9. mājasdarbs

Mājaslapā ir iesākta komponente <SocialGrid />. Atkarībā no lietotāja nospiestās pogas “Posts”, “Users”, “Comments” tiek izpildīts fetch un no servera atgriezti dati priekš katras attiecīgās kategorijas. Ir izveidota arī komponente <PostCard />, kas saņem un attēlo datus no “Post” kategorijas. Uzdevums:

1. Nepieciešams izveidot komponentes <UserCard /> un <CommentCard />, kas saņems un atteļos attiecīgo kategoriju datus.

2. Nepieciešams iekš <SocialGrid /> integrēt komponenti <Counter />. Šī komponete kontrolēs cik ierakstu priekš katras no kategorijām tiks atgriezti no servera.

Sagatavošanās:

Navigēt terminālī uz sava mājasdarba uzdevuma mapi.
Savā uzdevuma mapē terminālī izpildīt komandu:
npm install

Lai iedarbinātu webpack serveri un varētu apskatīt kodā veiktās izmaiņas pārlūkā, terminālī izpildam komandu:
npm start

Atveram pārlūkprogrammā adresi http://localhost:5173/, kur darbojas mūsu Vite serveris.

Uzdevuma izpilde:

1. Iekš <SocialGrid /> komponentes izveidot komponentes stāvokli (state) priekš **count** vērtības, ko kontrolēs <Counter /> komponente. Sākotnējā count vērtība tiek iestatīta kā 5.

```
const [count, setCount] = useState(5)
```

2. Pašlaik fetch no servera vienmēr pieprasa 10 rezultātus. Mēs vēlamies, lai pieprasīto rezultātu skaits sakristu ar `count` vērtību. Pieprasījuma URL pielāgojam parametra `_end` vērtību, lai tā būtu vienāda ar `count`:

```
`https://jsonplaceholder.typicode.com/${resourceType}?_start=0&_end=${count}`
```

3. fetch tiek izsaukts ar useEffect hook palīdzību. useEffect var saņemt 2 argumentus:
   - **funkciju ko izsaukt**
   - **dependancy masīvu**
   Ja dependancy masīvs netiek norādīts, useEffect tiks izsauks pēc katra komponentes attēlošanas (render). Ja šis masīvs tiek norādīts, useEffect tiks izsaukts tikai mainoties kādai no šī masīva vērtībām.

Dotajā brīdī šis masīvs satur tikai `resourceType` vērtību un useEffect tiks izsaukts lietotājam nospiežot uz kādas no kategoriju pogām, nomainot `resourceType` vērtību. Lai useEffect un līdz ar to fetch izpildītos arī uz `count` vērtības maiņu, jāpapildina dependancy masīvs ar `count`:

```
[resourceType ,count]
```

4. Nepieciešams izveidot komponentes <UserCard /> un <CommentCard />.
   Izveidojam attiecīgās mapes un attiecīgos **.css** un **.jsx** failus.
   Izveidot HTML struktūras un vizuālo noformējumu.
   Paraugam var izmantot komponenti <PostCard />.

5. Importēt šīs komponentes iekš <SocialGrid /> komponentes un izmantot tās iekš metodes renderCards **switch** izteiksmes atbilstošā **case**.
   <UsersCard /> kā props saņems un attēlos:
   name={item.name}
   email={item.email}
   website={item.website}
   <CommentsCard /> kā props saņems un attēlos:
   name={item.name}
   text={item.body}
   email={item.email}
   Gluži kā komponente <PostCard /> saņem un attēlo props: title, text.
   Darbojoties ar props šīm komponentēm, censties atdarināt tās pēc iespējas vairāk <PostCard /> paraugam, jo sevišķi tajā, kā tiek izmantoti props (padoti komponentei un izmantoti iekš komponentes koda).

6. Iekš <SocialGrid /> importēt <Counter /> komponenti, ievietot to tieši zem <GridControls /> return() funkcijā.
   <Counter /> komponentei būs nepieciešams padot props handleCountChange={handleCountChange} un count={count}. M
   Funkcija handleCountChange jāpapildina **switch** izteiksme. **case** "increment" palielinās `count` vērtību, bet **case** "decrement samazinās `count` vērtību.
   *** Papildus, brīvprātīgi** - izveidojam loģiju, lai count vērtība nevarētu tik iestatīta lielāka par 10 vai mazāka par 1.

Šo instrukciju imantot kopā ar norādēm komentāros iekš <SocialGrid /> un <Counter />.