Repetisjonsoppgåver i

Javascript, Express og databasar

NB: Dynamisk dokument. Me legg til og endrar innhald etter kvart.

Anbefalt utgangspunkt for teori og forklaringar til korleis ting fungerer er:

1. [IT 1 - Aschehoug univers (aunivers.no)](https://innhold.aunivers.no/fagpakker/realfag/informasjonsteknologi-1-2/it-1) (lærebok)
2. [W3Schools Online Web Tutorials](https://www.w3schools.com/)
3. [MDN Web Docs (mozilla.org)](https://developer.mozilla.org/en-US/)
4. Node JS: <https://indd.adobe.com/view/38062418-9d87-4b6e-b4fb-c907cd10f942>
5. ...

Innhold

[Steg 1 – Vanilla JS 2](#_Toc163040587)

[Utvalt oppgåve 1 - Variablar og enkle utrekningar 2](#_Toc163040588)

[Utvalt oppgåve 2 – Valgsetningar: Aldersgrense 2](#_Toc163040589)

[Utvalt oppgåve 3 – Valgsetningar: Førarkort 2](#_Toc163040590)

[Utvalt oppgåve 4 – Løkker og arrays: Tilfeldige tal 2](#_Toc163040591)

[Utvalt oppgåve 4 – Arrays og funksjonar: Biletgalleri 3](#_Toc163040592)

[Utvalt oppgåve 5 – Forms (AKA skjema) og objekt 3](#_Toc163040593)

[Utvalt oppgåve 6 – Matbutikk: createElement og appendChild 3](#_Toc163040594)

[Utvalt oppgåve 7 – async function og await 4](#_Toc163040595)

[Steg 2 – Express 5](#_Toc163040596)

[Oppgåve 1 – starte prosjekt, og ruter 5](#_Toc163040597)

[Oppgåve 2 – SQL og database 5](#_Toc163040598)

[Oppgåve 3 – Generere HTML frå JSON 5](#_Toc163040599)

[Oppgåve 4 – Opprette ein brukar via nettside 5](#_Toc163040600)

[Oppgåve 5 – Kryptering av passord 5](#_Toc163040601)

[Oppgåve 6 – Nytt miniprosjekt: Nettavis 6](#_Toc163040602)

[Bonussteg – SQL og datamodellering 6](#_Toc163040603)

[Oppgåve 1 - SQL - spørringar 6](#_Toc163040604)

[Oppgåve 2 - … 6](#_Toc163040605)

# Steg 1 – Vanilla JS

Lag «vanilla JS»-appar. Skriv koden sjølv, linje for linje!

## Utvalt oppgåve 1 - Variablar og enkle utrekningar

La brukaren skrive inn to sjølvvalte (stads)namn og berekn deretter differansen i antall teikn. Forsøk gjerne å bruke tekstfelt og ein knapp som i skjermbiletet under. Eventuelt **prompt for å gjere det enklare og tryggare**. Pass i tillegg på at du unngår minusteikn i utskrifta dersom den første staden har færre antall teikn enn den andre (hint: absolutt verdi).

Bonusoppgåve: Handter brukarfeil, som til dømes manglande input.

Nokre fleire krav du kan legge til?

Et bilde som inneholder tekst, skjermbilde, Font, programvare

Automatisk generert beskrivelse

[Løysningsforslag](https://github.com/hausnes/it2/blob/main/02-03-04-05-variablar-valg-l%C3%B8kker-funksjonar/stadsnavn-diff-v2-ferdig.html).

## Utvalt oppgåve 2 – Valgsetningar: Aldersgrense

Lag ei aldersgrense på nettsida di. Spør brukaren kor gamal h\*n er. Dersom over 18 kan du vise eit kattebilete pluss litt tekst, dersom ikkje skal du gje beskjed om at innhaldet dessverre er sperra (el.l.).

Hint: img src=”” er lovleg å bruke. Då vises ikkje eit bilete i heile tatt.

[Løysningsforslag](https://github.com/hausnes/it2/blob/main/02-03-04-05-variablar-valg-l%C3%B8kker-funksjonar/alderskontroll-if-else.html).

## Utvalt oppgåve 3 – Valgsetningar: Førarkort

Lag ein enkel app som kan hjelpe folk med å få vite kva setifikat dei kan ta. Du ber om alderen og presenterer deretter bileter og tekst som fortel kva dei kan ta sertifikat på (meir avansert om du vil: [https://lovdata.no/forskrift/2004-01-19-298/§3-1](https://lovdata.no/forskrift/2004-01-19-298/%C2%A73-1)):

1. Yngre enn 16. Ingenting, brukaren er for ung.
2. 16 år: moped
3. 18 år: bil
4. 21 år: lastebil
5. 24 år: buss

Kan du legge inn ei øvre aldersgrense òg? Kanskje som ein beskjed om at du må gjennomføre faste kontrollar dersom du er over eit visst antall år?

Løysningsforslag: [Tips til korleis du kan løyse (deler av) denne](https://github.com/hausnes/it2/blob/main/02-03-04-05-variablar-valg-l%C3%B8kker-funksjonar/if-else.html)

## Utvalt oppgåve 4 – Løkker og arrays: Tilfeldige tal

Lag ein array. Bruk ei løkke for å fylle denne med 200 tilfeldige heiltal mellom 1 og 100. Sjå gjerne MDN (Mozilla Developer Network) for hjelp til Math.random. Bruk færre tal og mindre variasjon (1-10, til dømes), dersom du har behov for at du kan telle over og sjekke manuelt sjølv.

Bruk løkker og variablar for å finne ut fylgjande (ikkje innebygde funksjonar i JS):

* Tel kor mange av tala i arrayen som er 5 eller høgare.
* Tel kor mange som er nøyaktig 4.
* Summer alle tala i arrayen.
* Finn gjennomsnittsverdien av alle tala.
* Finn det høgaste talet.
* Finn det lågaste talet.
* …

## Utvalt oppgåve 4 – Arrays og funksjonar: Biletgalleri

Lag eit enkelt biletgalleri vha. array. Meir konkret: Lagre bileta (navnet) i ein array og gjer det mogleg å bytte bilete ved å trykke på ein “neste-knapp”.

Her er det heilt sikkert fristande å finne ein ferdigløysning på nett el.l. Eg anbefalar å lage ein særs enkel versjon basert på det me har lært så langt, for å forstå prinsippet skikkeleg.

[Løysningsforslag](https://github.com/hausnes/it2/blob/main/02-03-04-05-variablar-valg-l%C3%B8kker-funksjonar/biletgalleriArray.html).

## Utvalt oppgåve 5 – Forms (AKA skjema) og objekt

Du skal lagre informasjon om deltakarar på eit arrangement. Du skal benytte eit form/skjema. Pass på at du benyttar gode input-typar i HTML (text, number, email osv.). Benytt required, placeholder, min, max osv.

* Fornavn
* Etternavn
* Adresse
* Postnummer
* Poststed
* Telefon
* Passord
* Fødselsår

Legg til lyttefunksjon på sjølve skjema (ikkje knappen).

Bruk evt.preventDefault for å ikkje gå til ei anna nettside når du skal behandle input frå skjema. Spør lærar!

Skriv ein oppsummering av alt som kom frå input. Eksempelvis «Jo Bjørnar Hausnes, Øvre Kråk… er 89 år gamal og …».

Lagre all informasjonen i eit objekt.

## Utvalt oppgåve 6 – Matbutikk: createElement og appendChild

En ny matbutikk på nett, «Rett hjem», ønsker din hjelp til å lage en bestillingsapplikasjon.

Hovedfokus: Applikasjonen skal vise fram tilgjengelige produkter med et lite bilde, et navn og en pris. Gå gjennom array med objekt og generer HTML dynamisk vha. createElement og appendChild.

Avansert: Når brukeren klikker på en matvare, skal den legges til i en «handlekurv», og en lyd kan spilles av. Hvis brukeren klikker på samme matvare flere ganger, skal antallet endres i «handlekurven». Innholdet i «handlekurven» skal hele tiden være synlig i applikasjonen, i tillegg til totalprisen for de valgte matvarene.

Igjen - her forventes det at du bruker en array med objekter (eller lignende) for å lagre matvarene. Bruk en løkke for å gå gjennom denne samlingen, der du benytter createElement og appendChild for å generere HTML-en.

Du bør og vurdere å bruke en lignende måte å lagre handlekurven på.

Tenk gjennom og begrunn kvifor dette er ein god måte å jobbe på.

## Utvalt oppgåve 7 – async function og await

Be om data (typisk 1 stk brukar til å byrje med) frå randomuser.me. Hugs at dette kan ta litt tid, så her må du bruke async function og await. Data kjem i JSON-format. Konverter etter behov.

**Steg 1**: Skriv ut ein del av informasjonen i console, til dømes namn, land og eit bilete. Kontroller at alt fungerer så langt.

**Steg 2**: Benytt createElement og appendChild, slik du gjorde i oppgåve 6, for å teikne opp dei tilfeldige brukarane på nettsida.

…

Meir kjem etter behov. Lag gjerne eigne liknande oppgåver sjølv, med eit tema du er interessert i.

# Steg 2 – Express

Skriv koden sjølv, linje for linje. Ta utgangspunkt i læreboka ([Javascript og Node (adobe.com)](https://indd.adobe.com/view/38062418-9d87-4b6e-b4fb-c907cd10f942)) for å få hjelp til dei ulike stega. Ver nøye med å gjere ein ting om gongen og forstå det du held på med.

## Oppgåve 1 – starte prosjekt, og ruter

Opprett eit express-prosjekt. Bruk npm for å initialisere prosjektet.

Installer pakken express.

Opprett app.js.

Lag grunnleggande rute til rot (/) – send «Testside fungerer» som resultat til den som besøker.

Lag grunnleggande rute til ei public-mappe og ei side der (HTML, CSS, JS) – bruk «sendFile».

Lat serveren lytte på localhost, port 3000.

Kontroller at rutene du har oppretta fungerer ved å besøke localhost:3000 i nettlesaren. Besøk dei ulike undersidene/rutene.

Lag fleire ruter og besøk desse for å teste at alt fungerer, og at du forstår kva som skjer.

Bruk eigne ord og forklar/diskuter (gjerne med ein nabo, eller lærar).

## Oppgåve 2 – SQL og database

Opprett ein database. Bruk SQL eller <https://sqlitestudio.pl/> for å lage ein enkel db med ein enkelt tabell «bruker». Den skal innehalde brukarid, epost, namn og passord. Fyll inn med eksempeldata, minst 3 stk. brukarar.

Installer pakken better-sqlite3 i Node-prosjektet ditt.

Lag ei rute i app.js som hentar ut all informasjonen frå databasen vha. ein spørring, og viser denne fram, gjerne som rein JSON no i første omgang.

…

## Oppgåve 3 – Generere HTML frå JSON

Me tek utgangspunkt i ruta frå oppgåve 2 som returnerte ei JSON-fil.

Lag eit HTML-dokument i public-mappa som du kallar brukere.html.

Hent JSON frå oppg. 2 (hugs async function og await), og generer nødvendig HTML (createElement og appendChild). HTML-en treng ikkje vere så avansert, du kan til dømes ha namn som h1 og e-post som p rett under.

Tips til utviding: Legg til profilbilete for brukaren. Korleis er det best å handtere dette?

…

## Oppgåve 4 – Opprette ein brukar via nettside

Lag ei nettside som gjer det mogleg å opprette ein brukar, og skrive denne til databasen.

Her må du hugse å:

* Skrive nødvendig SQL for å opprette ein brukar
* Lage ei rute som skriv til databasen
* Lage ei nettside med eit skjema som nyttar ruta over.
* Handter ulike moglege feil som kan dukke opp.

…

## Oppgåve 5 – Kryptering av passord

Benytt bcrypt for å kryptere passordet som blir benytta i oppgåve 4.

## Oppgåve 6 – Nytt miniprosjekt: Nettavis

Besøk ein artikkel på NRK.no og lag ein datamodell for korleis du meiner denne ser ut.

Steg 1: Gå gjennom artikkelen og tenk gjennom kva tabellar som er nødvendige, og kva relasjonar som gjeld mellom desse.

Steg 2: Teikn ein modell for tankane du sit igjen med frå steg 1 på papir. Legg opp til litt «problematisering», som viser at du forstår problem du kan havne i, og feil du bør unngå = diskuter/snakk høgt. Du må sjølv vise at mange-til-mange-relasjonar ikkje er lov, og kva du gjer for å løyse dette.

Steg 3: Lag ein digital versjon av modellen i MySQL Workbench.

Steg 4: Lag ein database basert på modellen, fyll inn eksempeldata og køyr nokre spørringar for å sjå om «ting» fungerer.

Lever inn:

* Bilete av teikna modell (frå papir)
* PNG/JPG av modell frå Workbench
* Skjermbilete av spørringar (2 stk.) som viser alle artiklane som tilfredsstiller fylgjande krav:
  + "Jo Bjørnar Hausnes" er ein av journalistane
  + "Jo Bjørnar Hausnes" og "Trond Sneåas Skauge" sine beste saker saman (mest populære basert på antall visningar)

…

# Bonussteg – SQL og datamodellering

## Oppgåve 1 - SQL - spørringar

Installer SQLite studio (SQLiteStudio), og øv på å skrive spørringar (SQL).

Last ned databasefila «world» (lenke: <https://dev.mysql.com/doc/index-other.html>). Opne denne i SQLite studio.

1. List opp alle land
2. Finn hovedstaden til et spesifikt land
3. List byer med befolkning over 5 millioner
4. Vis land med over 100 millioner befolkning
5. Finn landet med høyest forventet levealder
6. List de 5 mest befolkede landene i verden
7. Vis befolkningen til hver verdensdel. PS. Se på GROUP BY og SUM
8. Finn land som snakker engelsk
9. Sorter land etter befolkning i synkende rekkefølge
10. Tell antall land i hver verdensdel PS. Se på GROUP BY og COUNT
11. Vis byen i hvert land som har flest innbyggere i landet (Se på MAX og GROUP BY)
12. Finn alle offisielle språk i Europa
13. Informasjonen i databasen er utdatert. Velg de 5 største landene og 5 største byene og oppdater informasjon
14. …

## Oppgåve 2 - …

…