Iteration 1:

# Hvad skal vi fremstille

En liste over de emner/begreber fra faget Teknologi I anvender i jeres løsning med en kort beskrivende kommentar. Listen skal opbevares i jeres github-repository.

# Emner/begreber

* **IP-adresse** - (Internet protokol adresse) Vores hjemmeside bliver hostet på en logisk IP-adresse, og den enhed vi bruger har også en unik IP-adresse
* **Port** - For at komme ind på vores hjemmeside skal du igennem port 80
* **OSI** - Open Systems Interconnection er en syv lags model der bruges til at beskrive funktioner for et netværks system. Hvert lag har specifikke funktioner at udføre
* **TCP/IP** - TCP/IP er meget det samme som OSI modellen. TCP/IP modellen er en sammenpresset OSI model, i den forstand, at nogle af lagene er sat sammen til et, derfor har TCP/IP modellen kun 4 lag, og ikke syv. De fire lag beskrive dog de sammen funktioner som lagene i OSI modellen. OSI og TCP/IP modellen beskriver altså hvordan information bevæger sig gennem et netværk f.eks. til vores hjemmeside
* **LAN/ WLAN/ MAN/ WAN** - Forskellige typer af netværker, hvor LAN og WAN er nok de mest udbredte. Local area network er en gruppe af computerer, som er forbundet til hinanden i et mindre område eks. et kontor. Wide area network er i bund og grund internettet. For at komme på vores hjemmeside skal vores enhed have adgang til et netværk
* **Router** - Via en router er det muligt at komme på internettet så man kan komme ind på vores hjemmeside. En router sender information mellem to eller flere computer netværker
* **Mac-adresse** - Den fysiske adresse på en enhed
* **Learning** - Når noget bliver sendt til switchen lære den MAC-adressen
* **Flooding** - Sender packet til alle andre end afsenderen af dataen, for at finde en manglende MAC-adresse i switchens MAC-address table
* **Forwarding** - Der er forskellige måder at forward en frame på.
* Store and forward: Switchen kopier framet og tjekker den igennem efter fejl før den sender framet videre. Tage længst tid.
* Cut-Through: Switchen ser på MAC-Addressen og sender den bare videre uden at tjekke om fejl.
* Fragment Free: En blanding af de ovenstående to.Tjekker kun de første 64 bytes igennem.
* **Filtering** - Sker oftest når der skal floodes, betyder at den sender til alle andre MAC-adresser end den, som framet kom fra.
* **ARP** - Henter MAC-adressen gennem netværket
* **ARP-tabel** - Holder styr på alle IP- og MAC-adresser samt hvordan de er forbundet

Iteration 2:

# Hvad vi skal fremstille

Listen over de emner/begreber fra faget Teknologi I anvender i jeres løsning skal opdateres.

Uddyb forklaringen af de(n) protokol(ler) der bliver anvendt via jeres løsning (direkte eller indirekte)

# Emner/begreber/protokoller

* **HTTP** - Hyper Text Transfer Protocol, bliver primært brugt til kommunikation på internettet. HTTP er en client-server protocol, hvilket vil sige at request bliver sendt af en enhed som oftest vil være en Web-browser. Hver request vil blive sendt til en server som vil lave en respons. HTTP bruger TCP.
* **HTTPS** - Hyper Text Transfer Protocol Secure, det er en sikker version af HTTP. Det er en protokol som sikre kommunikationen mellem Web-browser og website. HTTPS er encrypted for at øge sikkerheden for data transfer.
* **DNS** - Domain Name System, bruger UDP**.** Den laver navnene fra hjemmesider om til IP adresser, eksemple på DNS - nslookup google.com - 216.58.211.110
* **TCP** - Transmission control protocol, er en type internet protokol (IP). TCP er “connection oriented”, som betyder at før data kan blive sendt, skal der være forbindelse mellem to hosts, også kaldet “3-way-handshake”. TCP passer bedst til applikationer der kræver stor pålidelighed, og transmissions tiden er mindre vigtig. TCP arrangere datapakkerne i en bestemt rækkefølge.
* **UDP** - User datagram protocol eller Universal datagram protocol, er en type internet protokol (IP). UDP er en “connection less” protokol, som betyder at der ikke er behov for, forbindelse for at sende data. UDP passer bedst til applikationer, der har behov for en hurtig og effektiv transmission f.eks. spil. UDP arrangere ikke datapakker i en bestemt rækkefølge. Alle pakker er uafhængig af hinanden.