Nivå 2.

Hva er Vue?

Vue er et JavaScript-rammeverk for å bygge brukergrensesnitt. Den bygger på standard HTML, CSS og JavaScript og gir en deklarativ og komponentbasert programmeringsmodell som hjelper deg effektivt å utvikle brukergrensesnitt, enten de er enkle eller komplekse.
Introduksjon.
Du har nå fullført nivå 1. HTML, CSS og Javascript.
Disse 3 er basen for alt og ved å ha en god forståelse for disse, så vil andre rammeverk bli lettere å forstå.
Det er flere rammeverk og biblioteker ute, men de 3 største er for øyeblikket:

- React, Vue og Angular

Hvor vi da kommer til å fokusere på Vue 3, i oppgavene nedenfor.
#Note: Kan hende at vue cli, GIT og node må installeres.

- Node: https://nodejs.org/en/download/
- GIT: https://git-scm.com/downloads
    - https://git-scm.com/book/en/v2/Getting-Started-First-Time-Git-Setup

Sjekk deretter at disse har blitt installert i CMD.

- Node: node -v
- npm: npm -v
- git: git --version

#Note 2: Det er viktig å bruke de ressursene man har tilgjengelig som en programmerer.
Ressurser som:

- Dokumentasjon Vue: https://vuejs.org/
- Youtube: https://www.youtube.com/
- Google: https://www.google.com/
- Stack Overflow: https://stackoverflow.com/
- Meldinger på telegram
- Det er viktig å gi beskjed dersom man står fast, slik at man kan få litt hjelp til å jobbe seg fremover.

Oppgaver.

Oppgave 1. Lage et nytt Vue prosjekt.
Ressources:

- https://vuejs.org/guide/quick-start.html#creating-a-vue-application
- npm init vue@latest

#Note: Velg Manually select features.

- Og følg instruksene som blir gitt.

Router må være med.
Sjekk at siden blir kjørt etterpå, ved å:

- cd inn til prosjektet.
- npm run dev.

Oppgave 2: Prøv deg litt frem i Vue.

- Åpne prosjektet i VS Code
- Se litt over arkitekturen på prosjektet.
- Da spesielt:
- views - selve sidene
- components: Bygge klosser
- Router: Navigasjon og routing i prosjektet.
- package.json: Informasjon om prosjektet og dependensies.
- Du behøver ikke å kunne alt, men heller vite litt grovt om hva de forskjellige seksjonene er.

Oppgaven her er:

- Finn helloworld.vue i src/components/
- Slett all innhold i denne filen.
- Bare se hvilket komponenter som ligger ute, og test dem ut.
- Se koden, skriv koden inn i helloWorld komponenten.
- Gå inn i nettleser, og se at komponenten blir fremvist.

Oppgave 2.5: Vi skal lage en counter i HomeView.vue.

Her lager vi en veldig enkel counter, med en knapp som øker tallet når mann trykker på knappen.

```
<template>
 <p>{{ count }}</p>
 <button @click="count++">Increase count</button>
```

```
</template>
<script setup>
import { ref } from 'vue';
const count = ref(0);
<script>
```

Oppgave 3: Lag en ny komponent.

Oppgaven er: Lag en egen komponent, og bruk denne i views/about.vue siden.

- Komponenten skal være ett nytt kort, og bli plassert i src/components/cards/card.vue
- Bruk koden fra linken nedenfor.
- Viktig at du skriver all kode manuelt, og ikke copy paste.
- https://www.w3schools.com/howto/howto_css_cards.asp
- Etter at denne komponenten er laget, importer den inn i views/about.vue
- Import Component from "../components/component.vue"

Sjekk denne siden, og se om kortet vises der.
Selve innholdet i kortet er ikke så viktig.

- Bare legg merke til strukturen man bruker for å lage kortet.
- template
- root element
- script
- styles
- Og hvordan man bruker denne komponenten i en annen fil.

Oppgave 4: Struktur av en Vue side.

Oppgave: Jeg vil at du skal vise frem dette kortet midt på siden horisontalt.
Altså | --- [card] --- |

Oppgave 5: Duplisere og fremvise kortet 5 ganger.

Jeg vil at du skal bruke denne komponenten 5 ganger, og få fremvist den slik:

- 3 kort først
- 2 kort på linjen nedenfor.

Ressurser:

- Grid: https://css-tricks.com/snippets/css/complete-guide-grid/
- Flexbox https://css-tricks.com/snippets/css/a-guide-to-flexbox/

Oppgave 6: v-model og print av en v-model.

V-model er også kalt 2 veis binding.
Noe som betyr: den fungerer begge veier.
Eksempel nedenfor:

- Vil da vise frem verdien som blir skrevet i en input, gjennom v-model.

 # I Template

```
<h1>{{ tittel }}</h1>
<input type="text" v-model=" tittel"></input>
```

 # I Script

```
<script setup>
Import Ref from "vue"
  const  tittel = ref("hallo")
</script>
```

Nå skal du via en v-model, forandre / legge på en tittel på kort komponenten.

- Teksten på denne tittelen skal da ligge som en v-model
- Du skal printe denne øverst på kortet.

Links:

- https://vuejs.org/guide/components/v-model.html#component-v-model
- https://vuejs.org/v2/guide/forms.html

Oppgave 7. La kortene bli unike.

Nå er det på tide at kortene blir unike.
Du skal lage et array med 5 objekter i seg.

- https://vuejs.org/v2/guide/list.html

Du må se nærmere på **v-for** og hvordan du kan **iterere** deg gjennom dette arrayet.
Når du er ferdig så skal det vises 5 unike kort på siden.

- Du kan selv bestemme hvilket nøkler som blir brukt, men vil anbefale disse:
- title
- description
- image

Eksempel:

 # I Template

```
  <div>
  <div v-for="(item, index) in myEpicArray" :key="index">
    <button> {{ item.title }} </button>
  </div>
</div>
```

 # I Script

```
<script setup>
import {ref} from 'vue'
```

```
const myEpicArray = ref ([
    { title: 'knapp 1' },
    { title: 'knapp 2' },
    { title: 'knapp 3' }
  ])
</script>
```

Oppgave 8. Vis frem 4 av 5 kort, ved bruk av en v-if statement.

- https://vuejs.org/v2/guide/conditional.html

Oppgave: Her vil jeg at du skal lære hvordan IF ELSE fungerer i Vue.

1. I Arrayet med objektene som du lagde, legg til en ny nøkkel.
- Denne skal du bruke, for å sjekke kort “statusen”
- 5 stk { title: “ “, visMeg: “ “ },
- Alle utenom 1, skal ha verdien visMeg: “ja”
- Den ene skal ha verdien: visMeg: “nei”
1. Bruk denne for å sjekke om kortet skal vises eller ikke.

   Syntax nedenfor:

- <div v-if=” “ >
- <div v-else>
1. Nå vil jeg at du skal bruke denne til å vise frem 4 av 5 kort.

Summering av det du har gjort til nå.
Du har:

- Satt opp et Vue prosjekt
- Du har laget en helt ny komponent, og brukt dette på en side.

- Hvordan en side er bygget opp.
- Brukt v-model til å printe en verdi i kortet.
- Iterert inn data via Vue.js sin versjon av en for loop.
- Filtrert data, via Vue.js sin versjon av en if statement.

Oppgave 9. Dialog.

En Dialog er det tilsvarende som en Modal.
Den skal ikke vises som default, men komme frem når du gjør noe.
Så nå vil jeg at du skal lage en ny komponent innad i /components mappen.
Du kan godt lage en sub-mappe for denne komponenten.

- Eksempel: src/components/dialogs/myDialog.vue

Dialogen skal inneholde et kort, med noe tekst i seg.
Du skal lage en knapp på de kortene som blir iterert inn, som da åpner dialogen ved klikk.
#Husk:

- Importer og plasser denne dialog komponenten, inn der du skal bruke den.
- Ha en v-model hvor dialogen er false som default.
- Ha en åpne og lukke funksjon.

Oppgave 10. Sende data via Props.

Props er da, “å sende data nedover til en child”.
Jeg vil nå at du skal klikke på denne kort knappen, og vise frem informasjonen til dette kortet i dialogen.
Eksempel:

 # I Parent Komponenten

```
<template>
  <Modal :showModal="showModal" @closeModal="showModal = !showModal" />
  <button @click="openModal">Open modal</button>
</template>
<script setup>
import { ref } from 'vue';
import Modal from '../components/Modal.vue';
const showModal = ref(false);
const openModal = () => {
  showModal.value = true;
  console.log(showModal.value);
};
</script>
```

 # I Dialog Komponenten

```
<template>
  <div v-if="showModal">
    <h1>This is an epic modal</h1>
    <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Velit, eos?</p>
    <button @click="$emit('closeModal')">Close me</button>
  </div>
</template>
<script setup>
import { defineProps } from 'vue';
```

```
const props = defineProps({
  showModal: {
    type: Boolean,
    required: true,
  },
});
</script>
<style scoped>
div {
  background-color: #fff;
  border-radius: 5px;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);
  padding: 20px;
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
  width: 400px;
}
</style>
```

Oppgave 11. Router

La oss nå lage 2 nye sider i src/views

- Resources og profil

2. Lag en standard mal på disse sidene.

- Template
- Root element
- Paragraph som har litt tekst.

3. På tide å bygge opp Router for disse to sidene.

- Gå inn i Router fila
- Legg merke til hvordan oppsettet her er gjort.

4. Lag 2 nye objecter, for disse 2 sidene.

- Ikke følg oppsettet til About, men heller det første objektet.
- Husk:
    - Importere komponenten (siden)
    - Lage ett nytt object for hver side.

5. Besøk disse sidene i browseren.

- /resources
- /profil

Oppgave 12. Tabell

På resource siden, så skal du nå lage en tabell.
Den skal inneholde disse feltene:

- Tittel
- Beskrivelse
- Link

Du skal kunne søke og filtrere i denne tabellen.
Det skal ligge en knapp på tabellen.

- Den skal ligge helt til slutt på hver rad.
- Når denne trykkes, så skal en dialog komme frem med informasjonen til denne linjen.

Selve linjen skal også kunne trykkes på.

- Den skal da expandere og vise frem informasjon.
- Når man trykker på en annen linje, så skal denne lukkes og den andre åpnes.

Oppgave 13. Navbar

Nå er det på tide å få fikset navbaren.

1. Lag denne til en egen komponent.

- Oppførsel til navbaren kan være:
    - Den skal vises hele tiden.

        - Forsvinne når man scroller ned.
        - DUkke opp når man scroller opp.
        - Være en 2 linjes navbar
        - Navbar med en sidebar knyttet til den.
        - Annet..
1. Importer og render den i app.vue filen.
2. Den skal inneholde en logo
- Logo skal kunne klikkes, og ta deg til startsiden
1. Link til de andre sidene som ble laget.
2. En Login knapp
- Denne behøver ikke å ha en login funksjon.
- Skal være en dialog
- Skal ha ett skjema, som simulerer en login.
- Skal ha en registrerings mulighet.

Tips: Kan være en egen registrerings dialog.

Oppgave 14. Footer

1. Lag en ny komponent.
2. Denne skal da være en footer, og importeres i app.vue.
3. Du bestemmer selv innholdet på footeren.
4. Den skal holde seg nede på siden, og ikke flyte oppover.

Oppgave 15. Opprydding.

1. Gå over koden som du har skrevet.
2. Rydd opp i den, og gjør den så fin som mulig.

Det som kommer til å bli fokusert på i hovedoppgaven er:

- Gjennomføring av den.
- Koden i seg selv, og hvordan den er skrevet.
- Commits og fremgangsmåte
- Presentasjon av hovedoppgaven.
- Visning på de forskjellige skjermstørrelsene
- Feedback, kommunikasjon og bruk av ressurser.

De neste oppgavene kommer når en database er gjort klar.
Gi beskjed når du har kommet til oppgave 15.