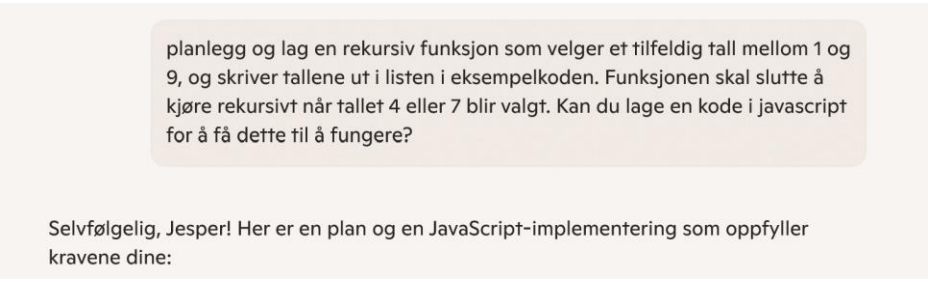
Arbeidskrav 3

Del 1 Rekursiv funksjon

Første man skulle gjøre var å finne ut av hva en rekursiv funksjon er og gjør. Dette fant jeg på to ulike nettsider. Den ene nettsiden fortalte meg hva funkjsonen er, mens det andre nettstedet viser eksempler av hvordan den fungerer i ulike eksempler som for eksempel å telle ned fra 3 til 1.

Når jeg skulle planlegge selve koden la jeg merke til at dette er noe vi har vært borti fra før av. Slik som å få et tilfeldig tall mellom 1 og 9. Jeg visste at man må bruke math.floor og math.random for å få til dette. Dette lærte vi også i en forelesning vi hadde om funksjoner hvor vi skulle returnere tilfeldige tall mellom a og b. Her brukte vi a og b, men i koden la jeg til « () * 9) + 1» fordi dette er for å lage en rekke mellom 1 og 9, og plusse på 1 for at det skal bli 10. Dette gjør også da at 10 ikke blir med i rekken. Deretter måtte jeg legge til i koden at tallene skulle komme opp ved ul-taggen ved id-navn numberList. Det er også en li-tagg hvor det står «Tall vil skrives ut her.». Først så tolket jeg dette som at dette var punktet hvor tallene skulle havne, men da måtte jeg ha fjernet li-taggen. Jeg har oppfattet det slik at det ikke var nødvendig å gjøre noe med html-koden i denne delen. For å få opp tallene i html koden ble jeg inspirert av det vi hadde gjort i timen om «arrays-with-objects» hvor vi hentet en array med en id, og fikk det dokumentert inn på selve dokumentet. Slik som jeg gjorde i denne koden. Steg tre var å få rekken til å stoppe på enten 4 ELLER 7. Da brukte jeg først «if» fordi det er en hendelse som kan oppstå. Deretter brukte jeg «||» fordi det skal enten være 4 eller 7. Jeg tenkte først at man kan bruke «&&», men dette betyr OG. I dette tilfellet er dette umulig å oppnå siden det er ett og ett tall som skal vises fram. For at funksjonen skal være rekursiv, så måtte jeg skrive funksjonen på nytt for at den skal gjenta seg.

AI sammenligning: Når jeg vil at AI skal gi meg en kode, vil jeg helst gjøre det så enkelt som mulig. På denne måten syntes jeg det er lettere å sammenligne og se hvor mange ulike måter man kan gjøre det på. Jeg kopierte teksten fra oppgave 1.2 og rett inn på CoPilot. Da fikk jeg den koden som har har lagt inn på rekursivAI.html. Denne koden er veldig ulik fra det jeg har gjort. Først og fremst, så er denne koden mer automatisert enn det min er. Jeg får en tallrekke til den kommer til 4 eller 7, men koden til AI trykker man på en knapp, også kommer ett og ett tall opp helt til man kommer til 4 eller 7. Den gir også beskjed om at den har stoppet fordi den har valgt 4 eller 7. Den bruker samme math.floor som meg for å velge de tilfeldige tallene. Den bruker også samme fremgangsmåte som meg for å definere hvilke tall den skal stoppe på.



Del 2 Passordgenerator

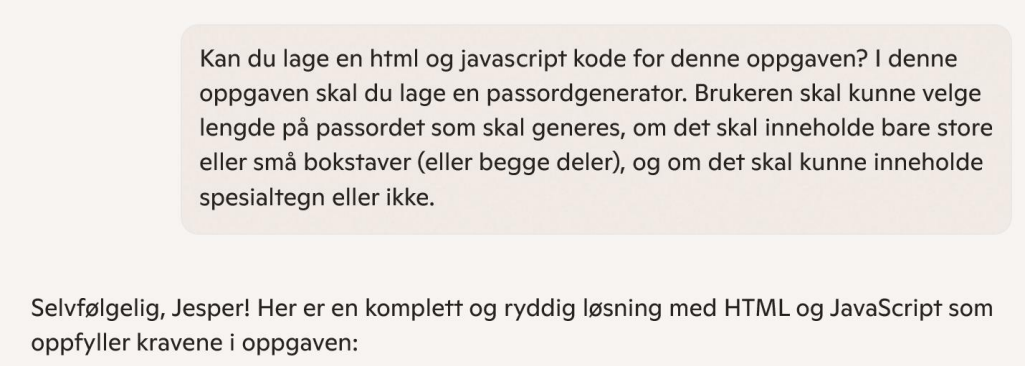
Først så skal man planlegge hvor man skal legge inn de ulike valgene på hvordan man skal filtrere passordet. Det står i oppgaven at det skal være symboler, små og store bokstaver eller begge. Jeg så også i koden at man hadde lagt til «includeNumbers». Selvom dette ikke stod i oppgaven, så valgte jeg å ta med dette i oppgaven allikavel, siden det var samme prosedyre. Dette jeg visste med en gang var at her måtte jeg gjøre noe i html for å få opp disse ulike valgmuligheten. Siden jeg har hatt webutvikling og føler meg komfortabel med html, var dette en grei sak. Jeg valgte å lage alle de ulike valgmuligheten som checkbokser fordi jeg følte at det var alle like og det ble mye av den samme prosedyren på de ulike valgmulighetene.

```
<form id="passwordform">
   <label for="length">Lengde på passord:</label>
   <input type="number" id="length" name="length"><br>
   <label for="includeUppercase">Store bokstaver</label>
   <input type="checkbox" id="includeUppercase" name="letters">
   <label for="includeLowercase">Små bokstaver</label>
   <input type="checkbox" id="includeLowercase" name="letters"><br>
   <label for="includeSymbols">Inneholde spesialsymboler</label>
   <input type="checkbox" id="includeSymbols" name="symbols">
   <label for="includeNumbers">Inneholde nummer</label>
   <input type="checkbox" id="includeNumbers" name="numbers"><br>
   <button type="button" onclick="generate()">Generer passord</button>
</form>
```

Under denne oppgaven så brukte jeg en YouTube video (se litteraturliste) som hjalp meg veldig med å skape et bilde av hvordan det skulle bygges opp. Den lærte meg også en ny ting som å stringe alle de ulike elementene sammen. Dette var .parseInt. Denne hjalp meg med å få alle de ulike elementene i en streng, for deretter å gå tilbake til den første verdien som i dette tilfellet er «length». Jeg brukte også .checked på de ulike valgmuligheten fordi de var i checkbokser, og hvor .checked gjør at koden forstår om disse muligheten er valgt. På «length» brukte jeg .value fordi dette var en verdi, og ikke en valgboks. Til slutt måtte jeg lage en array til selve passordet. Her fikk jeg mye hjelp av den tidligere nevnte YouTube-videon. Jeg måtte lage en tom array som å lage en liste, deretter while-løkke for å definere at lengden skal være mindre enn det som er definert. Deretter måtte jeg bruke math.floor for at den skal velge et tilfeldig tall mellom 0 og den gitte karakter lengden. Her brukte jeg da character som jeg hadde definert tidligere. Hva som inneholdt characters var gitt fra før av i startkoden. Det jeg måtte gjøre var å gi de riktige definisjoner som at upperCase faktisk inneholdt store bokstaver, og ikke små. Til slutt måtte jeg returnere karakterene for å lage en streng ut av det og returnere det ferdige passordet.

Sammenligning av AI: Akkurat som i oppgave 1, så kopierte jeg inn oppgave 2.2 for å se hva AI gjør annerledes. Her er det mer likheter enn det det var i sammenligningen av oppgave 1. Det koden gjør annerledes (og som jeg syntes er bedre) er at den gjør det litt mer tydligere hva som blir definerte. Som for eksempel store bokstaver sier den i en parantes at dette er da A-Z. En annen liten detalj til er at selve passordet blir grønt fordi AI har tatt seg i bruk av CSS. Det har ikke noe å si for oppgaven, men det ser litt penere ut. Selve utseendet på resultatet er også mer strukturert å se på, i motsetning til min kode. I min kode er alle valgboksene litt «oppå»

hverandre, mens AI er den strukturert som en liste. Jeg kunne også gjort dette, men valgte ikke å prioritere dette. Jeg syntes uansett det er gøy å se hva AI kan gjøre forskjellig fra meg.



Litteraturliste

Code Radiance (Regissør). (2023, januar 7). *Random Password Generator in JavaScript: Beginner Tutorial* [Videoopptak]. https://www.youtube.com/watch?v=m5XF1OQ6rSg

Lenovo. (2025). *How Does Recursion Work & What Are Its Advantages? | Lenovo US*. https://www.lenovo.com/us/en/glossary/recursion/

Tutorial, J. (2009). JavaScript Recursive Function. *JavaScript Tutorial*. https://www.javascripttutorial.net/javascript-recursive-function/

W3Schools. (u.å.). *JavaScript parseInt() Method*. Hentet 11. november 2025, fra https://www.w3schools.com/jsref/jsref_parseint.asp