Veebiarendus

Front-End arendus

Martti Raavel

martti.raavel@tlu.ee

Tänased teemad

- Meenutame eelmist loengut
- React ja Axios
- useEffect hook
- Dünaamilised marsruudid
- Tingimuslaused
- Flash messages

Millest rääkisime eelmisel korral?

React ja Axios

Oleme juba varem kasutanud Axios-i, et teha päringuid serverisse. Axios on lihtne ja paindlik HTTP klient, mis töötab nii brauseris kui ka Node.js-is. Nüüd, kui me oskame luua Reacti rakendust erinevate lehekülgedega ja oskame kasutajalt ka andmeid küsida, siis saame minna oma rakenduse integreerimisega API-ga, kuhu saame sisse logida, kust saame andmeid pärida jms.

Axios-e paigaldamine

Kõigepealt peame Axios-i oma projektile paigaldame. Selleks kasutame npm-i:

```
npm install axios
```

Axios-e kasutamine

Axiose kasutamiseks tuleb see vastavasse komponendi sisse importida. Näiteks:

```
import axios from 'axios';
```

Seejärel saame hakata juba päringuid tegema. Näiteks:

```
const response = await axios.post('https://blog.hk.tlu.ee/login', {
  email: 'user@user.ee',
  password: 'password'
  });
```

Saadud andmete kasutamine

Kui oleme päringu teinud, siis saame vastuse kätte ja ilmselt soovime selle vastusega ka midagi teha. Näiteks peale sisselogimist tagastatakse meile tõenäoliselt mingi token, mida peame hakkama kasutama kõikide järgmiste päringute puhul.

```
const token = response.data.token;
```

Tokeni saame omakorda salvestada kasutaja sessiooni või local storage-i.

```
localStorage.setItem('token', token);
```

Tokeni kasutamine

Kui meil on token olemas, siis saame seda kasutada kõikide järgmiste päringute puhul, et server teaks, et me oleme sisse logitud.

```
const token = localStorage.getItem('token');

const response = await axios.get('https://blog.hk.tlu.ee/posts', {
  headers: {
    Authorization: `Bearer ${token}`
  }
});
```

useEffect hook

Oletame, et me oleme nüüd rakendusse sisse loginud ja soovime näha postituste nimekirja. Kuidas teha nii, et kui me läheme postituste lehele, et siis automaatselt laetakse postitused alla?

Selleks kasutame useEffect hook-i.

useEffect hook - mis?

useEffect hook on Reacti hook, mis võimaldab meil teha midagi komponendi elutsükli jooksul. Näiteks saame kasutada seda, et teha päring serverisse, kui komponent laetakse.

```
useEffect(() => {
  const fetchPosts = async () => {
    const response = await axios.get('https://blog.hk.tlu.ee/posts');
    setPosts(response.data);
  };

  fetchPosts();
}, []);
```

useEffect hook - kuidas?

useEffect hook võtab vastu kaks argumenti: funktsiooni ja massiivi. Funktsioon on see, mida me soovime teha ja massiiv on see, millal me soovime seda teha.

Kui massiiv on tühi, siis tehakse päring ainult üks kord, kui komponent laetakse. Kui massiivis on mõni muutuja, siis tehakse päring iga kord, kui see muutuja muutub.

Dünaamilised marsruudid

Oletame nüüd, et meil on olemas lehekülg, kus kuvatakse postituste nimekiri ja iga postituse peal on nupp, millele vajutades saame minna postituse enda lehele, kus kuvataks nii postituse sisu, kui ka sellega seotud kommentaarid. Ei ole mõistlik ja ilmselt ka võimalik, et me teeksime iga postituse jaoks eraldi lehe.

Dünaamilised marsruudid - kuidas?

Selle asemel saame teha ühe postituse lehe, kuhu saame minna kasutades postituse `id`-d ja dünaamilist marsruuti. Näiteks:

```
<Route path="/posts/:id" component={Post} />
```

> `:id` on dünaamiline osa marsruudist, mis võimaldab meil kasutada seda osa muutujana.
>
> Näiteks kui me läheme aadressile `/posts/1`, siis `id` on `1`.

Dünaamilised marsruudid - kuidas?

Kui meil on dünaamiline marsruut, siis saame sellelt inforatsiooni (muutujat) lugeda omakorda postituse komponendi sees `useParams` hook-i abil. Näiteks:

```
const { id } = useParams();
const [post, setPost] = useState(null);

useEffect(() => {
  const fetchPost = async () => {
    const response = await axios.get(`https://blog.hk.tlu.ee/posts/${id}`);
    setPost(response.data);
  };

  fetchPost();
}, [id]);

return (
  <div>
    <h1>{post.title}</h1>
    <p>{post.body}</p>
  </div>
);
```

Tingimuslaused

Tingimuslaused on oluline osa iga rakenduse loomisel. Näiteks kui me soovime kuvada kasutajale mingit teadet, kui ta on sisse loginud, siis saame kuvada talle tervituse, kui ta ei ole sisse loginud, siis paluda tal sisse logida.

Tingimuslaused - kuidas?

Tingimuslauset saame kasutada mitmel erineval viisil. Näiteks:

- `if`-lauset kasutades
- Loogilist `&&`-operaatorit kasutades
- Ternaaroperaatorit kasutades

Näide: `if`-lause

```
import React from 'react';

function UserGreeting({ isLoggedIn }) {
  if (isLoggedIn) {
    return <h1>Welcome back!</h1>;
  }
  return <h1>Please sign up.</h1>;
}

export default UserGreeting;
```

Näide: Loogiline `&&`-operaator

```
import React from 'react';

function Mailbox({ unreadMessages }) {
  return (
    <div>
      <h1>Hello!</h1>
      {unreadMessages.length > 0 &&
        <h2>You have {unreadMessages.length} unread messages.</h2>
      }
    </div>
  );
}

export default Mailbox;
```

Näide: Ternaaroperaator

```
import React from 'react';

function Greeting({ isLoggedIn }) {
  return (
    <div>
      {isLoggedIn ? (
        <h1>Welcome back!</h1>
      ) : (
        <h1>Please sign up.</h1>
      )}
    </div>
  );
}

export default Greeting;
```

Flash messages

Flash messages on teade, mis kuvatakse kasutajale ajutiselt ja seejärel kaob. Näiteks kui kasutaja on edukalt sisse loginud, siis võiksime kuvada talle teate, et ta on sisse loginud ja seejärel see teade kaob.

Flash messages - kuidas?

Flash messages saame teha kasutades Reacti `useState` hook-i ja `setTimeout` funktsiooni. Näiteks:

```
const [message, setMessage] = useState(null);

const handleLogin = async () => {
  const response = await axios.post('https://blog.hk.tlu.ee/login', {
    email, password
  });

  if (response.data.token) {
    localStorage.setItem('token', response.data.token);
    setMessage('You have successfully logged in!');
    setTimeout(() => {
      setMessage(null);
    }, 5000);
  } else {
    setMessage('Login failed!');
  }
};

return (
  <div>
    {message && <div>{message}</div>}
    <button onClick={handleLogin}>Login</button>
  </div>
);
```

Kodune töö