# Koodin kommentointi/Readme

### Sivujen toiminnan suunnittelu

Aloitin luomalla suunnitelman käyttäjän omia sivuja varten. Ajatuksena oli, että kun käyttäjä kirjautuu sisään omilla tunnuksillaan onnistuneesti, **navbar** komponenttiin ilmestyy "oma sivu" vaihtoehto.

## Login -komponentin toiminta:

React-komponentti nimeltään Login, joka on tarkoitettu käyttäjän kirjautumisnäkymän toteuttamiseen. Seuraavassa selostetaan koodin toiminta vaiheittain:

### Tuonnit ja Tyylien Liittäminen

**useNavigate** tuodaan react-router-dom-kirjastosta, jota käytetään sivujen välillä navigointiin.

Login.css-tyylitiedosto liitetään mukaan, joka sisältää kirjautumislomakkeen tyylit.

Reactin **useState**-hook tuodaan, jonka avulla hallitaan komponentin tilaa.

**useUser** on konteksti-hook, joka mahdollistaa käyttäjätietojen käsittelyn (esim. kirjautuminen).

**index.css**-tyylitiedoston tuonti, joka sisältää yleiset tyylit.

**Link** tuodaan myös react-router-dom-kirjastosta, jota käytetään linkkien luomiseen.

Komponentin Rakenne

**Login** on funktionaalinen komponentti, joka sisältää kirjautumislomakkeen.

Kaksi tilamuuttujaa (**username ja password**) alustetaan tyhjiksi merkkijonoiksi useState-hookilla.

**navigate**-funktio alustetaan useNavigate-hookilla, jotta voidaan ohjata käyttäjä eri reitteihin.

**validate** on asynkroninen funktio, joka suoritetaan lomakkeen lähetystapahtumassa.

Jos käyttäjätunnus ja salasana eivät ole tyhjiä, yritetään kirjautua sisään login-funktiolla, joka saadaan **useUser**-hookista.

Kirjautumisen onnistuessa siirrytään etusivulle (**navigate**("/")).

Virhetilanteessa tulostetaan virhe konsoliin.

**Lomakkeen Renderöinti**

Komponentti palauttaa JSX-syntaksilla kirjautumislomakkeen, jossa on kaksi syöttökenttää: käyttäjätunnus ja salasana.

Syöttökentät päivittävät tilamuuttujia setUsername ja setPassword kun niiden arvo muuttuu.

Lomake sisältää kaksi painiketta: "Kirjaudu" lähetyspainikkeena ja "Poista tili" ilman toiminnallisuutta.

Link-komponentti tarjoaa reitin, josta pääsee luomaan uuden tilin (/CreateAcc).

Kokonaisuutena

Kun käyttäjä täyttää lomakkeen ja painaa "Kirjaudu"-painiketta, validate-funktio käynnistyy. Mikäli käyttäjätunnus ja salasana ovat kelvollisia, kirjautumisyritys tehdään. Onnistuneen kirjautumisen jälkeen käyttäjä ohjataan etusivulle. Epäonnistuneessa tapauksessa virhe näytetään konsolissa. Lisäksi käyttäjällä on mahdollisuus siirtyä tilinluontisivulle klikkaamalla "Luo tili" -linkkiä.

## personal-sivu

Kun käyttäjä siirtyy omalle sivulle, sovellus aukaisee **Personal** sivukomponentin. **Personal** sivulla käyttäjä voi joko hakea luomansa suosikkilistat, tai tarkastella omaa käyttäjäkohtaista ryhmäkomponenttia.

Mikäli käyttäjä haluaa siirtyä suosikkilistoihin, on hänen navigoitava "**suosikkilistat**" buttonin kautta **FavoriteLists** sivukomponentille.

FavoriteLists

hakee tietokannasta käyttäjän jo tekemät suosikkilistat, ja antaa mahdollisuuden myös uusien listojen luomiseen ja poistamiseen.

FavoriteLists hakee suosikkilistat personal\_pages taulusta tietokannasta, mutta jokaisen listan varsinainen sisältö haetaan erillisestä favorite\_list\_content taulusta.

Jokainen suosikkilista on yhdistetty vastaavaan listasisältöön, eli mikäli personal\_pages taulussa listan id on 1, tämä on yhdistetty favorite\_list\_content taulussa riveihin missä list\_id on 1, list\_id:n ollessa viiteavain, näin osataan hakea oikea sisältö oikeaan listaan.

### Specific list

Kun käyttäjä haluaa hakea tietyn listan sisällön, hän voi klikata listan nimeä, ja näin haetaan **SpecificList** komponentti, joka hakee listakohtaisen sisällön favorite\_list\_content taulusta. SpecificList sivukomponentilla on "lisää elokuvia" button, joka ohjaa käyttäjän takaisin aloitussivulle, jossa on toteutettu elokuvien hakeminen.

## Tietyn elokuvan haku

Käyttäjä voi täällä valita tietyn elokuvan, jolloin aukeaa elokuvakohtainen sivu **MoviePage**. MoviePage sivulla on sisäänkirjautuneelle käyttäjälle vaihtoehto lisätä elokuva suosikkilistaan, tämä on toteutettu pudotusvalikolla, joka hakee käyttäjän tekemät suosikkilistat, ja buttonilla joka lisää elokuvan valittuun suosikkilistaan. Kun lista on valittu ja buttonia painettu, MoviePage ottaa valitusta elokuvasta talteen JSON muodossa nimen, kuvan ja elokuvan kuvauksen, ja tallentaa ne listakohtaiseen "list\_content" sarakkeeseen.

Tämä järjestely mahdollistaa usean suosikkilistan luomisen, usean elokuvan lisäämisen listaan ja elokuvien poistamisen listalta. Lisätyt elokuvat ilmestyvät allekkain listakohtaiselle sivulle.

## Ryhmä-sivun toiminta

Kun käyttäjä haluaa siirtyä ryhmiin, hänen on navigoitava "**Ryhmät**" -painikkeen kautta **GroupSearchPage**-sivukomponentille. Tämä sivu hakee tietokannasta käyttäjien luomat ryhmät ja näyttää ne listana. Ryhmiä voi selata painamalla "seuraava sivu" -painiketta tai "edellinen sivu" -painiketta.

Jokaisessa ryhmässä on mahdollisuus liittyä ryhmään painamalla "**Liity ryhmään**" -painiketta. Tämä lähettää pyynnön liittyä ryhmään ryhmän **adminille**. "Liity ryhmään" -painikkeen oikealla puolella näkyy myös sen ryhmän kuvaus, joka laajentuu, kun hiiren kohdistin on kuvaustekstin päällä. Lisäksi käyttäjällä on mahdollisuus luoda uusi ryhmä syöttämällä haluamansa ryhmän nimen ja painamalla "Luo ryhmä" -painiketta.

Kun käyttäjä on luonut ryhmän, hän voi tämän jälkeen muokata ryhmäkohtaisia sivuja ja kirjoittaa ryhmän kuvauksen. Ryhmäsivut näyttävät erilaisilta riippuen siitä, onko kyseinen käyttäjä ryhmän admin vai vain perusjäsen. Ryhmän luoja voi antaa admin-oikeuksia muille ryhmän jäsenille, ja admin-oikeuksilla adminit voivat muokata sivuja ja tarvittaessa poistaa ryhmän jäseniä. Adminit eivät kuitenkaan voi poistaa toisia admineja, sillä nämä oikeudet ovat ainoastaan ryhmän luojalla.

## tyylitys käyttäen sass kuvauskieltä

Jokaiselle sivulle ja komponentille on luotu SASS-tyylitiedosto, joka määrittelee tyylejä. Koodissa käytetään SASS:n syntaksia ja ominaisuuksia, kuten muuttujia ja sisäkkäisiä sääntöjä. Seuraavassa selostetaan koodin toiminta:

@import "../index.sass" tuo toisen SASS-tiedoston, joka todennäköisesti sisältää yleisiä muuttujia ja tyylejä, joita käytetään tässä tiedostossa.

.formcontent-luokka:

Asettaa elementin näyttämään flexboxina ja suuntaa sen lapset pystysuunnassa.

Antaa elementille padding-arvot ja leveyden sekä marginaalin.

.formlabel-luokka:

Määrittää fonttikoon, painon ja marginaalin alapuolelle.

#submit-id:

Asettaa marginaalin, taustavärin, tekstin värin, reunuksettomuuden, padding-arvot, fonttikoon, marginaalin yläpuolelle, pyöristetyt kulmat, kirjasinpainon, kirjainvälin ja fontin perheen.

Sisäkkäiset &:hover-säännöt määrittävät mitä tapahtuu kun hiiri on elementin päällä (muuttaa tausta- ja tekstinväriä).

.input-luokka:

Asettaa fonttikoon, padding-arvot, nollaa reunuksen, lisää varjon, pyöristää kulmat ja asettaa leveyden.

&:active, &:focus ja &:hover ovat pseudo-luokkia, jotka määrittävät elementin tyylin kun se on aktiivinen, fokusoitu tai hiiren yli liikutetaan. Ne muuttavat varjon ulkonäköä, poistavat reunuksen ja ääriviivan, ja &:hover lisää taustavärin.

.linksign-luokka:

Asettaa paddingin ja marginaalin.

Koodissa käytetty $blk, $bg, $loact ja $active ovat SASS-muuttujia, joiden arvot on määritelty tuodussa index.sass-tiedostossa. Nämä muuttujat viittaavat eri väriarvoihin, jotka voidaan muuttaa keskitetysti index.sass-tiedostossa.

Yhteenvetona, tämä SASS-koodi luo tietyn visuaalisen ilmeen kirjautumislomakkeelle ja sen elementeille, kuten syöttökentille ja linkeille, käyttäen SASS:n ominaisuuksia kuten muuttujia ja sisäkkäisiä sääntöjä.

### Sivujen suunnittelu

Sivujen ulkoasu suunniteltiin käyttäen hyödyksi [Figmaa](https://www.figma.com/file/G7AYt2RZlfyDEjLPfafRss/Elokuvasivusto?type=design&node-id=59%3A1903&mode=design&t=CXYzDfr7hKAwTJgP-1). Figma tarjoaa suoran CSS ulosviennin ulkoasulle, joten oli helppo neuvotella, ja päättää sivujen ulkoasusta.

