Rapport for INFO134

Datasett.js:
Har tre forskjellige konstruktører som fungerer som et grensesnitt mot hvert av datasettene. Alle konstruktørene inneholder dem metodene som ble oppgitt at vi måtte ha med.

Sammenligning.js:
Tar in to kommunenummer og sammenligner dem mot hverandre ved hjelp av en tabell. Den kommunen med størst vekst i prosentpoeng blir markert med grønt i tabellen.

Oversikt.js:
Inneholder en «hide»-funksjon som gir oss muligheten at gå mellom de forskjellige vinduene uten å trenge å oppdatere siden. Den inneholder også onload metoden som laster opp dataen for oversikt fanen.

Detaljer.js:
Laster ned alle datasettene siden alle skal bli brukt i de respektive tabellene. Og siden utfører denne prosess. Den inneholder også noen hjelpe funksjoner som hindrer noe repetering av kode.

Index.css:
Denne filen står for all grafisk design på hjemmesiden og står for at rett format bli lagt til tabellene utfra størrelsen på skjermen.

Index.html:
Her blir de forskjellige tabellene opprettet men inneholder ingen elementer før de blir lagt til i Javascript koden. Inneholder også introduksjons fanen og «footerteksten». Står også for nedlastningen av Javascriptkoden.

1.

Datasettene lastes ikke ned samtidig og det er vanskelig å vite eksakt hva som blir lastet først, siden XMLHttpRequest() kjører asynkront. Datasettene blir garantert kjørt før koden siden koden er plassert inni «onload funksjonene».

```
Befolk.onload{
      Syssel.onload{
           Utdanning.onload{

         }
Utdanning.load();
             }
Syssel.load();
}
Befolk.load();
```

Disse funksjonene kaller på Requestene og inni der skriver vi koden som skal bli kjørt videre. Dette gjør at vi kan være sikker på at det ikke går an å klikke på knappene før noe blir lastet inn.

2.

Programmet vet det når onload funksjonen blir kjørt. Siden vi har nestet det som vist i oppgave 1, blir kode som er avhengig av datasettene kun kjørt når datasettet er ferdig lastet. Dessuten blir befolknings datasettet lastet ned før de andre scriptene siden den står over i html filen.

3.

Vi har løst dette ved hjelp av media queries. Når skjermen blir 760px bred vil media querien transponere tabellen.

```css
@media (max-width: 760px) {
    table {
        white-space: nowrap;
        font-size: 0;
        flex-wrap: nowrap;
        display: flex;
    }

    table * {
        font-size: 1rem;
    }

    tbody {
        display: flex;
    }

    tr {
        display: flex;
        flex-direction: column;
    }

    th,
    td {
        padding: .5rem;
        flex: 1 1;
    }
}
```

4.

Datasettene har ikke nøyaktig de samme kommunene. For å finne ut dette lagte vi et script som logget ut antall kommuner i hvert sett. Vi fant da at antallet var ulikt, noe som betyr at datasettene ikke inneholder de samme kommunene.

```javascript
var count = 0;
for(kommuner in syssel.getNames()) {
  count++;
}
console.log("syssel: " + count);

var count2 = 0;
for(kommuner in utdanning.getNames()) {
  count2++;
}
console.log("Utdanning: " + count2);
```