function ListPage() {

// Получение номера страницы из URL

const { page } = useParams();

const pageNum = Number(page) || 1;

const navigate = useNavigate();

const { data, isLoading, isError } = useQuery({

queryKey: ["items", pageNum],

queryFn: () => fetchItems(pageNum),

keepPreviousData: true // не убирать данные прошлой страницы, пока не загрузится новая

});

if (isLoading) return <div>Загрузка...</div>;

if (isError) return <div>Ошибка!</div>;

return (

<div>

<ul>

{data.items.map(item => (

<li key={item.id}>

<Link to={`/items/${item.id}`}>{item.title}</Link>

</li>

))}

</ul>

<div style={{ marginTop: 16 }}>

{pageNum > 1 && (

<button onClick={() => navigate(`/items/page/${pageNum - 1}`)}>

Назад

</button>

)}

{pageNum < data.totalPages && (

<button onClick={() => navigate(`/items/page/${pageNum + 1}`)}>

Вперед

</button>

)}

</div>

</div>

);

}

**Если нет настройки на back с номерами страниц**

Шаги:

1. Запрашиваем весь список с сервера один раз.

2. Делаем разбиение на страницы на клиенте:

   Например, храним текущую страницу в адресной строке, а отображаем только нужные элементы.

3. Номер страницы — в адресе (через React Router).

const ITEMS\_PER\_PAGE = 10; // по сколько показываем на странице

function ListPage() {

  const { page } = useParams();

  const pageNum = Number(page) || 1;

  const navigate = useNavigate();

  // Грузим весь массив, один запрос

  const { data: items, isLoading, isError } = useQuery({

    queryKey: ["all-items"],

    queryFn: fetchAllItems

  });

  if (isLoading) return <div>Загрузка...</div>;

  if (isError) return <div>Ошибка!</div>;

  // Всего страниц

  const totalPages = Math.ceil(items.length / ITEMS\_PER\_PAGE);

  // Элементы для текущей страницы

  const startIdx = (pageNum - 1) \* ITEMS\_PER\_PAGE;

  const endIdx = startIdx + ITEMS\_PER\_PAGE;

  const pageItems = items.slice(startIdx, endIdx);

  return (

    <div>

      <h1>Список элементов (страница {pageNum})</h1>

      <ul>

        {pageItems.map(item => (

          <li key={item.id}>

            <Link to={`/items/${item.id}`}>{item.title}</Link>

          </li>

        ))}

      </ul>

      <div>

        {pageNum > 1 && (

          <button onClick={() => navigate(`/items/page/${pageNum - 1}`)}>

            Назад

          </button>

        )}

        {pageNum < totalPages && (

          <button onClick={() => navigate(`/items/page/${pageNum + 1}`)}>

            Вперед

          </button>

        )}

      </div>

    </div>

  );

Сделать кнопки с номерами страниц ("1", "2", "3" и т.д.)

const ITEMS\_PER\_PAGE = 10;

function ListPage() {

const { page } = useParams();

const pageNum = Number(page) || 1;

const navigate = useNavigate();

const { data: items, isLoading, isError } = useQuery({

queryKey: ["all-items"],

queryFn: fetchAllItems

});

if (isLoading) return <div>Загрузка...</div>;

if (isError) return <div>Ошибка!</div>;

const totalPages = Math.ceil(items.length / ITEMS\_PER\_PAGE);

const startIdx = (pageNum - 1) \* ITEMS\_PER\_PAGE;

const endIdx = startIdx + ITEMS\_PER\_PAGE;

const pageItems = items.slice(startIdx, endIdx);

// Создаем массив с номерами страниц

const pageNumbers = [];

for (let i = 1; i <= totalPages; i++) {

pageNumbers.push(i);

}

return (

<div>

<h1>Список элементов (страница {pageNum})</h1>

<ul>

{pageItems.map(item => (

<li key={item.id}>

<Link to={`/items/${item.id}`}>{item.title}</Link>

</li>

))}

</ul>

<div>

{pageNumbers.map(num => (

<button

key={num}

onClick={() => navigate(`/items/page/${num}`)}

// Подсветка активной кнопки

disabled={num === pageNum}

>

{num}

</button>

))}

</div>

</div>

);

}