Projekt „joogipood“

1. Tee uus projekt: „npx create-react-app joogipood“
2. Pane see käima sisestades järgmised käsud „**cd joogipood**“ ja seejärel „npm start“:

„cd“ tähendab lühendit „change directory“ ning see vahetab kausta. Npm start käivitab projekti aadressil localhost:3000 .

1. Et hoida projektis puhtust, kustuta failid mis midagi juurde ei paku. See viskab errori, kuna tuleb kustutada ka importid ning nende kasutuskohad.
2. Võtame kasutusele Routingu, selleks lisame projekti „react-router-dom“ mooduli.

Sulge projekt sisestades „ctrl + c“ terminalis. Kirjuta terminali „npm i react-router-dom“. Pead olema „joogipood“ kaustas. Installeerimine lisab võimekuse liikumaks erinevate URL-del.

1. Lisame index.js faili <BrowserRouteri>, kindlasti peab sellest ka impordi tegema.
2. Lisame App.js faili Link, Route ja Routes impordi react-router-dom node\_modules kaustast – selle impordi abil saame hakata Link, Route ja Routes võimekusi HTMLis kasutama.
3. Teeme kolm erinevat URL aadressi, millele on võimalik minna: /, /lisa, /halda, <Routes> sisse, <Route path ja element /> abil
4. Tee,e 3 uut faili, Avaleht, LisaJook, HaldaJooke, pages kausta sisse. Võta nad kõik kasutusele <Route /> element loogeliste sulgude sees, lisa ka impordid.
5. Lisame nupud, millega on võimalik URL-l liikuda, kasutased <Link to=““> tag’e
6. Lisa uus FAIL src sisse „joogid.json“. Pane selle sisse tühjad kandilised sulud.

Meenutame loogikat failist number 15, tagasisided. Kuidas seal käis andmete muutmine ja väljakuvamine:

Text

Description automatically generated

Kui mõelda, kuidas peaks käima aga tagasisidede näitamine MÕNEL TEISEL LEHEL kui andmed ON SIIN failis? Ükskõik millisel. Seda ei saa aga teha kui andmed on siinses failis, useState sulgude sees.

Lahendus? Kasutada võõrast faili.

Number 1 aga teeme TÄPSELT SAMA LOOGIKA SIIA nagu tagasisided.js faili oli. Step-by-step:

Tee HaldaJoogid.js sisse uus **useState**, mille sulgude sisse väärtuseks Array (kandilisest sulust kandiliste sulgudeni), mis koosneb algväärtusena 5-st elemendist**: ["Põltsamaa", "Aura", "Kadarbik", "Saku", "Värska"]**. Pane muutuja nimeks „**joogid**“, uuendamise nimeks „**uuendaJoogid**“. Kindlasti on vaja ka useState osas **importi**.

Kuva kõik joogid välja HTMLs **.map** funktsionaalsuse abil. Võid ise valida, mida paned map() sisse noole ette: map(*element* => <div>) map(*jook* => <div>) map(*e* => <div>) jne. Lihtsalt seda pead ka järgnevalt noole taga kasutusele hakkama võtma. Mina panin seekord print-screenidel **element =>**

Teeme igaühele kustutamise võimaluse:

1. Loo igaühele nupp, sisuks „x“
2. Tee JS poole peal (returnist üleval) funktsioon nimega const kustuta = (), kus võtad sulgude vahelt vastu järjekorranumbri
3. Tee iga nupu sisse onClick, mis kutsub selle funktsiooni välja ning saadab sulgude vahel talle index muutuja
4. Index muutuja on järjekorranumber – seda peab ise looma, kirjutades .map( element => ) asemel: .map( **(element,index)** => ) . Selle **parempoolse „index“** saadadki onClick sees nupule vajutades funktsiooni kaasa.
5. Kustutatakse **.splice** abil.

Kustutamiseks on vaja võtta see array ning öelda järjekorranumber, millist kustutatakse ning mitu tükki kustutatakse: [].splice(0,1) <- kustutab esimese. Meie teeme: joogid.splice(index,1);

1. Kui see on tehtud, siis kustutamine tegelikult toimib, aga leht ei uuene. See käib vaid läbi useState funktsiooni. Võtame selle kasutusele: uuendaJoogid(joogid.slice()) . See tähendab, et uuendatakse „joogid“ muutujat tema enda abil.

Pane tähele, et üleval on „**splice**“ ja allpool on „**slice**“.

Lisa uue joogi lisamise võimalus:

Tee HTML kõige alla label, input, button.

Seo input useRef muutujaga (nimeks nt jookRef). UseRef võimaldab kuulata kõike input sisse sisestavat. Lisa see useRef kõige üles, impordi useRef.

Lisa HTMLi seos input ja ref osas: ref={jookRef}

Lisa uus funktsioon: const lisaUusJook = () => {

Kutsu see välja nupule klikkides – onClick.

Funktsiooni sees tee järgnevat:

joogid.push(jookRef.current.value); 🡨 lisab input sisse sisestatud väärtuse KÕIGE LÕPPU juurde.

Järgmiseks: uuendaJoogid(joogid.slice())

Kontrolli, kas saad lisada, kustutada, välja kuvada.

Nüüd jõuame faili juurde. Faile kasutame selleks, et saaksime välja kuvada jooke ka teisel lehel ning see uueneks kui me lisame või kustutame. Praegu näiteks lisame ja kustutame, aga meil on VÕIMATU seda saata Avalehele. Selleks aga, et neid 5-te jooki saaks jagada nii Avalehele kui Haldamise lehele, peame tõstma need eraldi faili ja neid sealt kasutama hakkama.

Tee uus fail „joogid.json“, vajutades parem klõps -> „new file“ src kausta peale.

Tõstame kõik 5 jooki useState sulgude seest faili sisse (**koos** kandiliste sulgudega)

Nii Avaleht kui ka HaldaJooke hakkavad neid tooteid nüüd sealt seest kasutama. Selleks peab mõlemas failis tegema **impordi** ja selle **faili sisestama sinna, kuhu algselt läksid tooted**.

Kui HaldaToode on tehtud, siis:

Tasub koheselt tähele panna anomaaliat, mis võib tekkida *kustutamisel.*

Hetkel kustutatakse joogid muutuja seest üks jook ära ning lisatakse joogid muutuja sisse uus jook.

Visuaalselt justkui kõik toimiks, aga tegelikult mitte, sest ME EI KUSTUTA FAILIST. Kui tahame, et muutuksid kõik lehed, kes neid andmeid kasutavad, peame kustutama failist ja lisama faili ehk kasutama tootedFailist muutujad nii .push() kui ka .splice() ees.

Nüüd toimuvad muutused failis.

Võtame kõik failis olevad joogid nüüd kasutusele ka Avalehel, täpselt sama loogikaga.

Kliendilt tuleb soov, et nad ei soovi samal lehel jooke lisada. Kustutamine on neile OK, aga lisamine peaks nende sõnul käima siiski eraldi lehelt. Tegemist on standardse lahendusega, et lisatakse eraldi lehelt.

A picture containing application

Description automatically generated

Selleks tõstame HaldaTooted.js lehelt lisamise välja. Teeme lihtsalt ridade välja tõstmise, ehk ctrl + x ehk cuttimise (või hiirega parem klõps -> cut). Tõstame kõik mis on seotud lisamisega välja. Ainuke asi, millest teeme copy-paste on faili import.

Nüüd on aga „uuendaJoogid()“ erroris:

Text

Description automatically generated

Tuletame meelde, kus me kasutasime uuendaJoogid funktsiooni -> useState paremast poolest. Tegemist oli HTMLi uuendamisega. Kui me teeme aga jookide lisamist ERALDI lehelt, siis pole ju ka HTMLi uuendamist vaja – HTMLi uuendab URLi vahetus. Tuletame meelde reeglit: HTMLi uuendab kas URLi vahetus või useState. Enne oli selleks useState, sest kogu loogika oli ühel lehel. Nüüd on aga URLi vahetus, kuna on 2 erinevat lehte: jookide lisamise ja jookide haldamise. Ehk: peaksime uuendaJoogid funktsiooni kustutama, enam pole HTMLi vaja useState abil uuendada.

Pärast seda peaks olema kõik on roheline ja töötab. Saad jooke lisada ja URLi vahetusega neid ka Avalehel ja Toodete haldamise lehel näha.

Pane tähele, et refreshiga kaovad kõik lisatud joogid.

Katseta ka, kui kustutad faili sisu terves mahus ära, mis juhtub: veebileht on erroris.

Kui lisad faili sisse tühjad kandilised sulud: [] , veebileht töötab, sest nüüd ei ole tegemist enam tühjusega, vaid tühja array’ga.