews/user_mailer_preview.rb
```

Seuraavaksi muokattiin tiedostoa app/mailers/user_mailer.rb lisäämällä luokka UserMailer. Jokainen generoitu Mailer-luokka periytyy ApplicationMailerista.

```
class UserMailer < ApplicationMailer
end
```

ApplicationMailer puolestaan periytyy ActionMailer::Basesta, ja sitä voi käyttää sellaisten attribuuttien määrittelyyn, jotka ovat kaikille Mailereille yhteisiä.

```
class ApplicationMailer < ActionMailer::Base
    default from: "from@fanttila.fi"
    layout "mailer"
end
```

UserMailerilla, joka on app/mailers/user_mailer.rb tiedostossa, ei ole alunperin metodeja. Siksi nämä lisätään maileriin, joka lähettää spesifejä sähköposteja.

```
class UserMailer < ApplicationMailer
  default from: "notifications@fanttila.fi"

  def welcome_email
    @user = params[:user]
    @url  = "http://fanttila.fi/login"
     mail(to: @user.email, subject: "Tervetuloa
     Fanttilaan!")
  end
end
```

Seuraavaksi luotiin welcome_email toimintoa varten HTML-tiedosto, jossa on tervetuloa-viestin sisältö. Tämän tiedoston nimeksi piti laittaa welcome_email.html.erb, jonka piti sijaita app/views/user_mailer/- kansiossa. Viestiksi kirjoitettiin seuraava:

```
<h1>Tervetuloa Fanttilaan, <%= @user.name %></h1>
<p>Olet onnistautuneesti kirjautunut Fanttilaan ja
käyttäjänimesi on: <%= @user.login %>.<br>
</p>
```

```
<p>Kirjautuaksei sisään seuraa tätä linkkiä: <%= link_to
'login`, login_url %>.
</p>
<p>Kiitos, että tulit mukaan Fanttilaan!</p>
```

Seuraavaksi piti kutsua mailer-metodia, joka renderöi email-näkymän. Ensin piti luoda User scaffold:

```
$ bin/rails generate scaffold user name email login
$ bin/rails db:migrate
```

Tämän jälkeen piti muokata create-toimintoa UserControllerissa, jotta käyttäjälle voidaan lähettää sähköposti, kun hänen tilinsä on luotu.

```
class UsersController < ApplicationController
  def create
    @user = User.new(user_params)

    respond_to do |format|
      if @user.save
        UserMailer.with(user:
@user).welcome_email.deliver_later
        format.html { redirect_to user_url(@user),
notice: "User was successfully created." }
        format.json { render :show, status: :created,
location: @user }
      else
        format.html { render :new, status:
:unprocessable_entity }
        format.json { render json: @user.errors,
status: :unprocessable_entity }
      end
    end
 end
```

On myös mahdollista asettaa sähköposti lähtemään heti, mutta tällä kertaa päätettiin pitäytyä mallin mukaisesti deliver.late-moodissa. Sen lisäksi voi esimerkiksi määritellä kuvien ja liitteiden lisäämisen sähköpostiin. Näin ei kuitenkaan tällä kerralla tehty.

Seuraavaksi generoitiin URL:t action mailer näkymissä. Tämä tehtiin config.application.rb -tiedostossa asettamalla sinne host-arvo:

```
config.action_mailer.default_url_options = { host: "fanttila.fi" }
```

Tässä välissä olisi voinut muokata sähköpostin lähetysasetuksia, kuten niiden ajoittamista, mutta tässä vaiheessa vaikutti siltä, että riittää, että mennään oletusasetuksilla. Seuraavaksi konfiguroitiin action_mailer config/environments/$RAILS_ENV.rb-tiedostossa. Sinne lisättii:

```
config.action_mailer.smtp_settings
```

Tämän alle lisättiin yksityiskohtaiset konfiguraatiot joinmastodon.org/amdin/config-ohjeen mukaisesti.

Tämän jälkeen törmättiin ongelmaan. Tässä vaiheessa olisi pitänyt pystyä esikatselemaan tervetuloviestiä lisäämällä luokka UserMailerPreview test/mailers/previews/-kansiossa tähän tapaan:

```
class UserMailerPreview < ActionMailer::Preview
  def welcome_email
    UserMailer.with(user: User.first).welcome_email
  end
end
```

Prosessin olisi kuulunut muodostaa test/mailers/previews/ -kansio, mutta yrityksistä huolimatta kansiota ei löytynyt. Tämän johdosta päätettiin, että fanttila.fi:in liittymään haluavat käyttäjät hyväksytään manuaalisesti komennolla:

```
RAILS_ENV=production bin/tootctl accounts modify userName –
approve
```

Fanttilaa kuitenkin halutaan jatkokehittää, joten sähköpostiongelma ratkaistaan opinnäytetyön raamien ulkopuolella.

8 ACTIVITYPUB

Mastodonin taustalla on ActivityPub-protokolla. Se on teknologia, jonka avulla eri sosiaalisen median alustat voivat kommunikoida niin, ettei joka alustalle tarvitse tehdä uutta profiilia, vaan yksi tehty profiili käy kaikkiin palveluihin tämän protokollan alla.

Spesifikaatio määrittelee ActivityPub:in kahdeksi, toisilleen sukua olevaksi ja vuorovaikutuksessa olevaksi protokollaksi. Kyseessä on sosiaalinen API (social API), joka sallii asiakkaan (client) toimivan käyttäjän puolesta.

Serveriltä serverille protokolla, federoitu API, sallii jakamisen eri toimijoiden välillä eri servereillä sitoen ne samaan sosiaaliseen verkostoon.

Tämä spesifikaatio tarkoittaa sitä, että vaikka voi ottaa käyttöön vain jommankumman, ei ole kuin pieni vaiva tehdä molemmat. Tässä tulee mukaan kolmen kohdan sääntöjenmukaisuus:

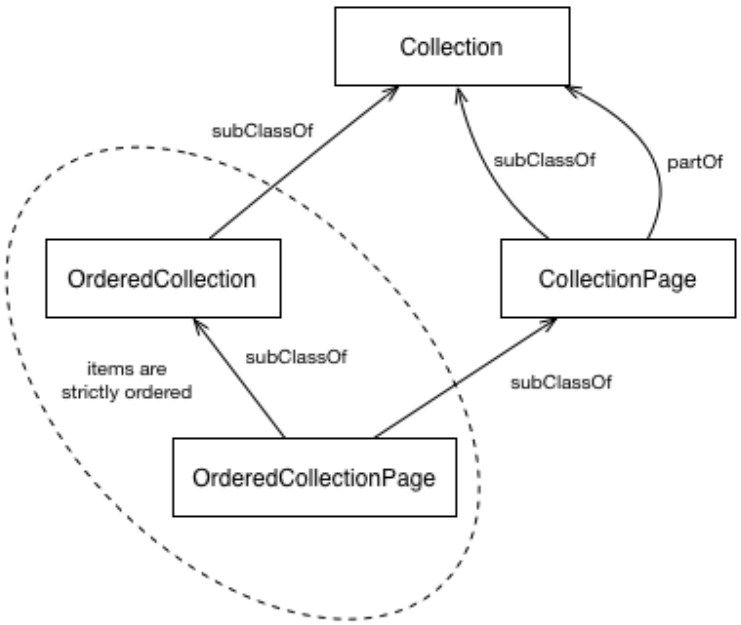
1. Client- Tämä nimitys koskee kaikkia implementaatioita, jotka koskevat asiakas-palvelin -protokollaa.
2. Palvelin- Tämä nimitys koskee kaikkia palvelin-osioita asiakas-palvelin -protokollassa.
3. Federoitu palvelin- Tämä nimitys koskee kaikkia federaatioprotokolla-implementaatioita.

ActivityPub:illa on myös oma sanasto, jonka voi jakaa kahteen osioon; ydintyypit ja laajennetut tyypit. Ydintyypit kattavat Activityn eli Toiminnon yleisluontoisen rakenteen ja Laajennetut tyypit ovat ominaisuuksia, jotka kattavat tietyn tyyppiset toiminnot ja tuotokset.

8.1 Ydintyypit

Ydintyypit:
Object- Toimii perustana kaikille objekteille, mukaan lukien muut ydintyypit.
Link- On epäsuora varmennettu referenssi URL:n identifioimaan resurssiin.
Activity- Objektin alatyyppi, joka kuvaa jonkinlaista toimintoa joka saattaa tapahtua, tapahtuu tai on jo tapahtunut.

IntransitiveActivity- Instansseja, jotka ovat Activityn alatyyppejä ja jotka edustavat intransitiivisia toimintoja.
Collection- on Objektin alatyyppi, joka edustaa järjestettyjä tai ei-järjestettyjä Object tai Link instansseja.
OrderedCollection- on Kokoelma-alatyyppi, jossa loogisen kokoelman jäsenet ovat aina järjestyksessä.
CollectionPage- käytetään edustamaan selkeästi erottuvia kohteita Kokoelmassa.
OrderedCollectionPage- käytetään edustamaan järjestettyjä kohteiden alityyppejä, jotka sijaitsevat OrderdCollectionissa.



Kuva 21. Eri kokoelmien (collection) yhteydet toisiinsa. w3.org

8.2 Laajennetut tyypit

Laajennetut tyypit on jaettu kolmeen eri osa-alueeseen: Activity Types, Actor Types ja Object Types.

Activity-tyyppejä on 28 erilaista:

Accept, Add, Announce, Arrive, Block, Create, Delete, Dislike, Flag, Follow, Ignore, Invite, Join, Leave, Like, Listen, Move, Offer, Question, Reject, Read, Remove, TentativeReject, TentativeAccept, Travel, Undo, Update ja View.

Niistä yleisimmät ovat:
Create- käytetään, kun käyttäjä luo uutta sisältöä.
Update- viittaa siihen, kun olemassa olevaa objektia, kuten postaus, profiili tai mediatiedosto, on muokattu tai päivitetty.
Delete- käytetään, kun sisältö on poistettu verkostosta.
Follow- käytetään, kun käyttäjä haluaa seurata toista käyttäjää, periaatteessa toisen käyttäjän sisällön tilaus.
Unfollow- kun käyttäjä lopettaa toisen käyttäjän seuraamisen.

Actor Tyypit ovat objektityyppejä, jotka voivat suorittaa aktiviteetteja. Tämän ydintyypit ovat:
Application- kuvaa ohjelmistosovellusta.
Group- edustaa formaaleja tai epäformaaleja Actor-kollektiiveja.
Organisation- edustaa organisaatiota.
Person- edustaa yksilöä.
Service- edustaa minkä tahansa tyyppistä palvelua.

Objekti ja Linkkityypit
Kaikki objektityypit perivät pohjaobjektin ominaisuudet. Linkkityyppit puolestaan perivät pohjalinkin ominaisuudet. Jotkut spesifit objektityypit ovat alatyyppejä tai spesiaaleja versioita yleisluontoisista objektityypeistä. Esimerkiksi Like-tyyppi on Activity-tyypin spesialisaatio.

Object Types:
Article- Edustaa minkä tahansa tyyppistä, moniosaista tekstiä.
Audio- edustaa minkä tahansa tyyppistä äänitiedostoa.
Document- Edustaa minkä tahansa tyyppistä dokumenttia.
Event- edustaa minkä tahansa tyyppistä tapahtumaa.
Image- mitkä tahansa kuvatiedostot.
Note- edustaa lyhyttä tekstiä, tyypillisesti vähemmän kuin yksi kappale.
Page- edustaa verkkosivua.
Place- edustaa loogista tai fyysistä lokaatiota.

Profile- on sisältöobjekti, joka kuvaa toista Objektia. Käytetään yleensä kuvaamaan Actor Tyyppejä.
Relationship- kuvaa kahden yksilön välistä yhteyttä. Subject ja object ominaisuuksia käytetään yhteydessä olevien yksilöiden suhteen tunnistamiseen.
Tombstone- edustaa sisältöobjektia, joka on deletoitu. Voidaan käyttää Kokoelmassa osoittamaan, että paikalla on ollut objekti, mutta se on poistettu.
Video- edustaa minkä tahansa tyyppistä videodokumenttia.

Link Types:
Mention- spesifi linkki, joka edustaa @-mainintaa.

Acticitypub-protokollaa voi käyttää monen tyyppisissä sovelluksissa. Yksi niistä on Pixelfed, joka on Instagramin tapainen kuviin pohjautuva sosiaalisen median alusta, joka on pikkuhiljaa nousemassa taitelijoiden ja valokuvaajien suosioon, sillä siellä ei heidän työnsä jää piiloon algoritmien vuoksi. Tämä on yhteistä kaikille ActivityPubiin pohjatuvilla sovelluksilla eli postaukset näytetään kronologisessa järjestyksessä eikä tykkäyksien ja jakojen perusteella.

9 YHTEENVETO

Työn tarkoituksena oli luoda oma Mastodon-linuxpalvelin. Ensimmäisiä asioita, joita tuli selvittää, oli sopivan Ubuntu-pilvipalvelun löytäminen. Mastodonissa olevat seuraajat, jotka olivat olleet paljon tekemisissä palvelinten kanssa, suosittelivat Hetzneriä, johon sitten päädyttiin. Haasteita kohdattiin paljon, mihin osittain oli syynä prosessissa sattuneet virheet. Esimerkiksi ensin luotiin Fanttila-palvelin, vaikka aluksi olisi pitänyt luoda SSH-avaimet. Tämä korjattiin muutaman yrityksen kautta. SSH-salasanat unohdettiin useampaan otteeseen ja niiden uudelleen asettaminen oli työlästä ja tämän takia jouduttiin olemaan yhteydessä Hetznerin tukeen useamman kerran.

Sinällään Mastodon-asennusohjeet olivat selkeät, mutta huolimattomuus ohjeita seuratessa aiheutti ongelmia ja sen takia piti useaan kertaan palata taaksepäin prosessissa. Näitä ongelmia yritettiin pääosin ratkoa Google-hauilla ja viime kädessä kysymällä neuvoa sellaisilta mastodon-käyttäjiltä, joilla oli jo oma instanssi.

Itse Fanttila.fi instanssi saatiin toimimaan lukuun ottamatta sähköpostivahvistusta. Tämä vaatii vielä tarkempaa perehtymistä SMTP:hen. Kuitenkin tästä projektista jäi päälle innostus Fanttila.fi-mahdollisuuksista. Eräs seuraaja antoi kommentin, että söpöydellään Fanttila voisi olla se, mikä innostaa käyttäjiä Mastodonin pariin. Varsinkin, jos tehdään Fanttila-niminen applikaatio, jota kautta ihmiset pystyisivät luomaan oman käyttäjäprofiilin. Tulevaisuuden tavoitteena on toteuttaa tämä jossain vaiheessa.

Tiivistettynä koen, että projekti oli sopivan haastava ja se jätti innon palvelimien maailmaan. Ongelmanratkaisutaitoni kehittyivät huomattavasti ihan uudelle tasolle, joka lisäsi ammatillista itsevarmuutta jättäen olon, että minusta tulee tieto- ja viestintätekniikan ammattilainen.

LÄHTEET

ActivityPub.Rocks 2020. Introduction to ActivityPub. Hakupäivä 12.9.2024
https://socialhub.activitypub.rocks/t/introduction-to-activitypub/508

Atmosphere 2024. Protocol overview. 4.9.2024
https://atproto.com/guides/overview

Avg 2024. What is TCP/IP. Hakupäivä 25.11.2024
https://www.avg.com/en/signal/what-is-tcp-ip

Fortinet 2024. What is Transmission Protocol TCP/IP. Hakupäivä 25.11.2024
https://www.fortinet.com/resources/cyberglossary/tcp-ip

Github 2024. Generating a new SSH key. 3.9.2024
https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent

Hetzner 2024. Basic cloud configuration. Hakupäivä 3.9.2044
https://community.hetzner.com/tutorials/basic-cloud-config#step-1---insert-first-line

Hetzner 2024. Add SSL-certificate with Lets Encrypt. Hakupäivä 26.11.2024
https://community.hetzner.com/tutorials/add-ssl-certificate-with-lets-encrypt-to-nginx-on-ubuntu-20-04

Hetzner 2024. Hetzner rescue system. Hakupäivä 2.12.2024
https://docs.hetzner.com/robot/dedicated-server/troubleshooting/hetzner-rescue-system/#resetting-the-root-password

Hetzner 2024. How to SSH key. 3.9.2024
https://community.hetzner.com/tutorials/howto-ssh-key

Hetzner 2024. Migrate to Hetzner webhosting. Hakupäivä 19.9.2024
https://community.hetzner.com/tutorials/migrate-to-hetzner-web-hosting

Hurja 2024. Mikä on pilvipalvelu. Hakupäivä 5.9.2024
https://www.hurja.fi/blogi/mika-on-pilvipalvelu/

Hurja 2024. Pilvipalveluiden palvelumallit. Hakupäivä 5.9.2024
https://www.hurja.fi/blogi/pilvipalveluiden-palvelumallit-paas-iaas-baas-ja-saas/

Innofactor 2024. Mikä on pilvipalvelu. Hakupäivä 5.9.2024
https://www.itewiki.fi/p/mika-on-pilvipalvelu

Json-LD.org. Introduction. Hakupäivä 12.9.2024
https://json-ld.org/learn.html

Mashable 2017. Lekach, Sasha: The coder who built Mastodon is 24. Hakupäivä 16.8.2024
https://mashable.com/article/eugen-rochko-mastodon-interview

Mastodon 2024. Servers. Hakupäivä 15.8.2024
https://joinmastodon.org/servers

Mastodon 2024. Join Mastodon. Hakupäivä 15.8.2024
https://joinmastodon.org/

Mastodon 2024. Admin config. Hakupäivä 5.12. 2024
https://docs.joinmastodon.org/admin/config/

Mastodon 2024. Admin setup. Hakupäivä 5.12.2024
https://docs.joinmastodon.org/admin/setup/

Ruby On Rails 2024. Action mailer basics. Hakupäivä 9.12.2024
https://guides.rubyonrails.org/action_mailer_basics.html#action-mailer-configuration

StackOverflow 2015. Web.1 bundler: command not found: puma. Hakupäivä 5.12.2024
https://stackoverflow.com/questions/32532461/web-1-bundler-command-not-found-puma

TechnologyAdvice 2024. How to view your SSH-keys in Linux, MacOS and Windows. Hakupäivä 3.9.2024
https://www.techrepublic.com/article/how-to-view-your-ssh-keys-in-linux-macos-and-windows/

Techtarget 2024. Definition cloud server. Hakupäivä 5.9.2024
https://www.techtarget.com/searchcloudcomputing/definition/cloud-server

Termipankki 2024. Haku client. Hakupäivä 16.8.2024
https://termipankki.fi/tepa/fi/haku/client

W3.org 2018. ActivityPub. Hakupäivä 10.9.2024
https://atproto.com/guides/overview

W3.org 2018. Activity Streams vocabulary. Hakupäivä 10.9.2024
https://www.w3.org/TR/activitystreams-vocabulary/

W3.org 2018. Activity Streams 2.0. Hakupäivä 10.9.2024
https://www.w3.org/TR/activitystreams-core/#dfn-collectionpage