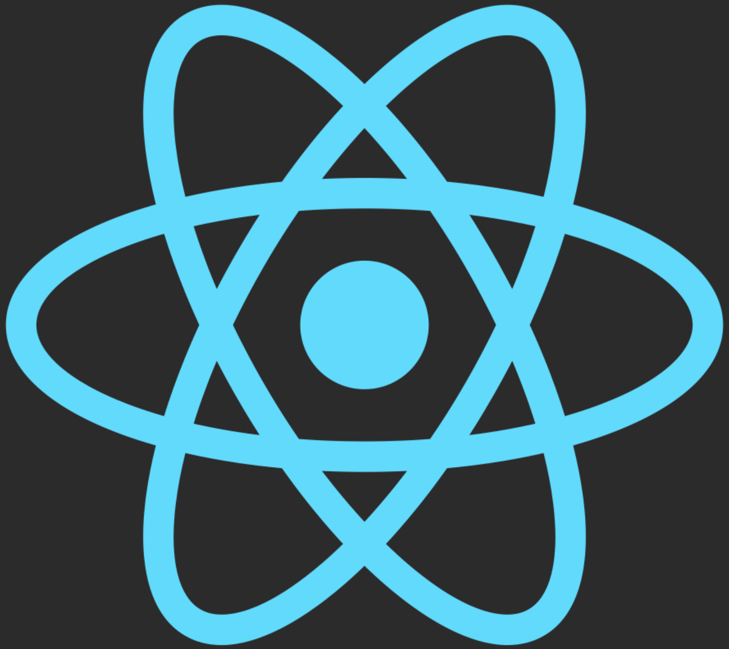
REACT



Matija Katanec

mkatanec@croz.net

Prezentacija: Josip Tomić

O Reactu

Javascript library za izradu korisničkih sučelja

Održava ga Facebook i community

Koristi proširenje JavaScripta nazvano JSX

Ispod haube koristi Virtual DOM

Brz i jednostavan za korištenje

Poznati korisnici:

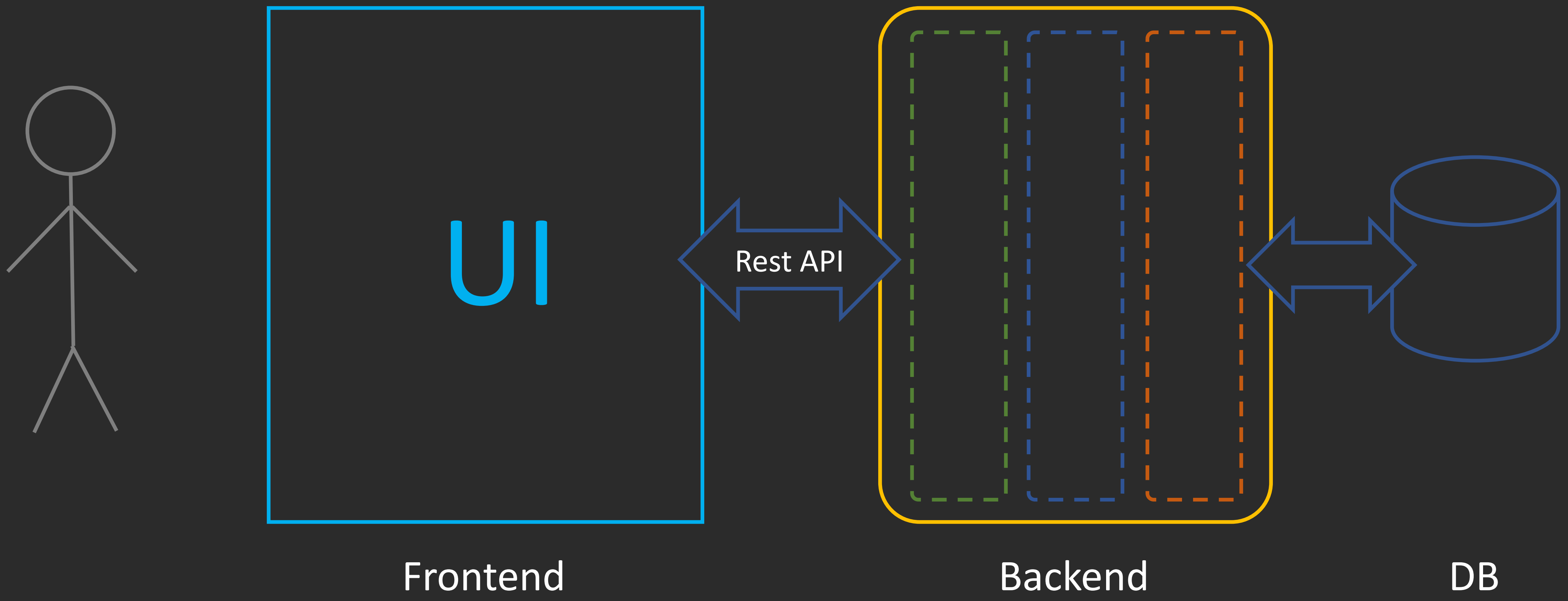
O single page aplikacijama

Web stranice / aplikacije

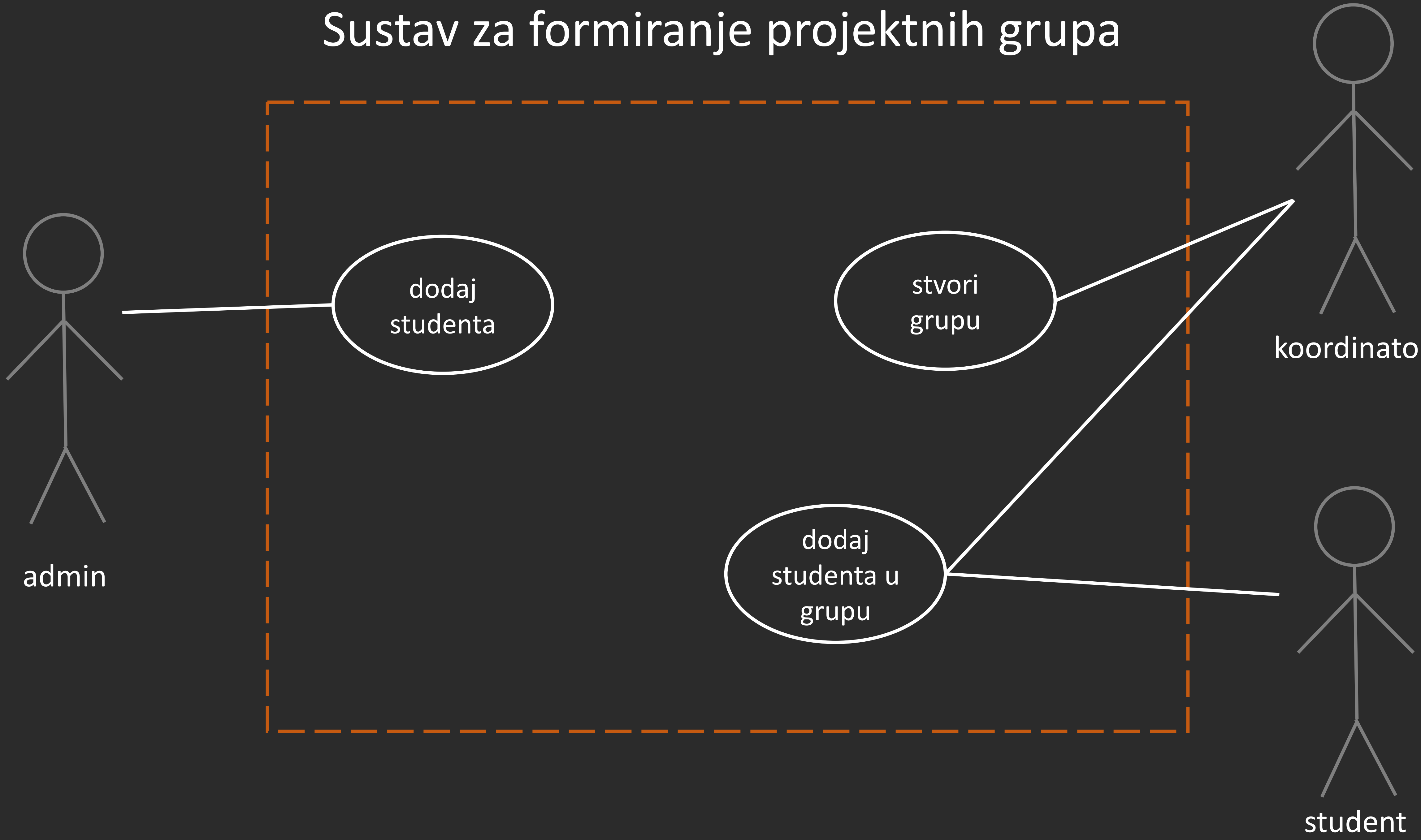
Dinamičko prikazivanje sadržaja ovisno o radnjama korisnika

Zahtjeva samo jedno loadanje stanice

Ponašanje slično desktop aplikacijama


UI
Rest API
Frontend
Backend
DB

Sustav za formiranje projektnih grupa



Alati potrebni za rad

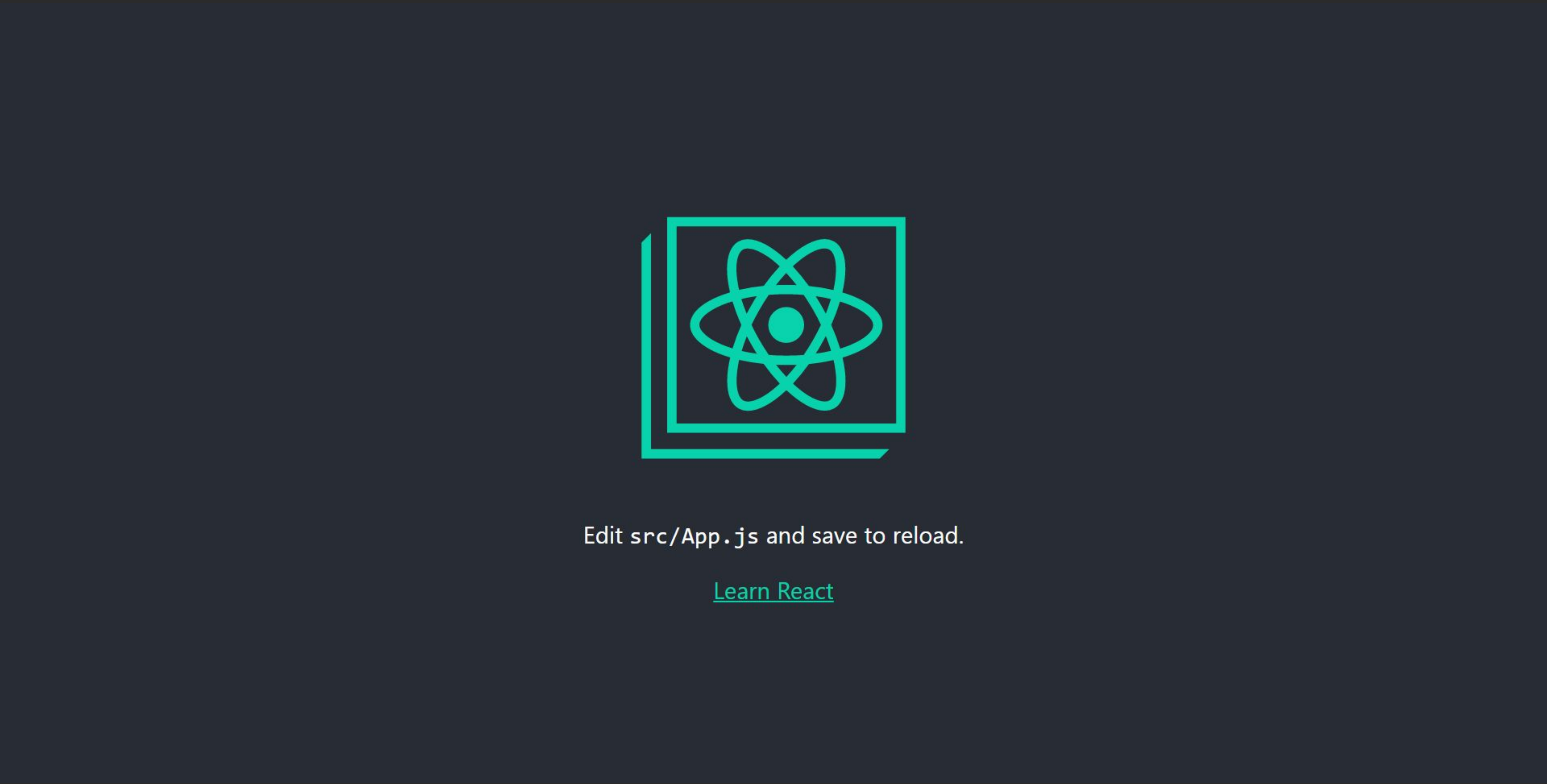
node + npm

IDE

Kreiranje novog React projekta

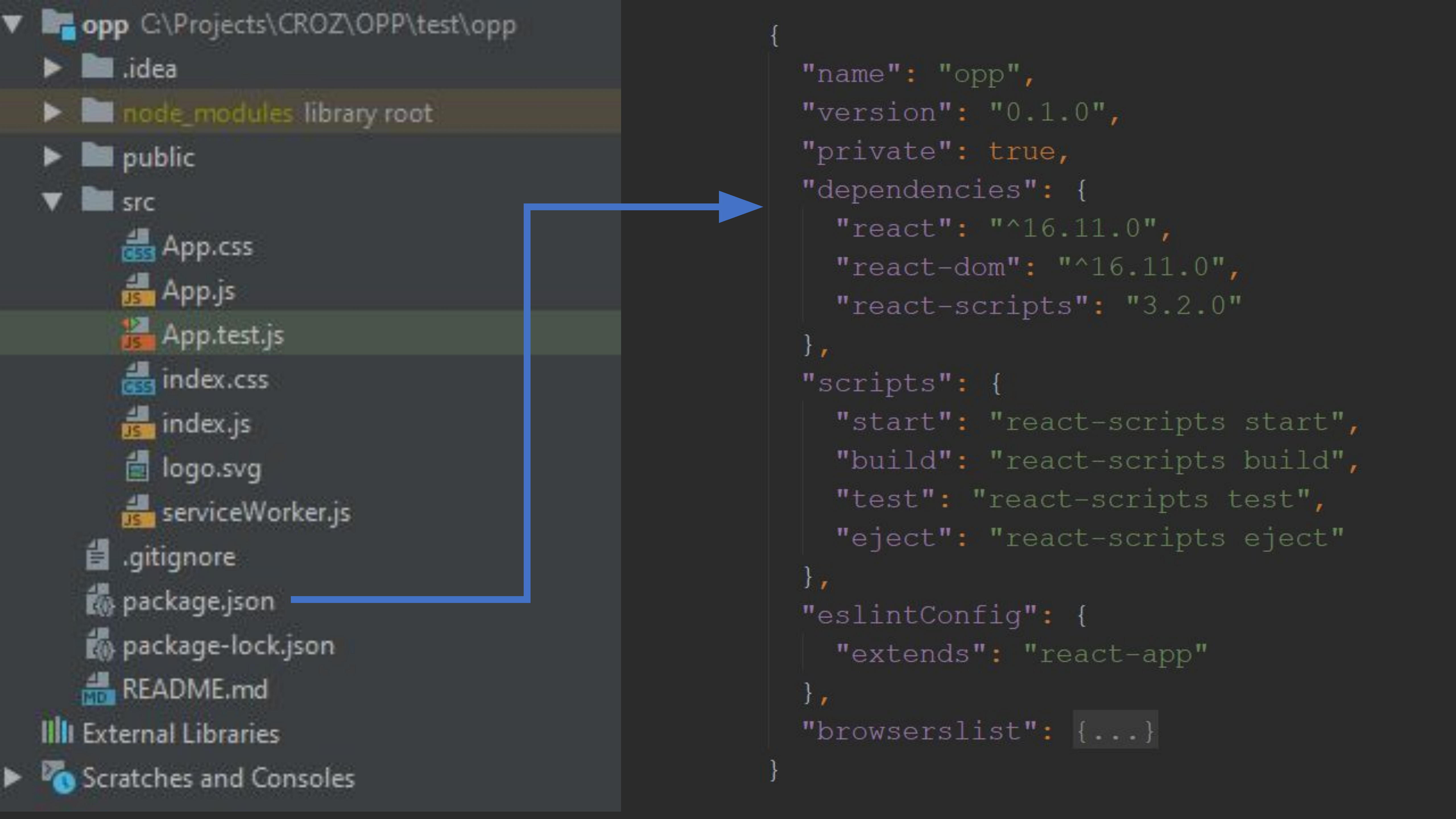
https://reactjs.org/docs/create-a-new-react-app.html#create-react-app

```
jtomic@jtomic MINGW64 /c/Projects/CROZ/OPP
$ npx create-react-app opp
```



Edit `src/App.js` and save to reload.

Learn React



```
{
  "name": "opp",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "react": "^16.11.0",
    "react-dom": "^16.11.0",
    "react-scripts": "3.2.0"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": "react-app"
  },
  "browserslist": {...}
}
```

```
import React from 'react';
import logo from './logo.svg';
import './App.css';

function App() {
  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo" alt="logo" />
        <p>
          Edit <code>src/App.js</code> and save to reload.
        </p>
        <a
          className="App-link"
          href="https://reactjs.org"
          target="_blank"
          rel="noopener noreferrer"
        >
          Learn React
        </a>
      </header>
    </div>
  );
}

export default App;
```

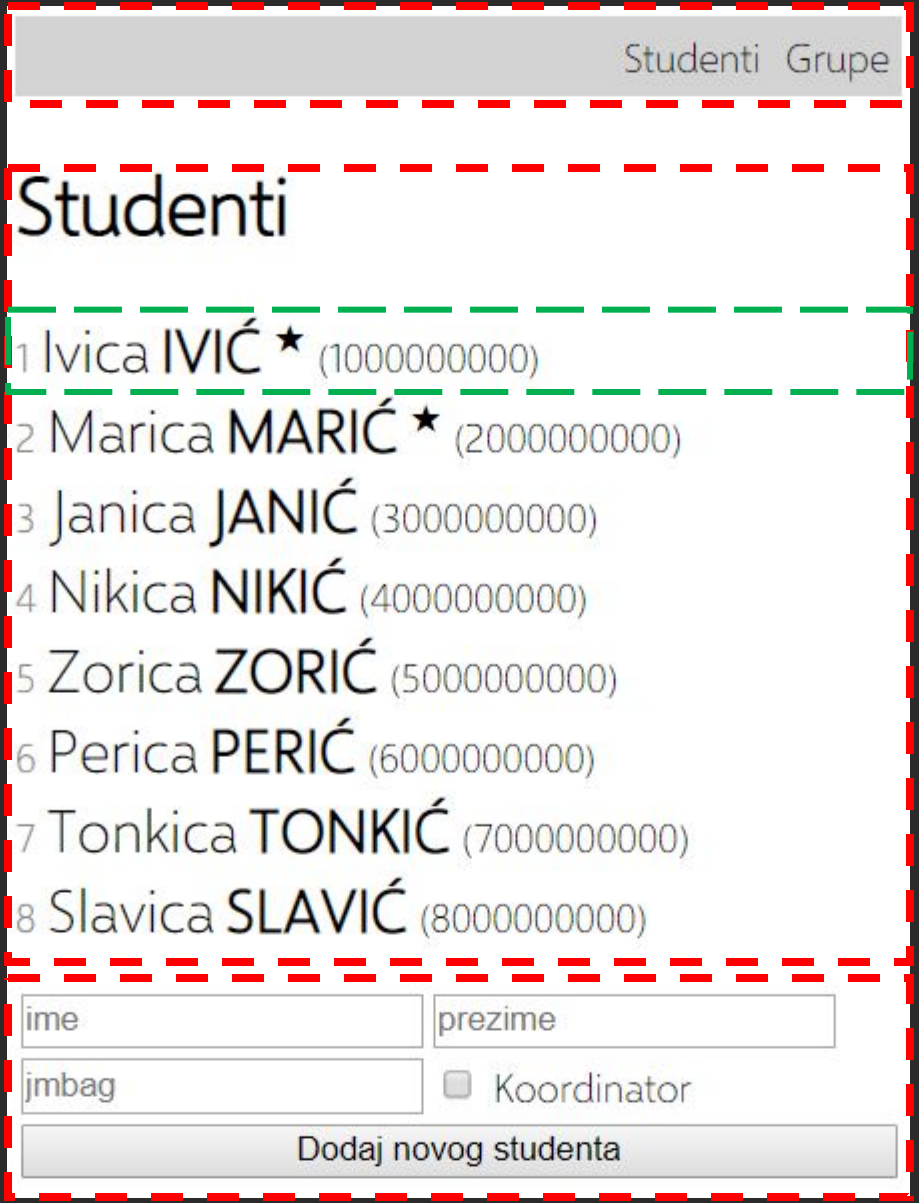
JSX

```
import React from 'react';
import logo from './logo.svg';
import './App.css';

function App() {
  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo" alt="logo" />
        <p>
          Edit <code>src/App.js</code> and save to reload.
        </p>
        <a
          className="App-link"
          href="https://reactjs.org"
          target="_blank"
          rel="noopener noreferrer"
        >
          Learn React
        </a>
      </header>
    </div>
  );
}

export default App;
```

Primjer JS funkcije dodavanja liste

```
function studentHtml(s) {
  return `
    <li id="student-${s.id}" ${s.removed? 'class="removed"' : ''}>
      <span class="id">${s.id}</span>
      <span class="given name">${s.givenName}</span>
      <span class="family name">${s.familyName}</span>
      ${s.lead? '<span class="lead">★</span>' : ''}
      <span class="jmbag">${s.jmbag}</span>
      <span onclick="deleteStudent(${s.id})"
            class="delete">✕</span>
    </li>
  `;
}
```

PRIMJER SUČELJA



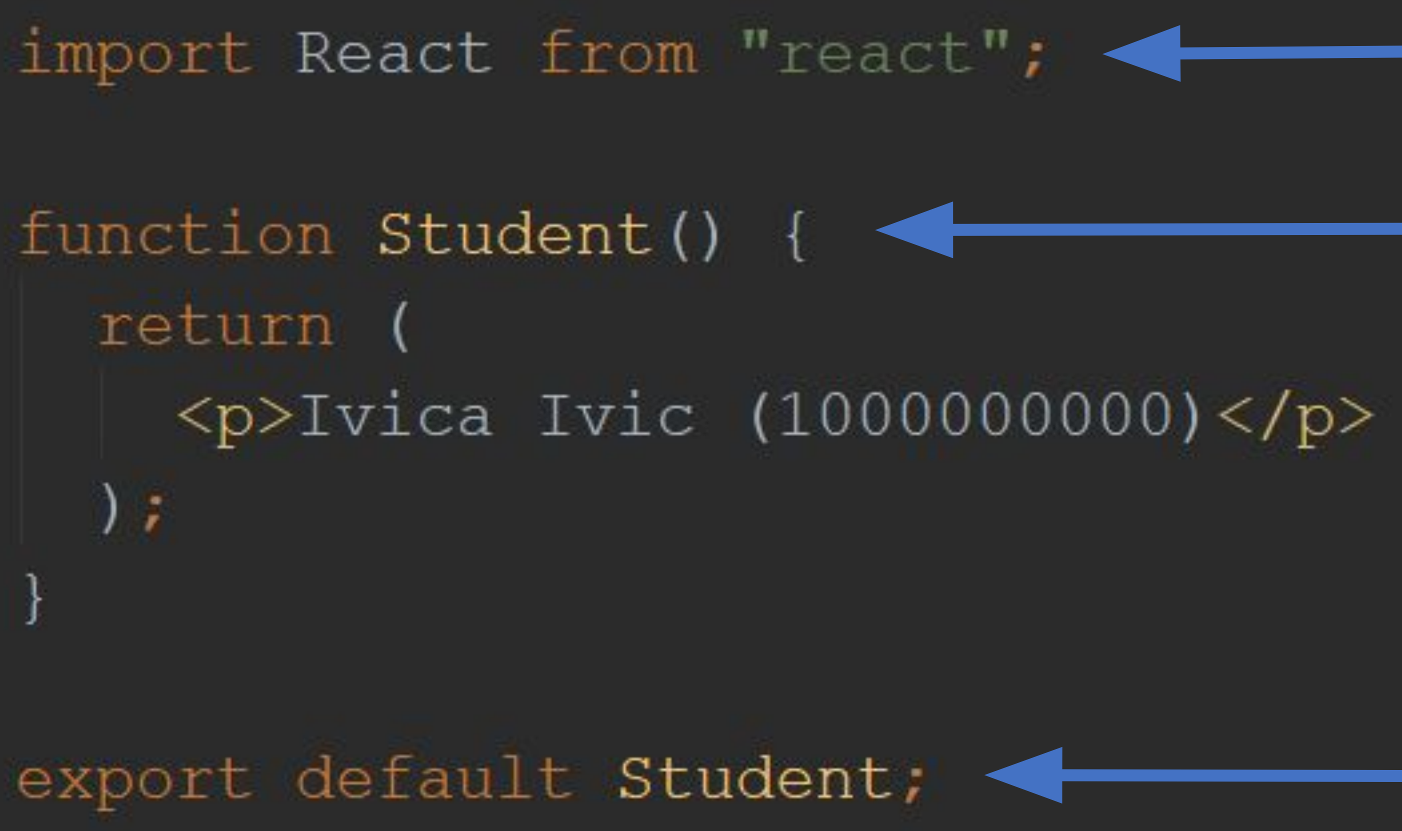
Studenti Grupe

Studenti

1 Ivica IVIĆ ★ (1000000000)
2 Marica MARIĆ ★ (2000000000)
3 Janica JANIĆ (3000000000)
4 Nikica NIKIĆ (4000000000)
5 Zorica ZORIĆ (5000000000)
6 Perica PERIĆ (6000000000)
7 Tonkica TONKIĆ (7000000000)
8 Slavica SLAVIĆ (8000000000)

Izrada prve komponente

Student.js



```
import React from "react";

function Student() {
  return (
    <p>Ivica Ivic (1000000000)</p>
  );
}

export default Student;
```

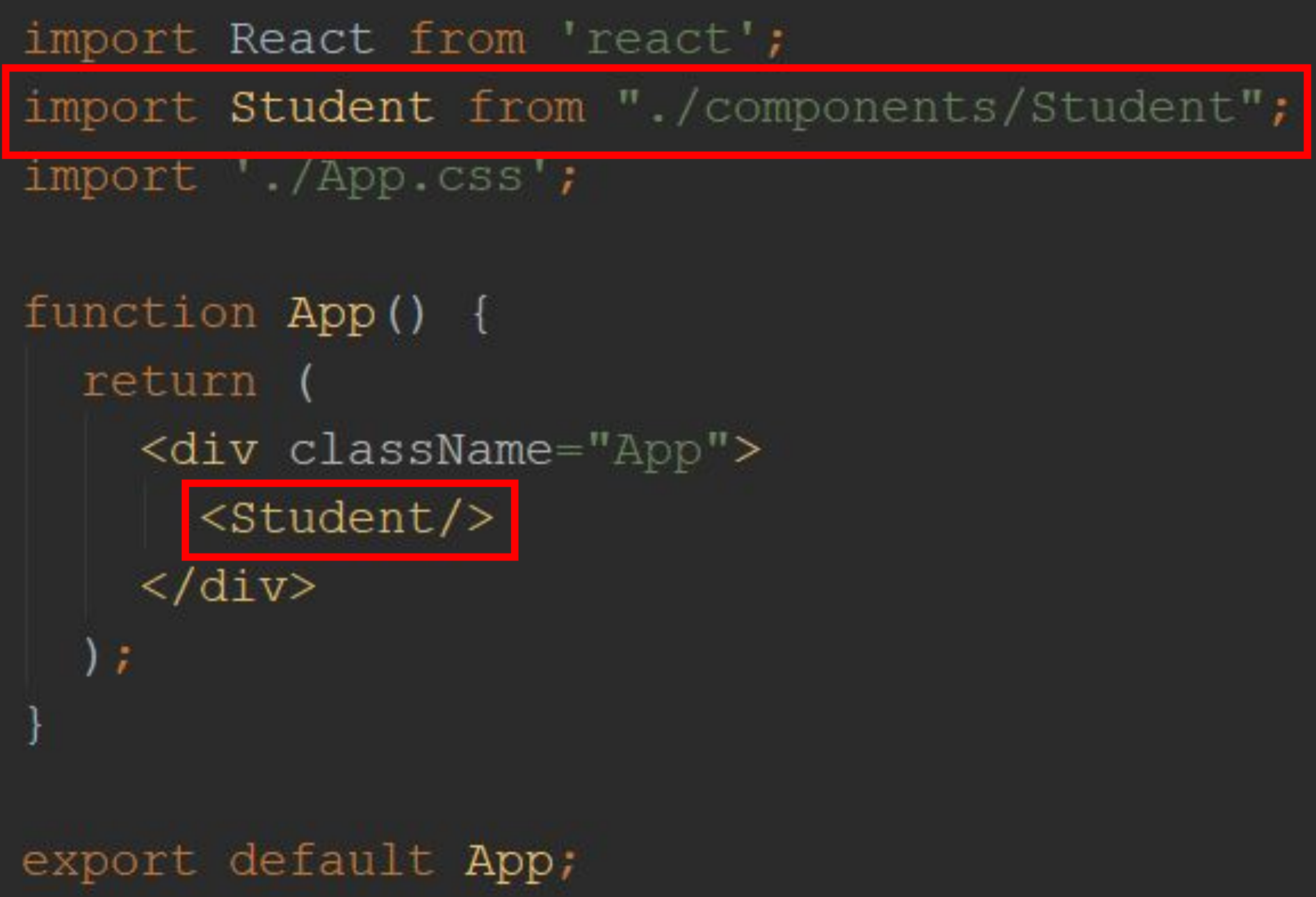
“import” React knjiznice

definiranje komponente

HTML koji opisuje komponentu

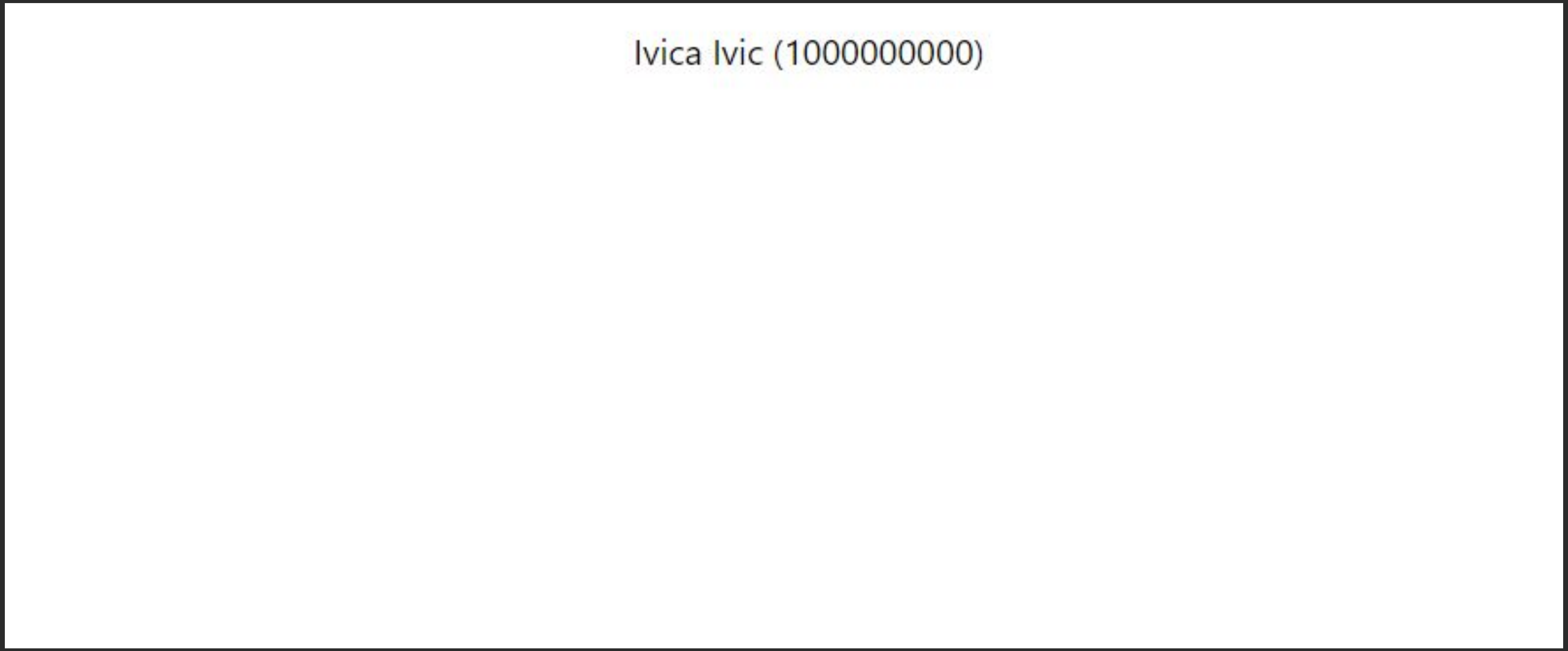
“export” komponente

App.js



```
import React from 'react';
import Student from "./components/Student";
import './App.css';

function App() {
  return (
    <div className="App">
      <Student/>
    </div>
  );
}

export default App;
```

Rezultat



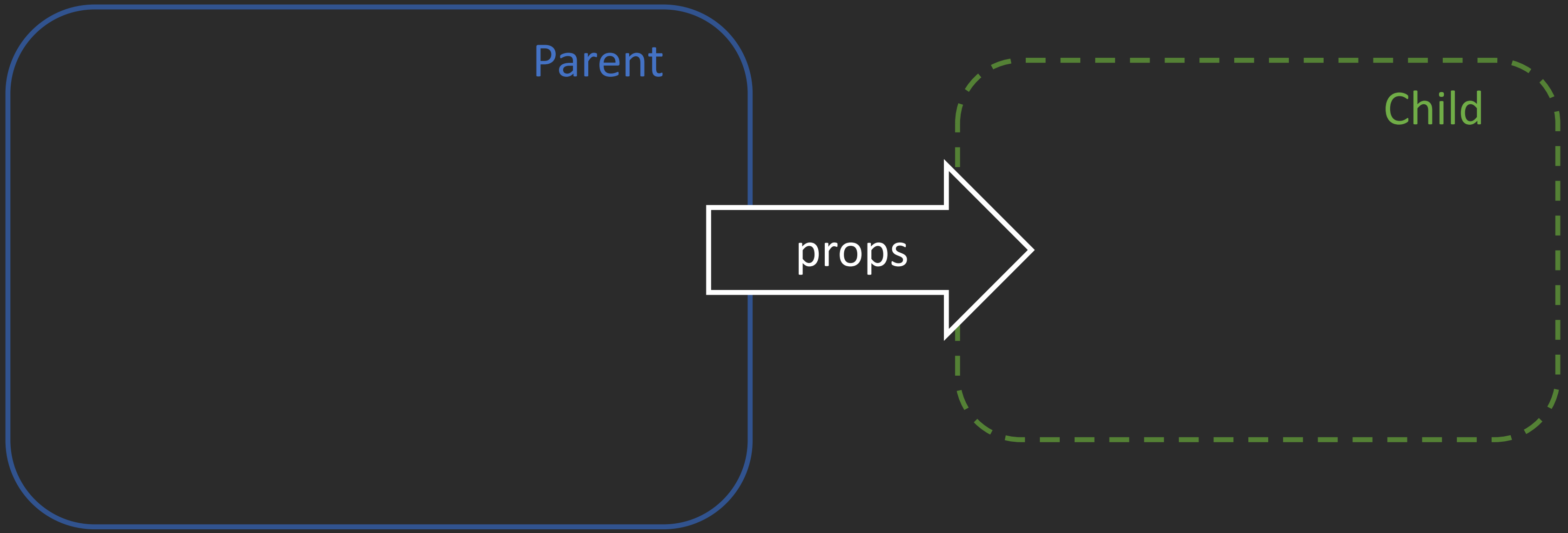
```
Ivica Ivic (1000000000)
```

Svojstva komponente

Još znani i kao propovi

Omogućuju utjecanje na izgled i ponašanje komponente

Komponenta ih ne smije mjenjati



Student.js

```
import React from "react";

function Student(props) {
  const {jmbag, givenName, familyName} = props.student

  return (
    <p>{givenName} {familyName} ({jmbag})</p>
  );
}

export default Student;
```

Citanje svojstva

App.js

```
function App() {
  const student = {
    givenName: 'Ivica',
    familyName: 'Ivic',
    jmbag: '1000000000'
  };

  return (
    <div className="App">
      <Student student={student}/>
    </div>
  );
}
```

Definiranje varijable

Predaja svojstva

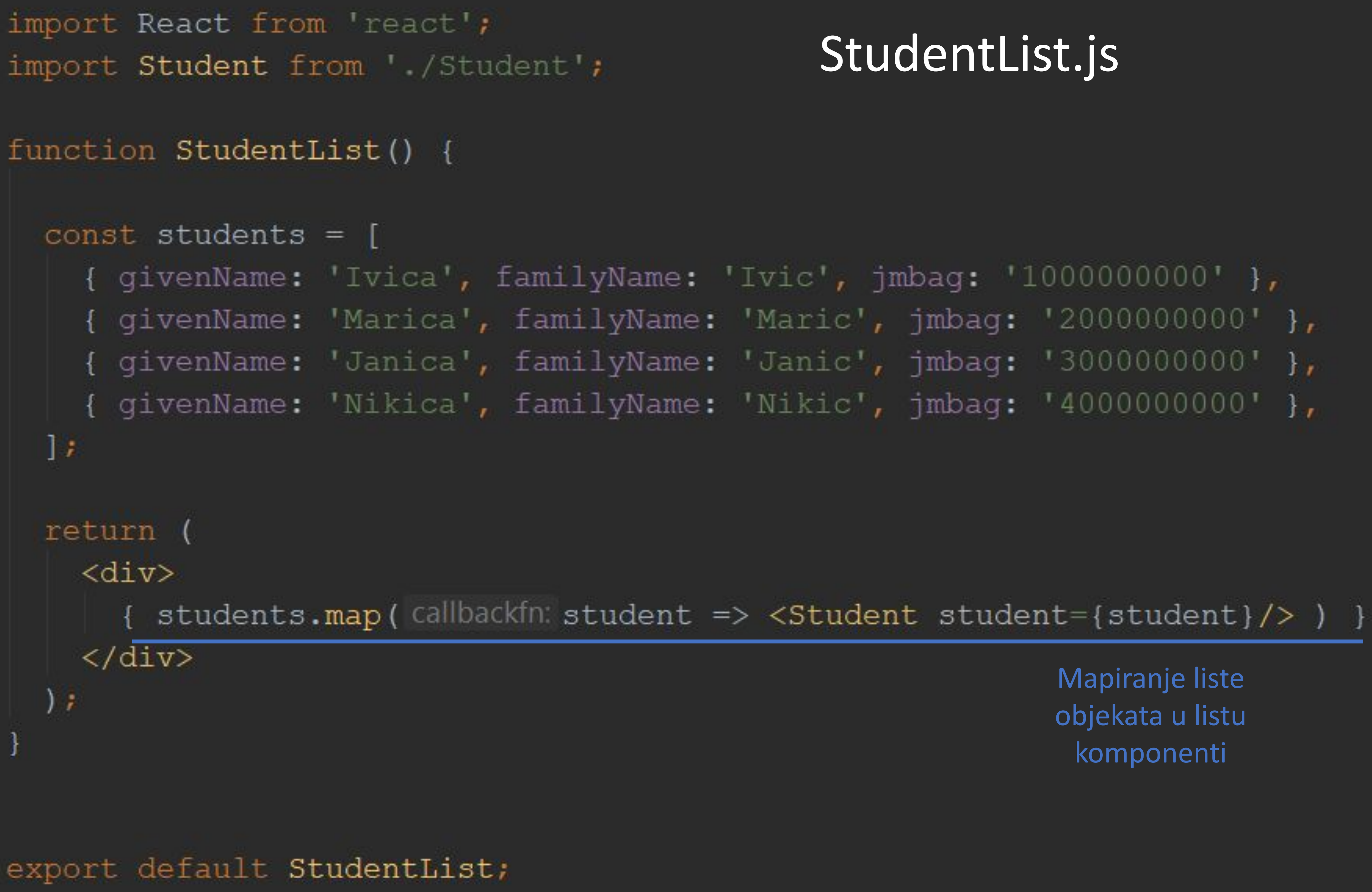
Destruktuiranje objekta

```
const {jmbag, givenName, familyName} = props.student;
```

```
const jmbag = props.student.jmbag;
const givenName = props.student.givenName;
const familyName = props.student.familyName;
```

Lista studenata

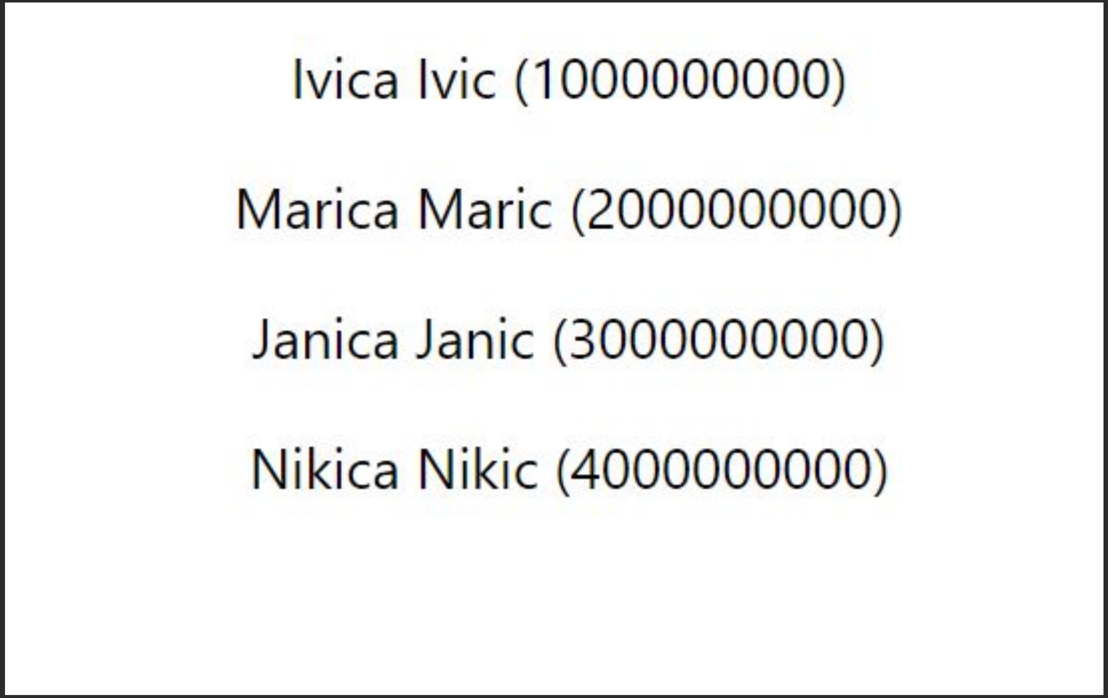
StudentList.js



```
import React from 'react';
import Student from './Student';

function StudentList() {

  const students = [
    { givenName: 'Ivica', familyName: 'Ivic', jmbag: '1000000000' },
    { givenName: 'Marica', familyName: 'Maric', jmbag: '2000000000' },
    { givenName: 'Janica', familyName: 'Janic', jmbag: '3000000000' },
    { givenName: 'Nikica', familyName: 'Nikic', jmbag: '4000000000' },
  ];

  return (
    <div>
      { students.map( callbackfn: student => <Student student={student}/> ) }
    </div>
  );
}


export default StudentList;
```

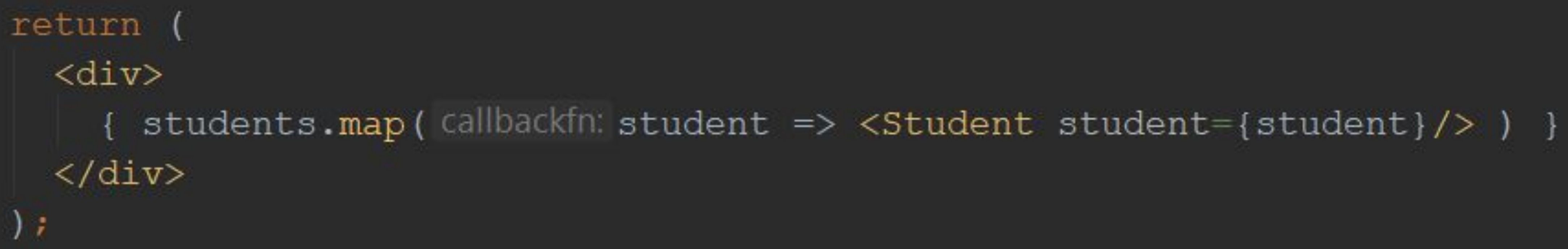
Mapiranje liste objekata u listu komponenti

Rezultat

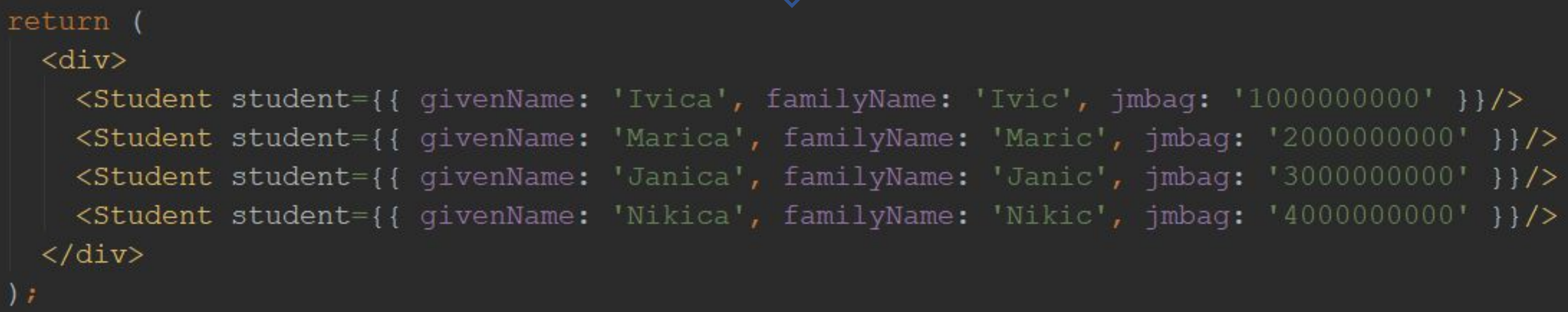


```
Ivica Ivic (1000000000)

Marica Maric (2000000000)

Janica Janic (3000000000)

Nikica Nikic (4000000000)
```

Liste komponenti

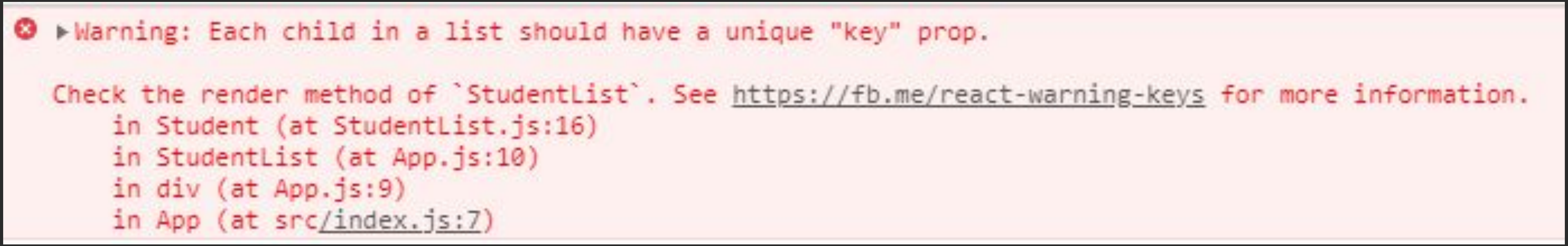


```
return (
  <div>
    { students.map( callbackfn: student => <Student student={student}/> ) }
  </div>
);
```



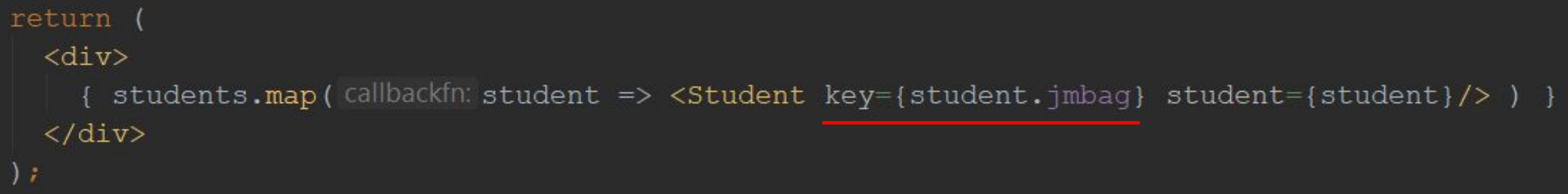
```
return (
  <div>
    <Student student={{ givenName: 'Ivica', familyName: 'Ivic', jmbag: '1000000000' }}/>
    <Student student={{ givenName: 'Marica', familyName: 'Maric', jmbag: '2000000000' }}/>
    <Student student={{ givenName: 'Janica', familyName: 'Janic', jmbag: '3000000000' }}/>
    <Student student={{ givenName: 'Nikica', familyName: 'Nikic', jmbag: '4000000000' }}/>
  </div>
);
```

Greška u konzoli ?



```
▶Warning: Each child in a list should have a unique "key" prop.

Check the render method of `StudentList`. See https://fb.me/react-warning-keys for more information.
    in Student (at StudentList.js:16)
    in StudentList (at App.js:10)
    in div (at App.js:9)
    in App (at src/index.js:7)
```

Liste komponenti



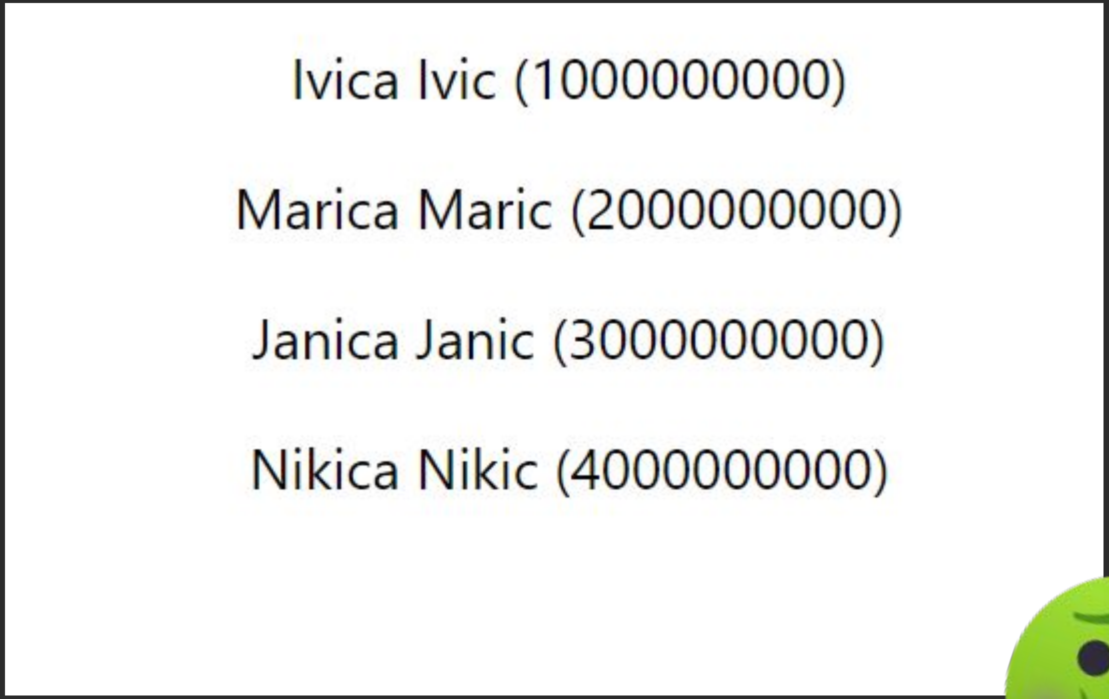
```
return (
  <div>
    { students.map( callbackfn: student => <Student key={student.jmbag} student={student}/> ) }
  </div>
);
```

Optimalno azuriranje isrctavanja listi

Koristiti atribut objekta koji je unikatan (id, jmbag…)

Ako takvog nema, koristiti indeks u listi

Stiliziranje komponenti



```
Ivica Ivic (1000000000)

Marica Maric (2000000000)

Janica Janic (3000000000)

Nikica Nikic (4000000000)
```

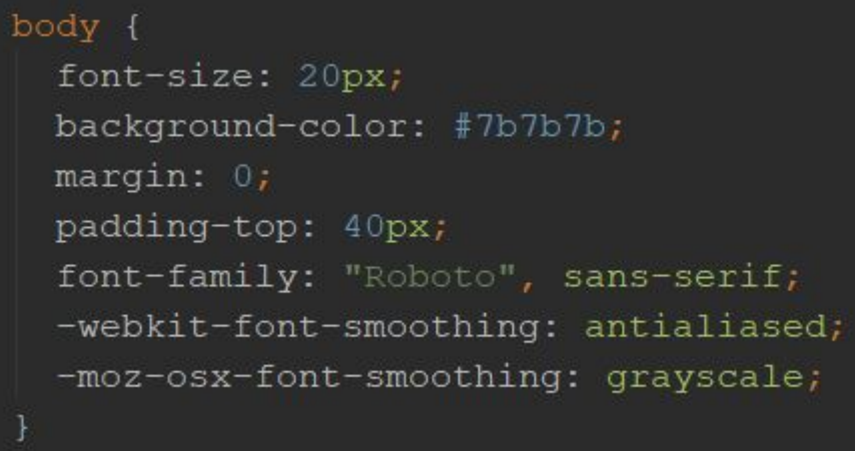
StudentList.css

```css
.StudentList {
    margin: 10px;
    padding: 15px;
    box-shadow: 0 8px 10px 0 rgba(0, 0, 0, 0.6);
    display: block;
    background-color: white;
}

.StudentList h2 {
    margin: 0 0 5px 0;
}
```

StudentList.js

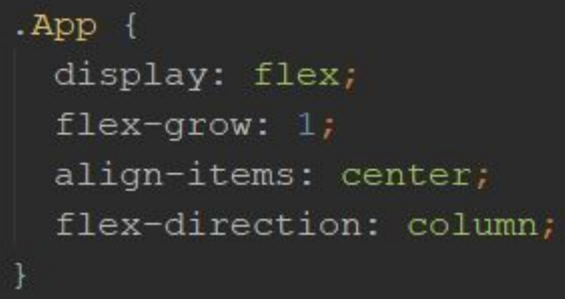
```javascript
import React from 'react';
import Student from './Student';
import './StudentList.css';

function StudentList() {

  const students = [
    { givenName: 'Ivica', familyName: 'Ivic', jmbag: '1000000
    { givenName: 'Marica', familyName: 'Maric', jmbag: '20000
    { givenName: 'Janica', familyName: 'Janic', jmbag: '30000
    { givenName: 'Nikica', familyName: 'Nikic', jmbag: '40000
  ];

  return (
    <div className='StudentList'>
      {
        students.map( callbackfn: student =>
          <Student key={student.jmbag} student={student}/>
        )
      }
    </div>
  );
}


export default StudentList;
```

index.css



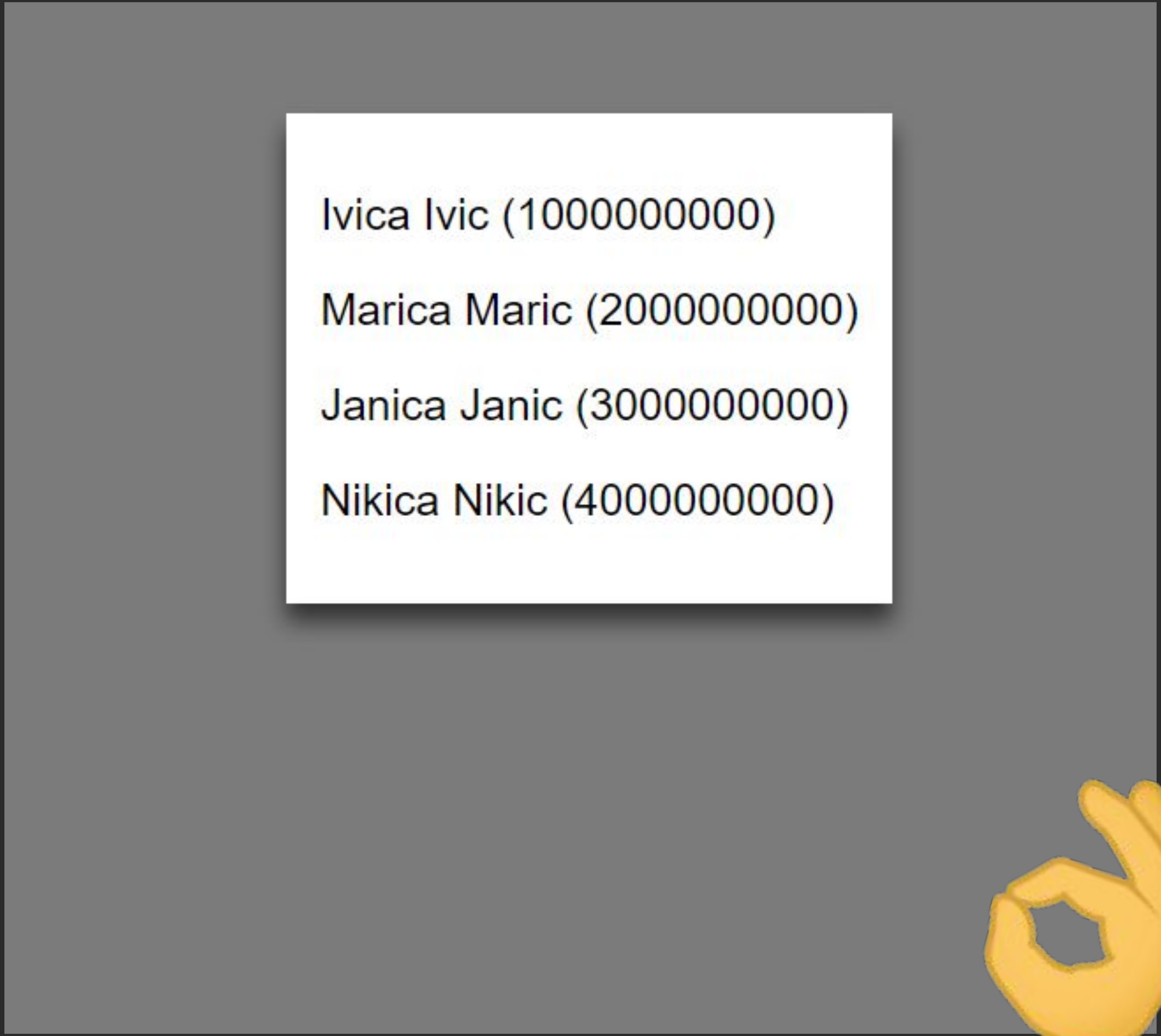
```css
body {
  font-size: 20px;
  background-color: #7b7b7b;
  margin: 0;
  padding-top: 40px;
  font-family: "Roboto", sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
}
```

App.css



```css
.App {
  display: flex;
  flex-grow: 1;
  align-items: center;
  flex-direction: column;
}
```

Rezultat



```
Ivica Ivic (1000000000)

Marica Maric (2000000000)

Janica Janic (3000000000)

Nikica Nikic (4000000000)
```

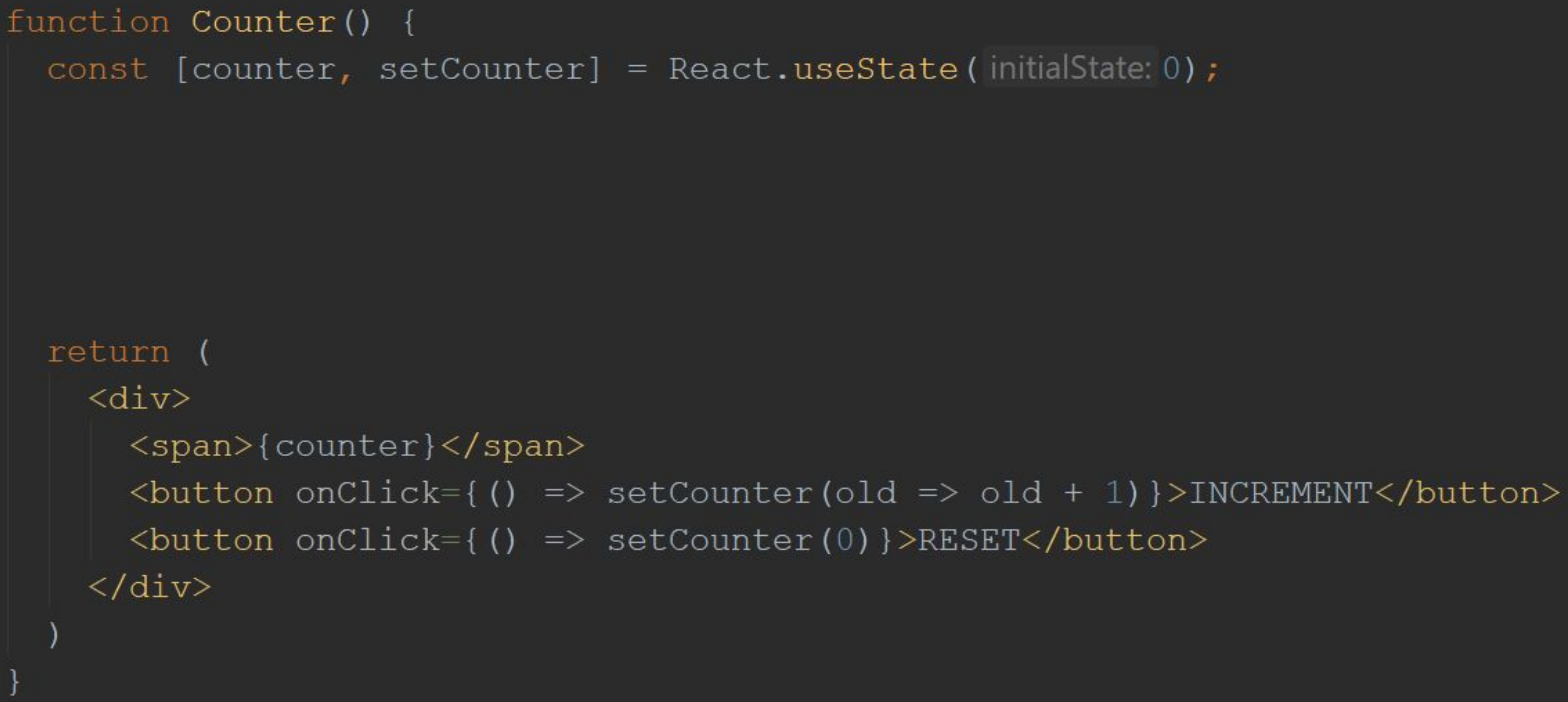
Dohvat pravih podataka s backenda

No prije toga…

1. State Hook

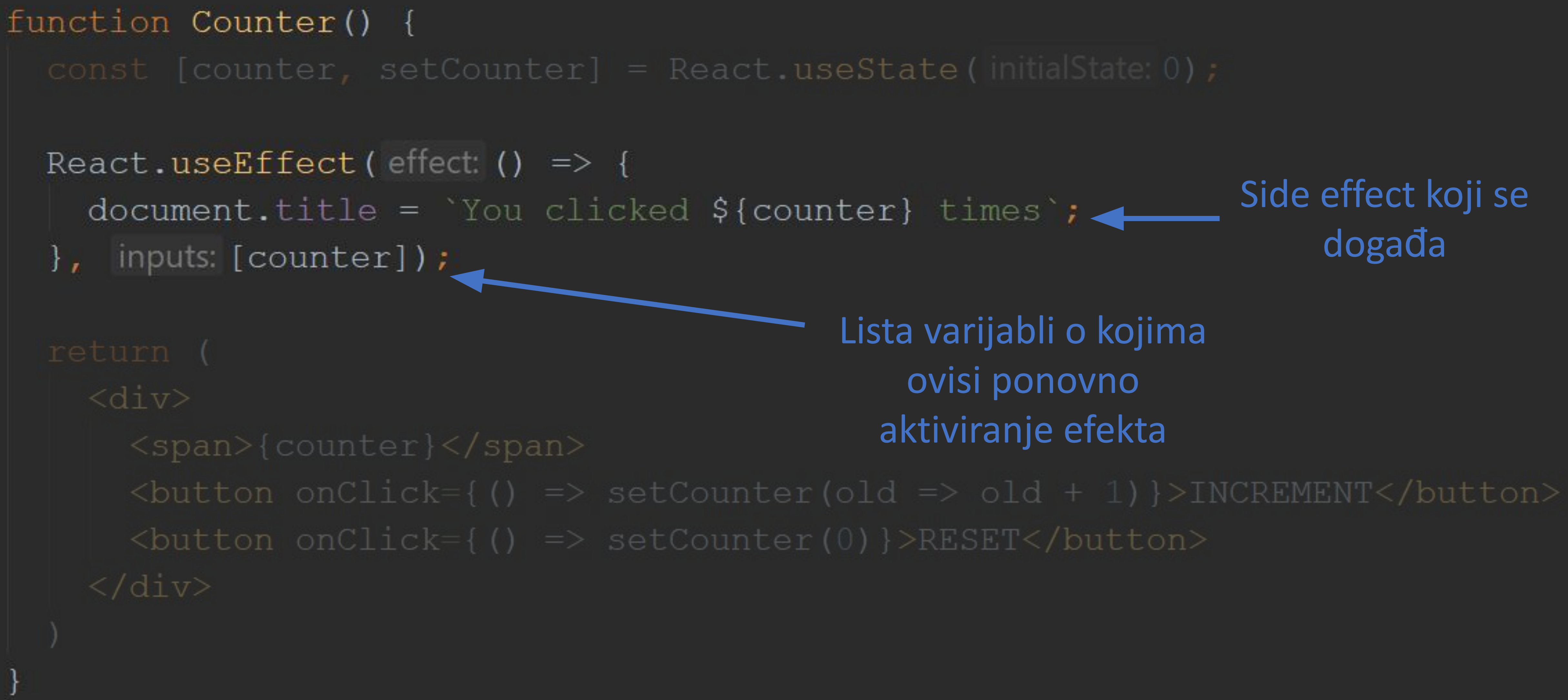
```
function Counter() {









  return (
    <div>



    </div>
  )
}
```

Effect hook



```
function Counter() {
  const [counter, setCounter] = React.useState( initialState: 0);






  return (
    <div>
      <span>{counter}</span>
      <button onClick={() => setCounter(old => old + 1)}>INCREMENT</button>
      <button onClick={() => setCounter(0)}>RESET</button>
    </div>
  )
}
```

Effect hook



```
function Counter() {
  const [counter, setCounter] = React.useState( initialState: 0);

  React.useEffect( effect: () => {
    document.title = `You clicked ${counter} times`;
  },  inputs: [counter]);

  return (
    <div>
      <span>{counter}</span>
      <button onClick={() => setCounter(old => old + 1)}>INCREMENT</button>
      <button onClick={() => setCounter(0)}>RESET</button>
    </div>
  )
}
```

Side effect koji se događa

Lista varijabli o kojima ovisi ponovno aktiviranje efekta

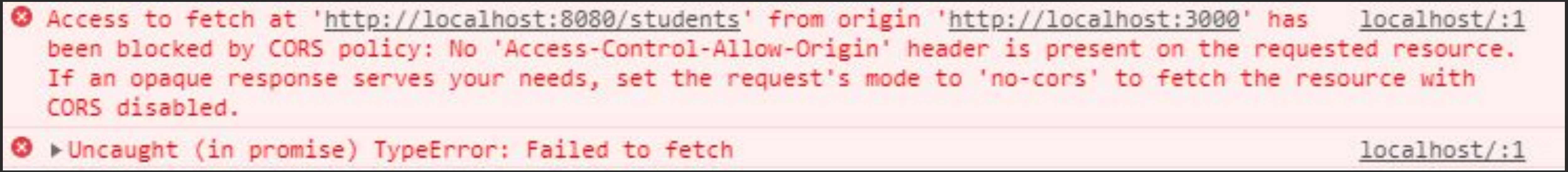
Dostupno na backendu

Putanja:	/students
Metoda:	GET
Vraća:	Listu svih studenata

```
function StudentList() {









  return (
    <div className='StudentList'>
      {
        students.map( callbackfn: student =>
          <Student key={student.jmbag} student={student}/>
        )
      }
    </div>
  );
}
```

StudentList.js

Problem

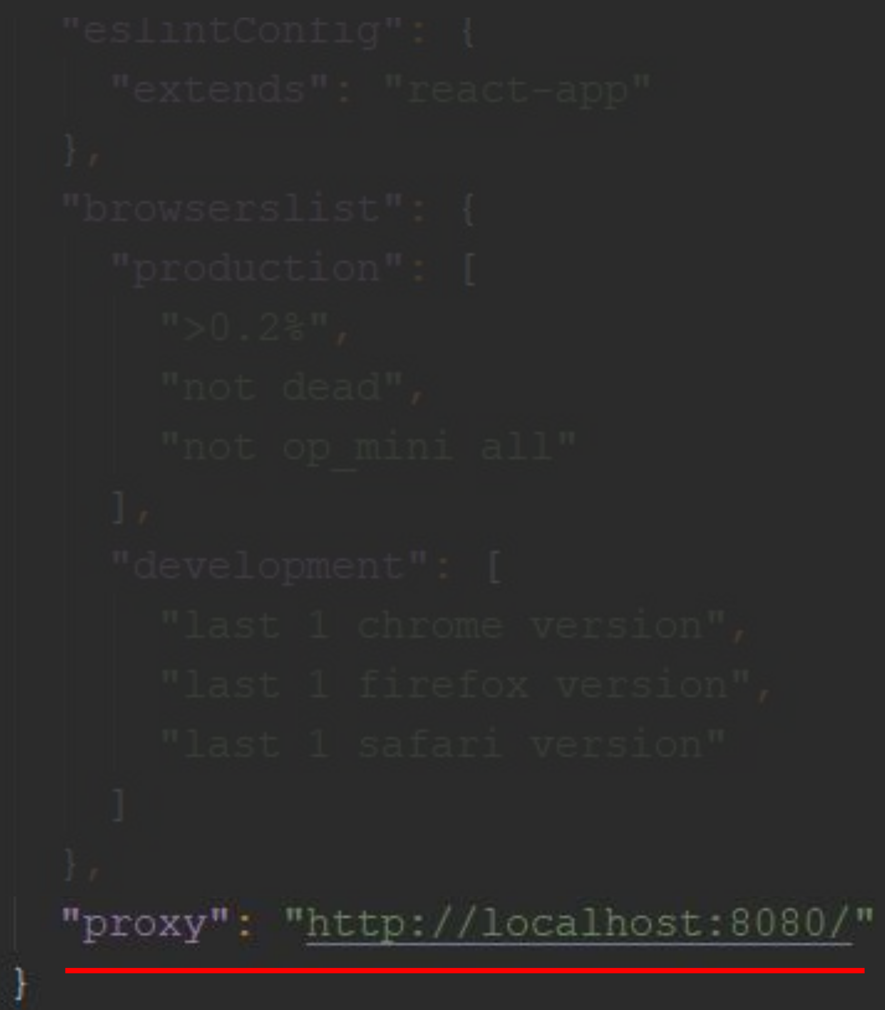


```
Access to fetch at 'http://localhost:8080/students' from origin 'http://localhost:3000' has    localhost/:1
been blocked by CORS policy: No 'Access-Control-Allow-Origin' header is present on the requested resource.
If an opaque response serves your needs, set the request's mode to 'no-cors' to fetch the resource with
CORS disabled.
▶Uncaught (in promise) TypeError: Failed to fetch                                               localhost/:1
```

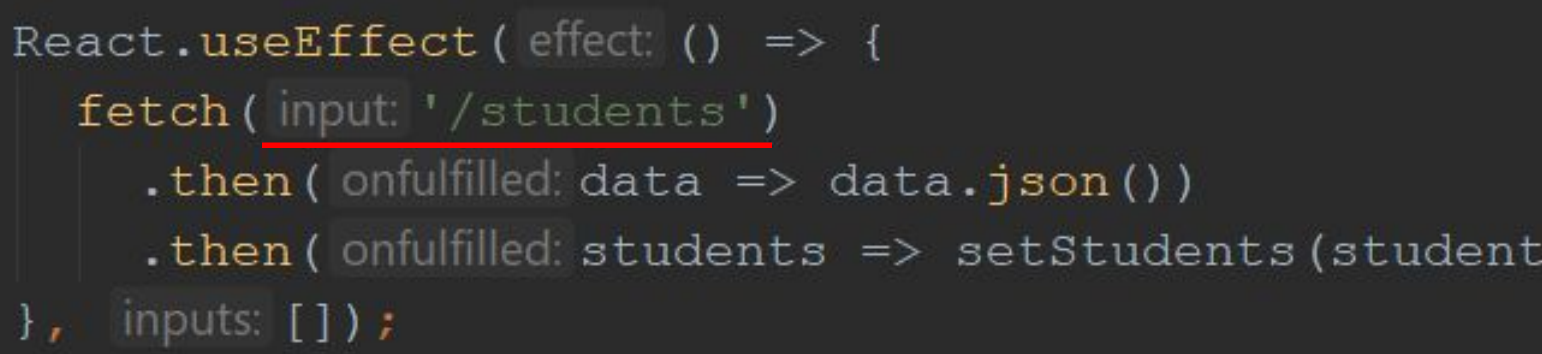
???

Rješenje

package.json

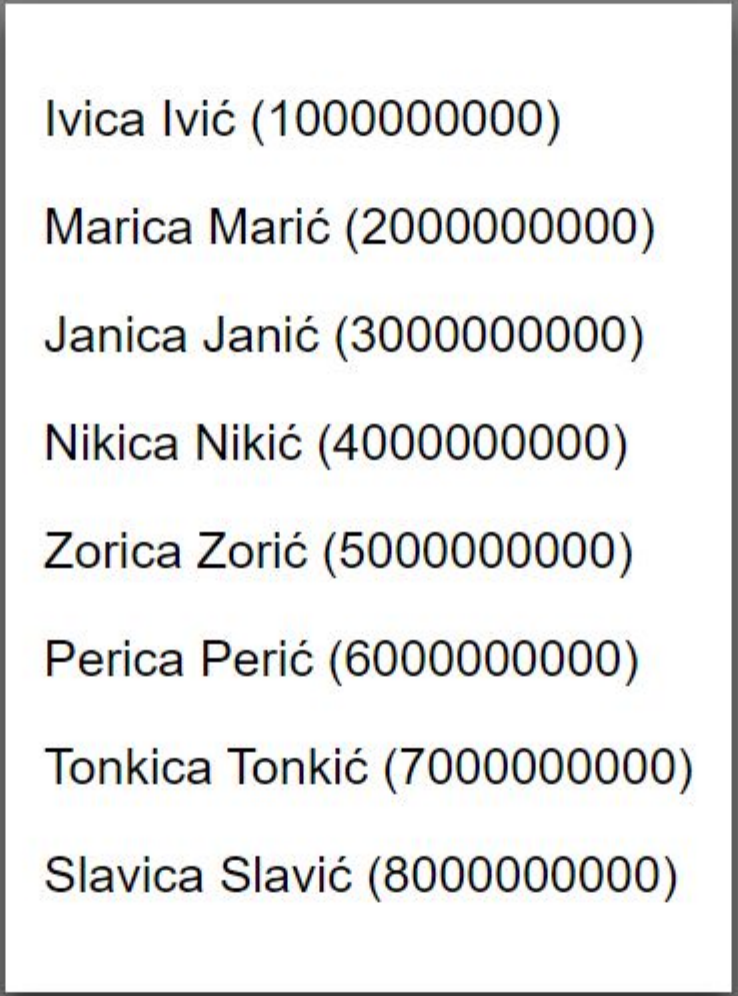


```
  "eslintConfig": {
    "extends": "react-app"
  },
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  },
  "proxy": "http://localhost:8080/"
}
```



```
React.useEffect( effect: () => {
  fetch( input: '/students')
    .then( onfulfilled: data => data.json())
    .then( onfulfilled: students => setStudents(student
}, inputs: []);
```

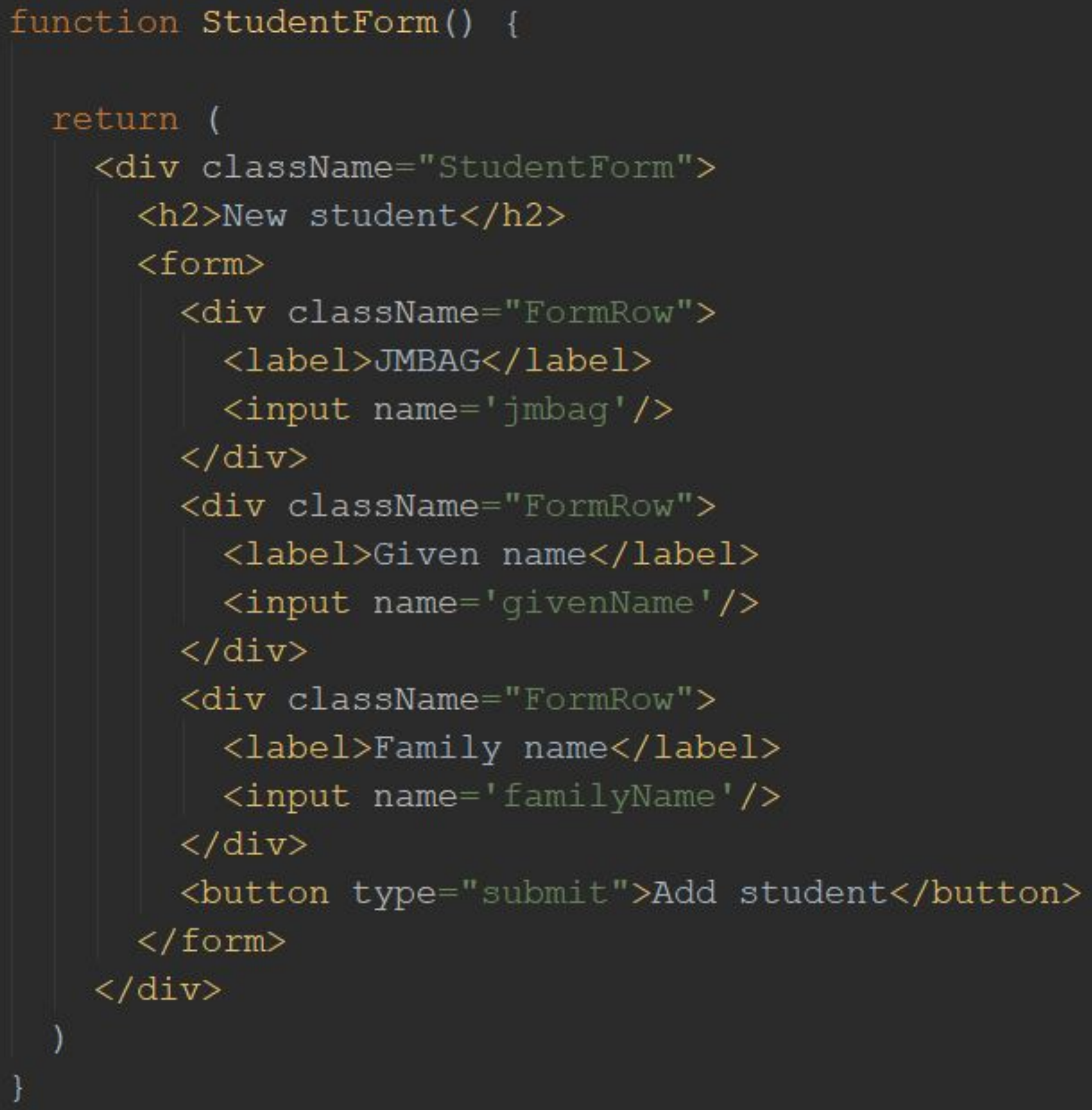
Rezultat



```
Ivica Ivić (1000000000)

Marica Marić (2000000000)

Janica Janić (3000000000)

Nikica Nikić (4000000000)

Zorica Zorić (5000000000)

Perica Perić (6000000000)

Tonkica Tonkić (7000000000)

Slavica Slavić (8000000000)
```

Dodavanje novih studenata

StudentForm.js



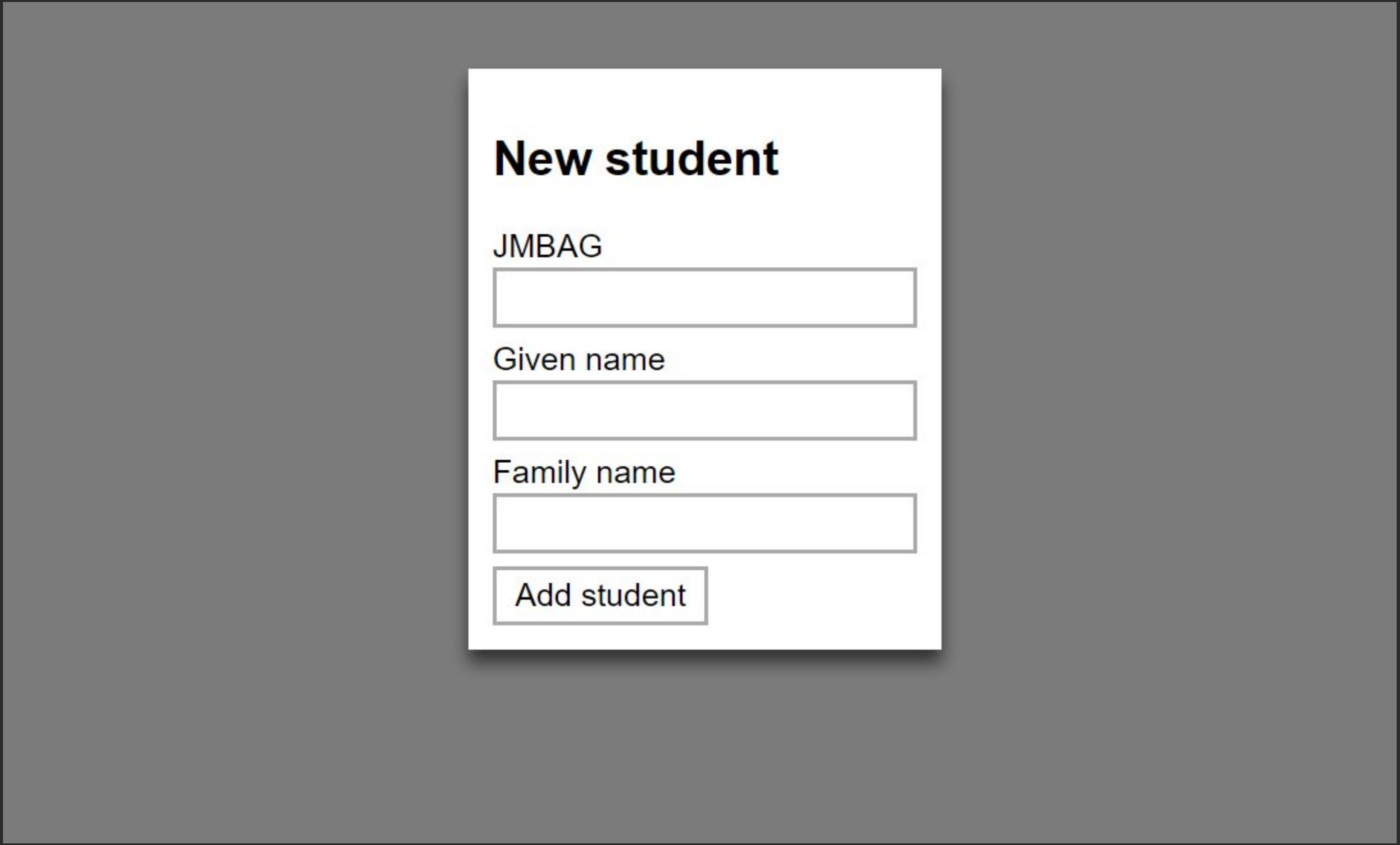
```
function StudentForm() {

  return (
    <div className="StudentForm">
      <h2>New student</h2>
      <form>
        <div className="FormRow">
          <label>JMBAG</label>
          <input name='jmbag'/>
        </div>
        <div className="FormRow">
          <label>Given name</label>
          <input name='givenName'/>
        </div>
        <div className="FormRow">
          <label>Family name</label>
          <input name='familyName'/>
        </div>
        <button type="submit">Add student</button>
      </form>
    </div>
  )
}
```

StudentForm.css

```css
.FormRow {
    margin: 8px 0;
}

.StudentForm .FormRow:first-child {
    margin-top: 0;
}

.FormRow label {
    display: block;
    margin-bottom: 2px;
}

.FormRow input {
    font-size: 20px;
    padding: 5px;
    border: 2px solid #aaa;
}

.StudentForm button {
    background: none;
    border: 2px solid #aaa;
    padding: 5px 12px;
    font-size: 20px;
    cursor: pointer;
}
```

```css
.StudentForm button:hover {
    background-color: #aaa;
}

.StudentForm button:disabled {
    color: #ddd;
}

.StudentForm {
    margin: 10px;
    padding: 15px;
    box-shadow: 0 8px 10px 0 rgba(0, 0, 0, 0.6);
    display: block;
    background-color: white;
}
```

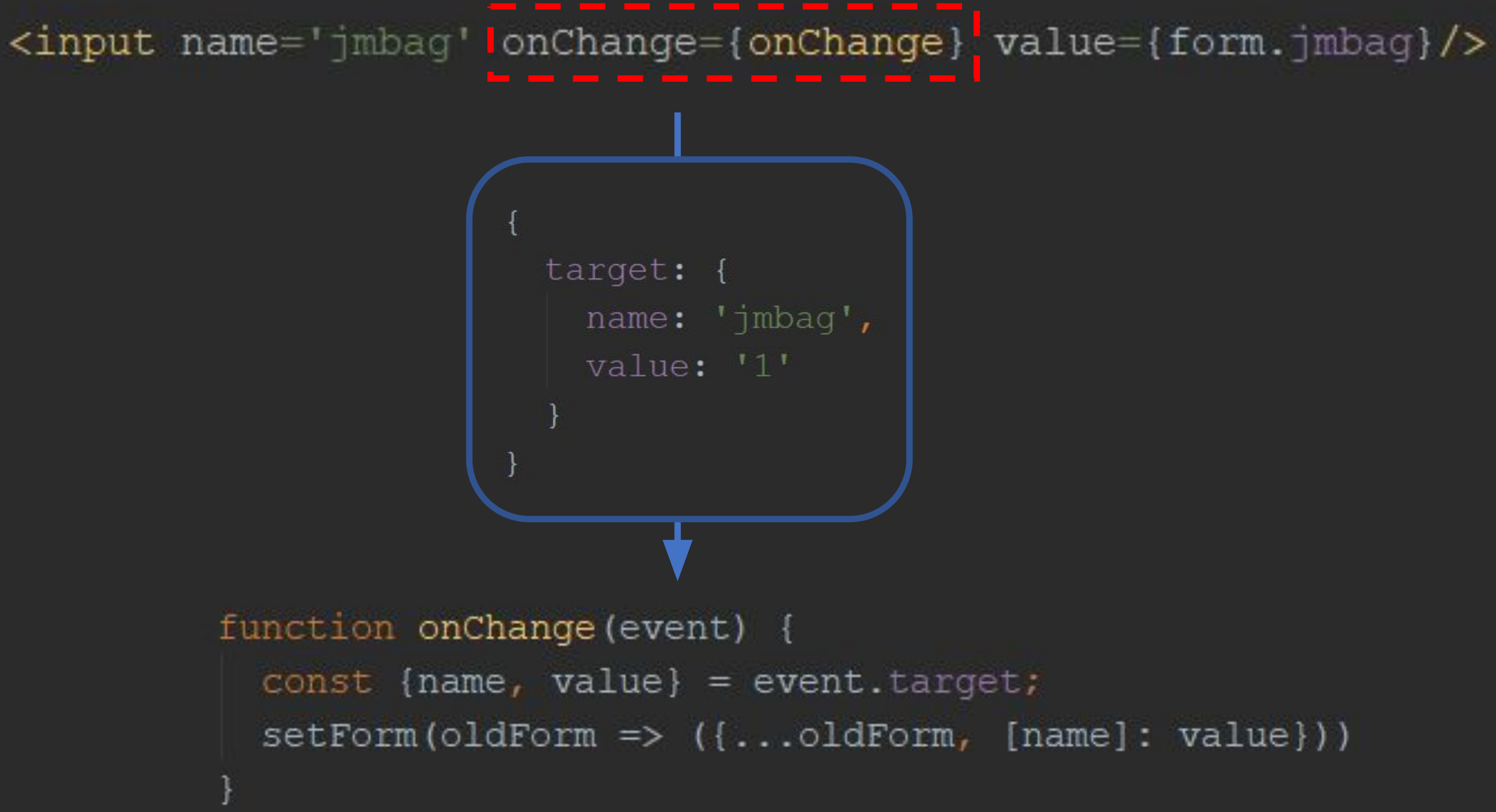
Rezultat



Dodavanje rucnog pracenja izmjena forme

```
function StudentForm() {







  return (
    <div className="StudentForm">
      <h2>New student</h2>
      <form>
        <div className="FormRow">
          <label>JMBAG</label>
          <input name='jmbag'                                       />
        </div>
        <div className="FormRow">
          <label>Given name</label>
          <input name='givenName'                                           />
        </div>
        <div className="FormRow">
          <label>Family name</label>
          <input name='familyName'                                            />
        </div>
        <button type="submit">Add student</button>
      </form>
    </div>
  )
}
```

Klik broja “1” na tipkovnici u jmbag polju



```
<input name='jmbag' onChange={onChange} value={form.jmbag}/>
```

```
{
  target: {
    name: 'jmbag',
    value: '1'
  }
}
```

```
function onChange(event) {
  const {name, value} = event.target;
  setForm(oldForm => ({...oldForm, [name]: value}))
}
```

```
function onChange(event) {
  const {name, value} = event.target;
  setForm(oldForm => ({...oldForm, [name]: value}))
}
```

```
setForm( form: {givenName: '', familyName: '', jmbag: '1'})
```

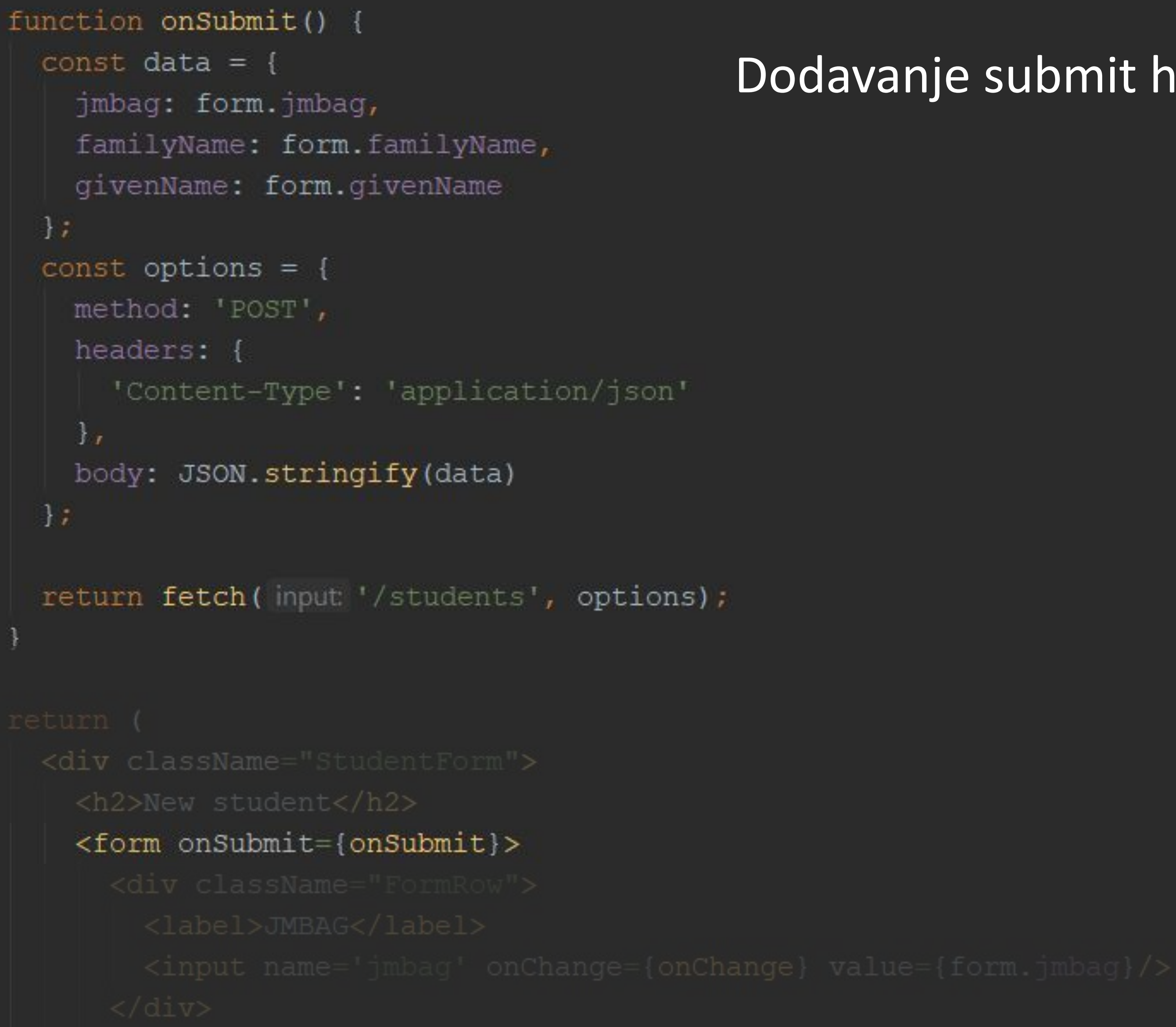
```
<input name='jmbag' onChange={onChange} value={form.jmbag}/>
```

U polju se prikaze broj 1

Dostupno na backendu

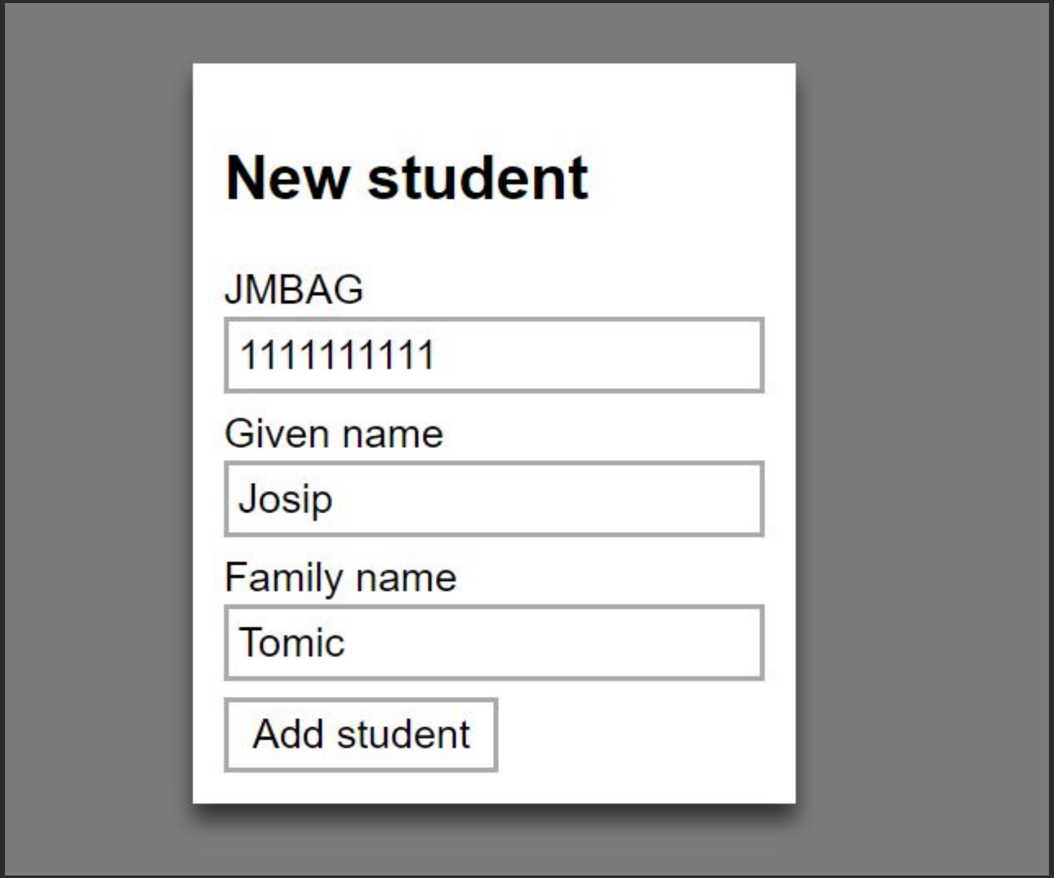
Putanja:	/students
Metoda:	POST
Body:	Objekt koji sadržava ime, prezime i jmbag
Content-Type:	application/json
Vraća:	Kreiranog studenta

Dodavanje submit handlera



```
function onSubmit() {
  const data = {
    jmbag: form.jmbag,
    familyName: form.familyName,
    givenName: form.givenName
  };
  const options = {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json'
    },
    body: JSON.stringify(data)
  };

  return fetch( input: '/students', options);
}

return (
  <div className="StudentForm">
    <h2>New student</h2>
    <form onSubmit={onSubmit}>
      <div className="FormRow">
        <label>JMBAG</label>
        <input name='jmbag' onChange={onChange} value={form.jmbag}/>
      </div>
```

Ako isprobamo dodati studenta na klik gumba “Add student” stranica se refresha, a u url-u dobijemo sljedeće:


New student

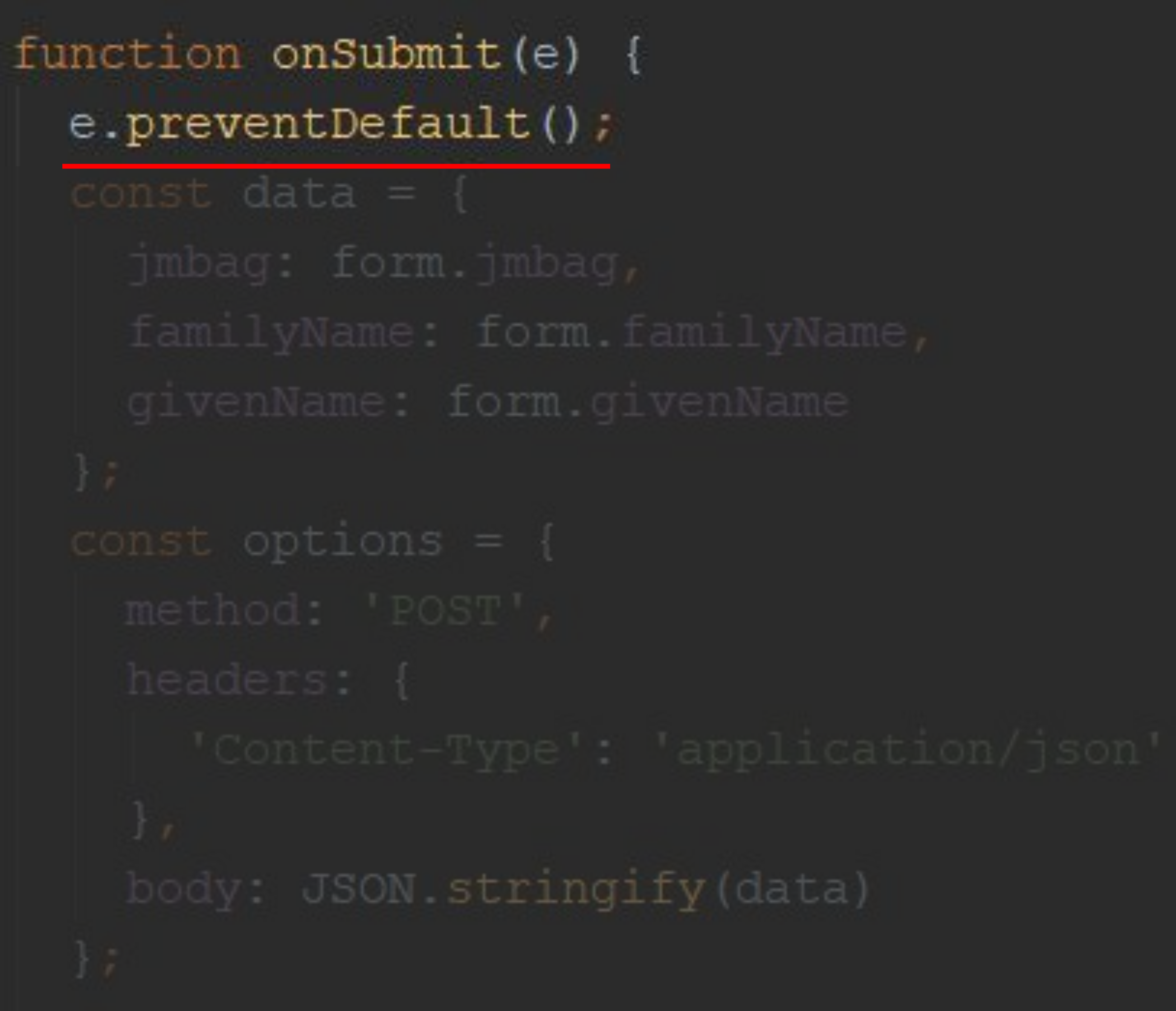
JMBAG: 1111111111

Given name: Josip

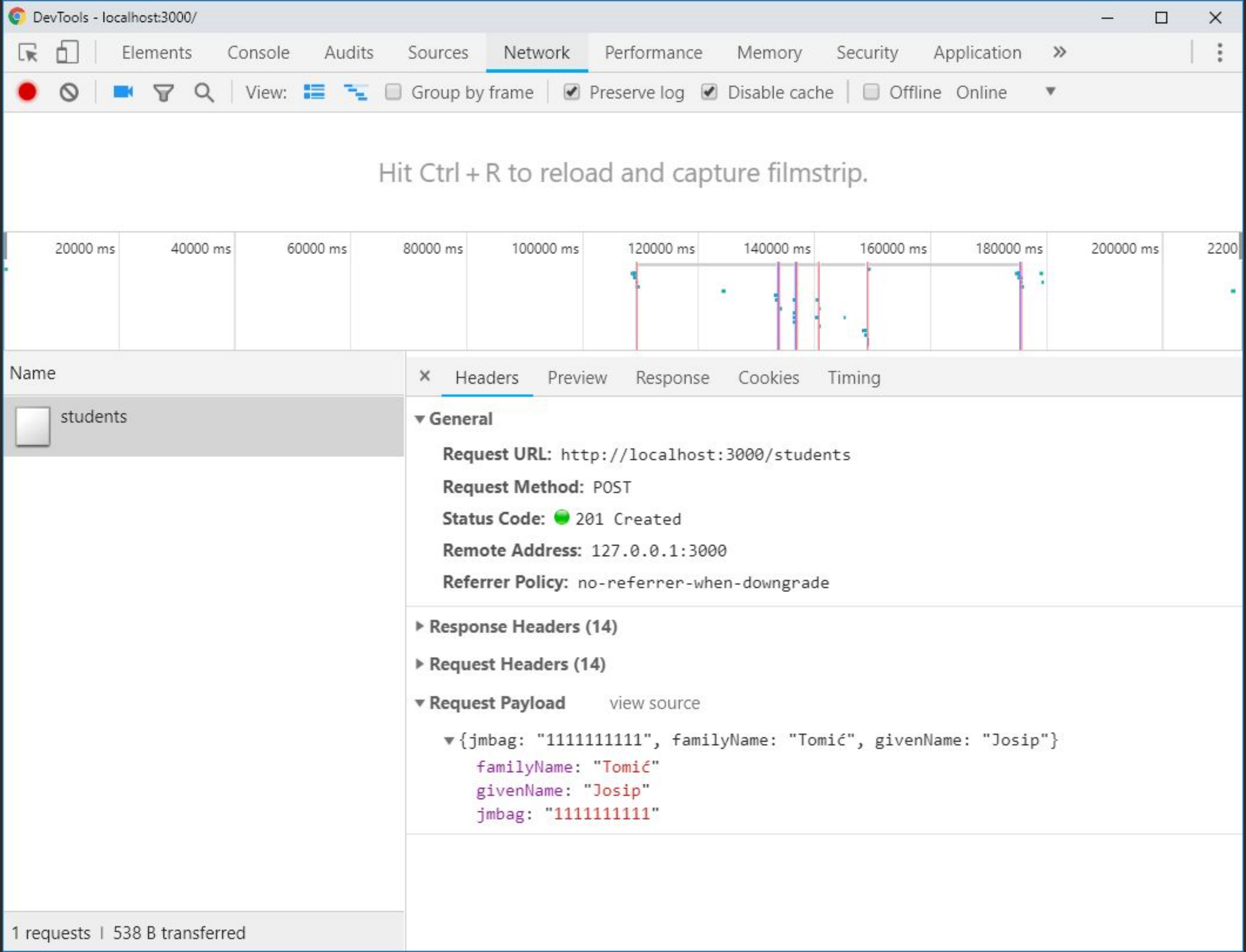
Family name: Tomic

http://localhost:3000/?jmbag=1111111111&givenName=Josip&familyName=Tomic

Sprječavanje zadanog ponašanja forme



```
function onSubmit(e) {
  e.preventDefault();
  const data = {
    jmbag: form.jmbag,
    familyName: form.familyName,
    givenName: form.givenName
  };
  const options = {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json'
    },
    body: JSON.stringify(data)
  };
```



General

Request URL: `http://localhost:3000/students`

Request Method: `POST`

Status Code: ● `201 Created`

Remote Address: `127.0.0.1:3000`

Referrer Policy: `no-referrer-when-downgrade`

Response Headers (14)

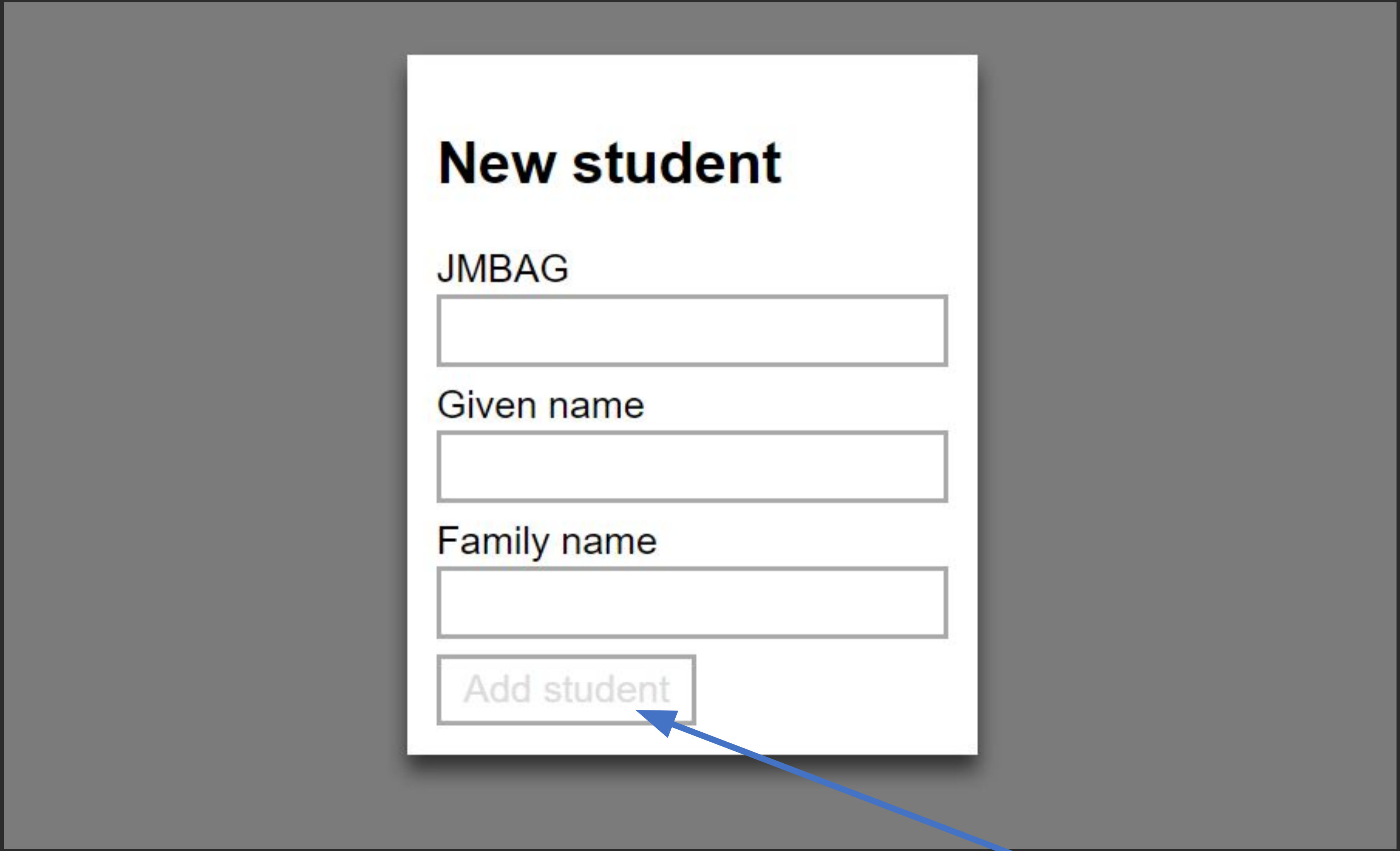
Request Headers (14)

Request Payload view source

```
▼{jmbag: "1111111111", familyName: "Tomić", givenName: "Josip"}
    familyName: "Tomić"
    givenName: "Josip"
    jmbag: "1111111111"
```

Dodavanje validacija

```
function isValid() {



}

return (
  <div className="StudentForm">
    <h2>New student</h2>
    <form onSubmit={onSubmit}>
      <div/>
      <div/>
      <div/>
      <button type="submit"                                        >Add student</button>
    </form>
  </div>
)
```



New student

JMBAG

Given name

Family name

Gumb je onemogućen

React router

V6.3

Zašto routing, ako imamo Single Page App?

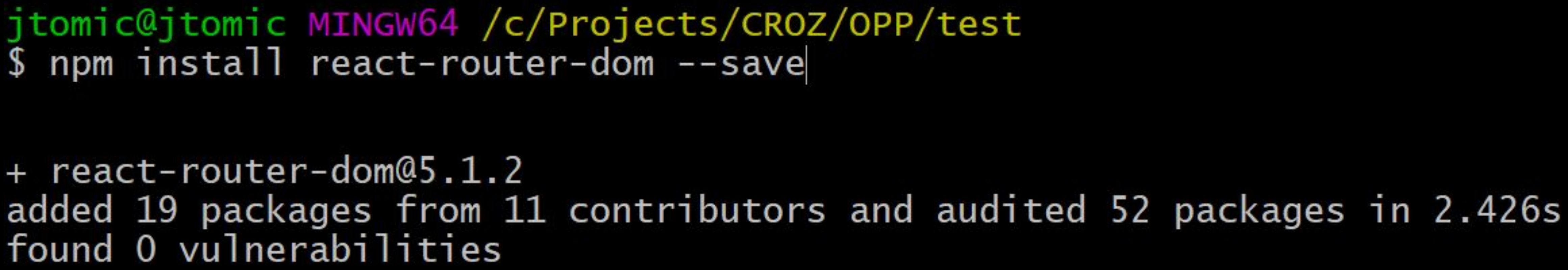
Korisnici su naviknuti vidjeti reprezentativni url stranice na kojoj se nalaze

Lakše dijeljenje sadržaja drugim korisnicima

Pomaže logičkom razdjeljivanju featura aplikacije

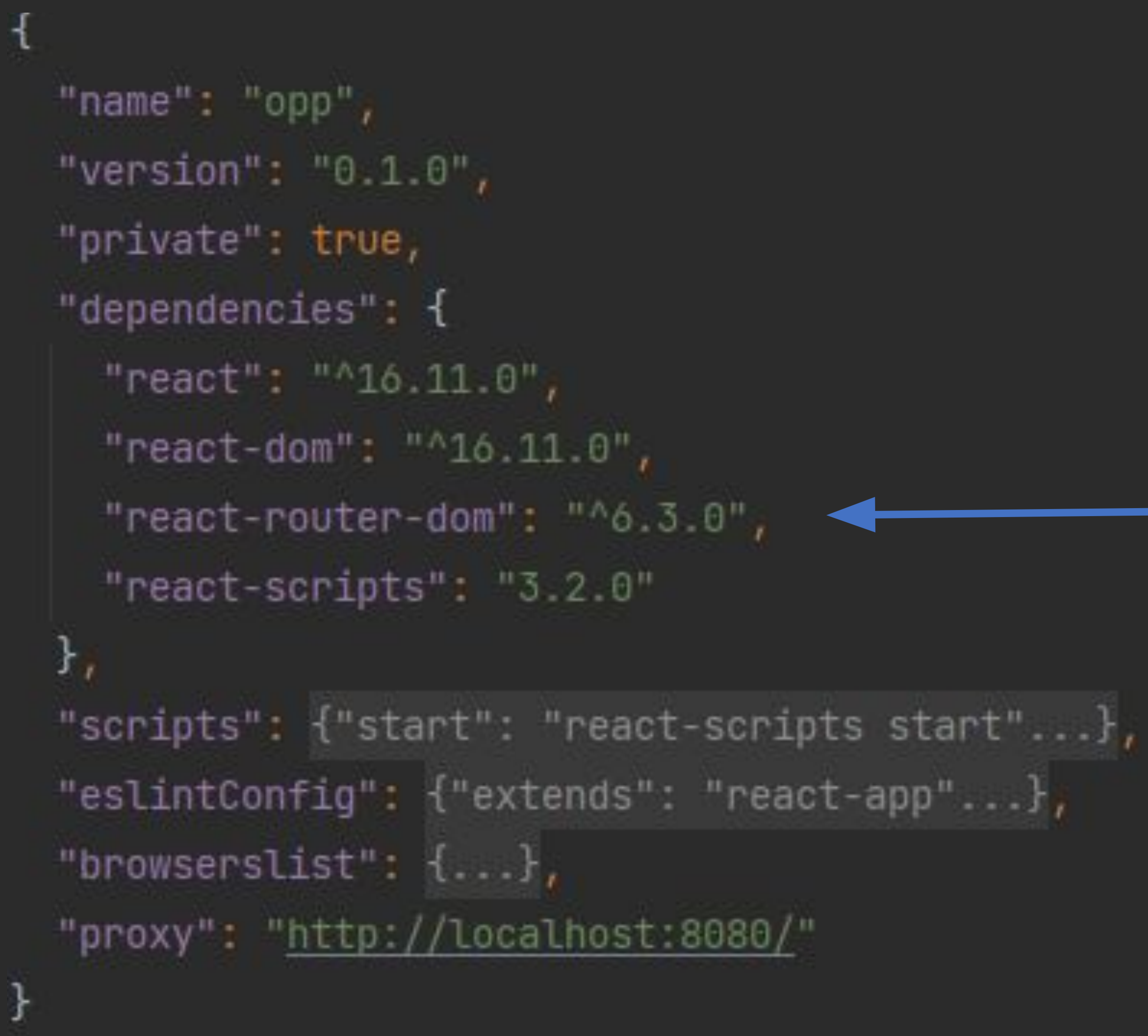
Kod postaje uredniji

Dodavanje dependencya



```
jtomic@jtomic MINGW64 /c/Projects/CROZ/OPP/test
$ npm install react-router-dom --save


+ react-router-dom@5.1.2
added 19 packages from 11 contributors and audited 52 packages in 2.426s
found 0 vulnerabilities
```

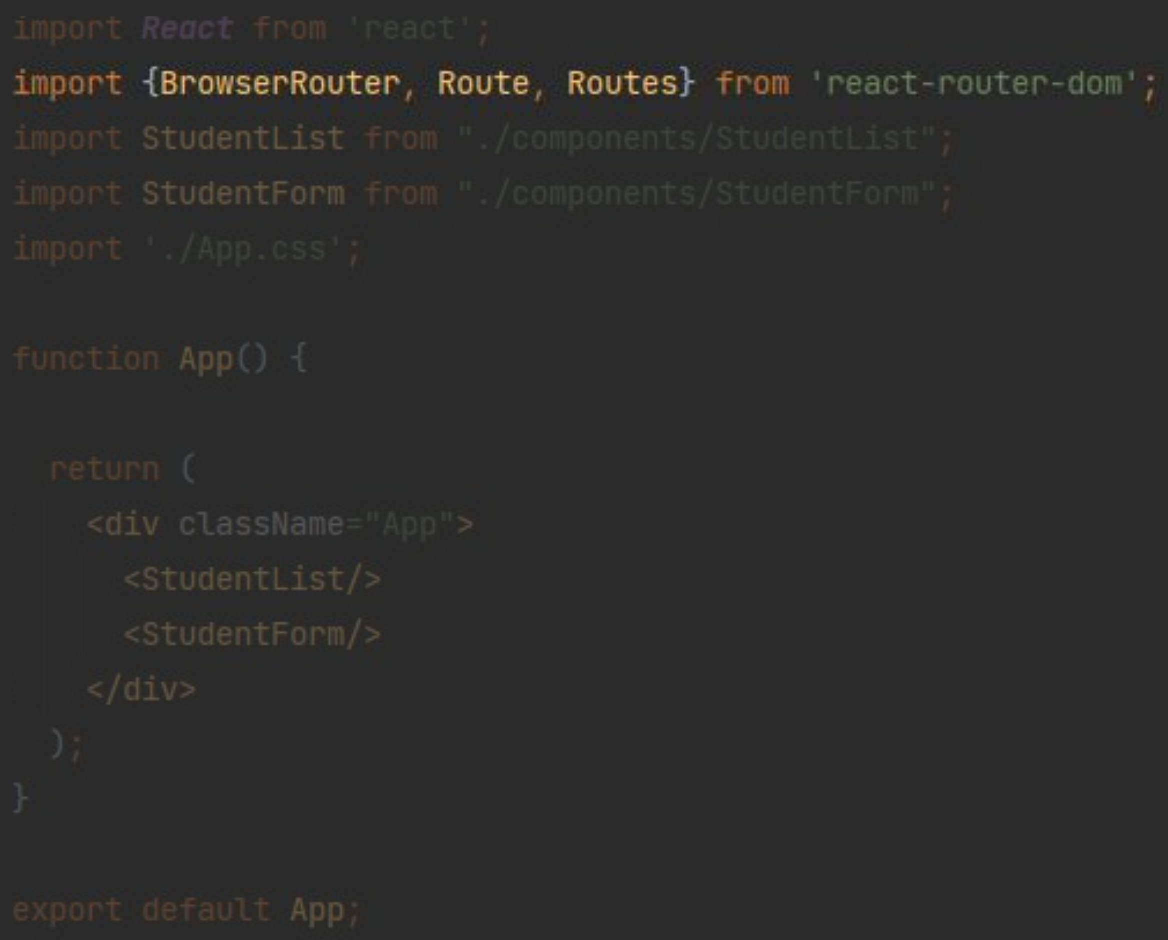
package.json



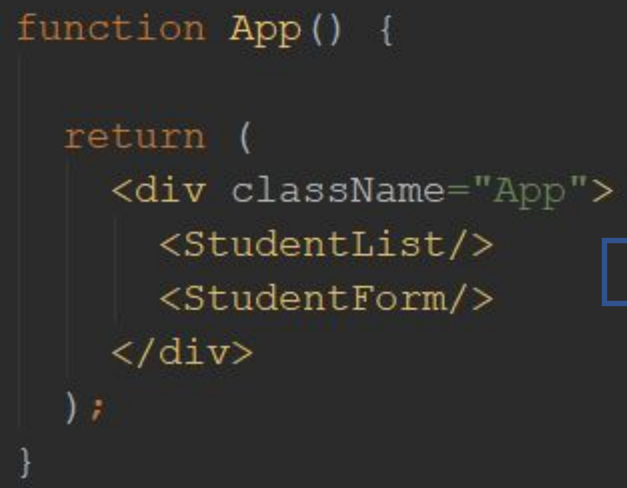
```
{
  "name": "opp",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "react": "^16.11.0",
    "react-dom": "^16.11.0",
    "react-router-dom": "^6.3.0",
    "react-scripts": "3.2.0"
  },
  "scripts": {"start": "react-scripts start"...},
  "eslintConfig": {"extends": "react-app"...},
  "browserslist": {...},
  "proxy": "http://localhost:8080/"
}
```

Dodan dependency

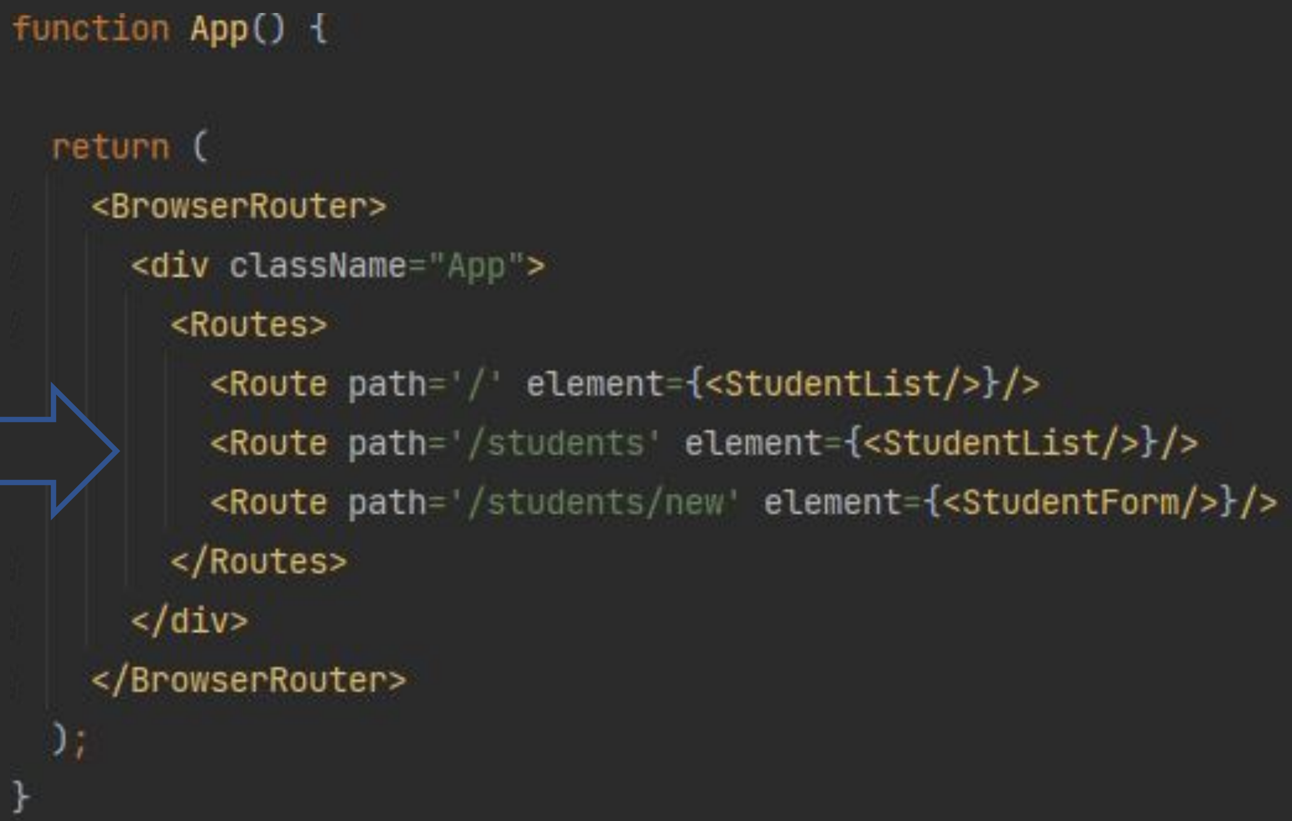
App.js



```
import React from 'react';
import {BrowserRouter, Route, Routes} from 'react-router-dom';
import StudentList from "./components/StudentList";
import StudentForm from "./components/StudentForm";
import './App.css';

function App() {

  return (
    <div className="App">
      <StudentList/>
      <StudentForm/>
    </div>
  );
}

export default App;
```



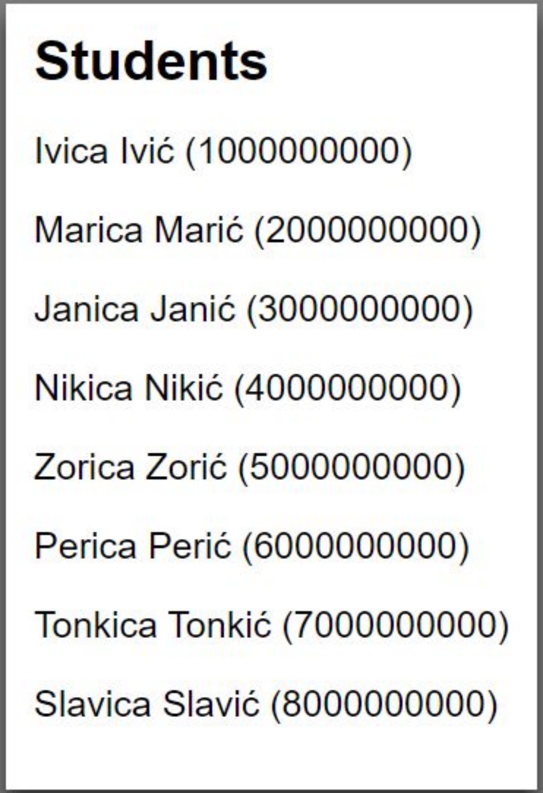
```
function App() {

  return (
    <div className="App">
      <StudentList/>
      <StudentForm/>
    </div>
  );
}
```



```
function App() {

  return (
    <BrowserRouter>
      <div className="App">
        <Routes>
          <Route path='/' element={<StudentList/>}/>
          <Route path='/students' element={<StudentList/>}/>
          <Route path='/students/new' element={<StudentForm/>}/>
        </Routes>
      </div>
    </BrowserRouter>
  );
}
```

http://localhost:3000/students

http://localhost:3000/students/new



Students

Ivica Ivić (1000000000)

Marica Marić (2000000000)

Janica Janić (3000000000)

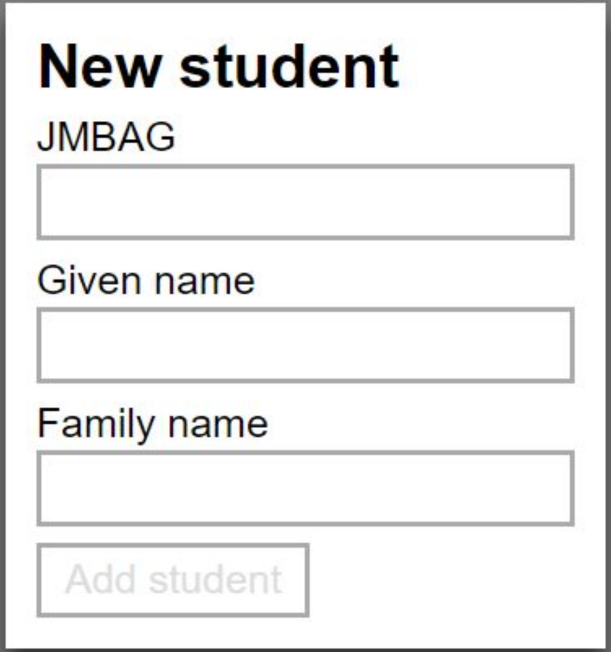
Nikica Nikić (4000000000)

Zorica Zorić (5000000000)

Perica Perić (6000000000)

Tonkica Tonkić (7000000000)

Slavica Slavić (8000000000)

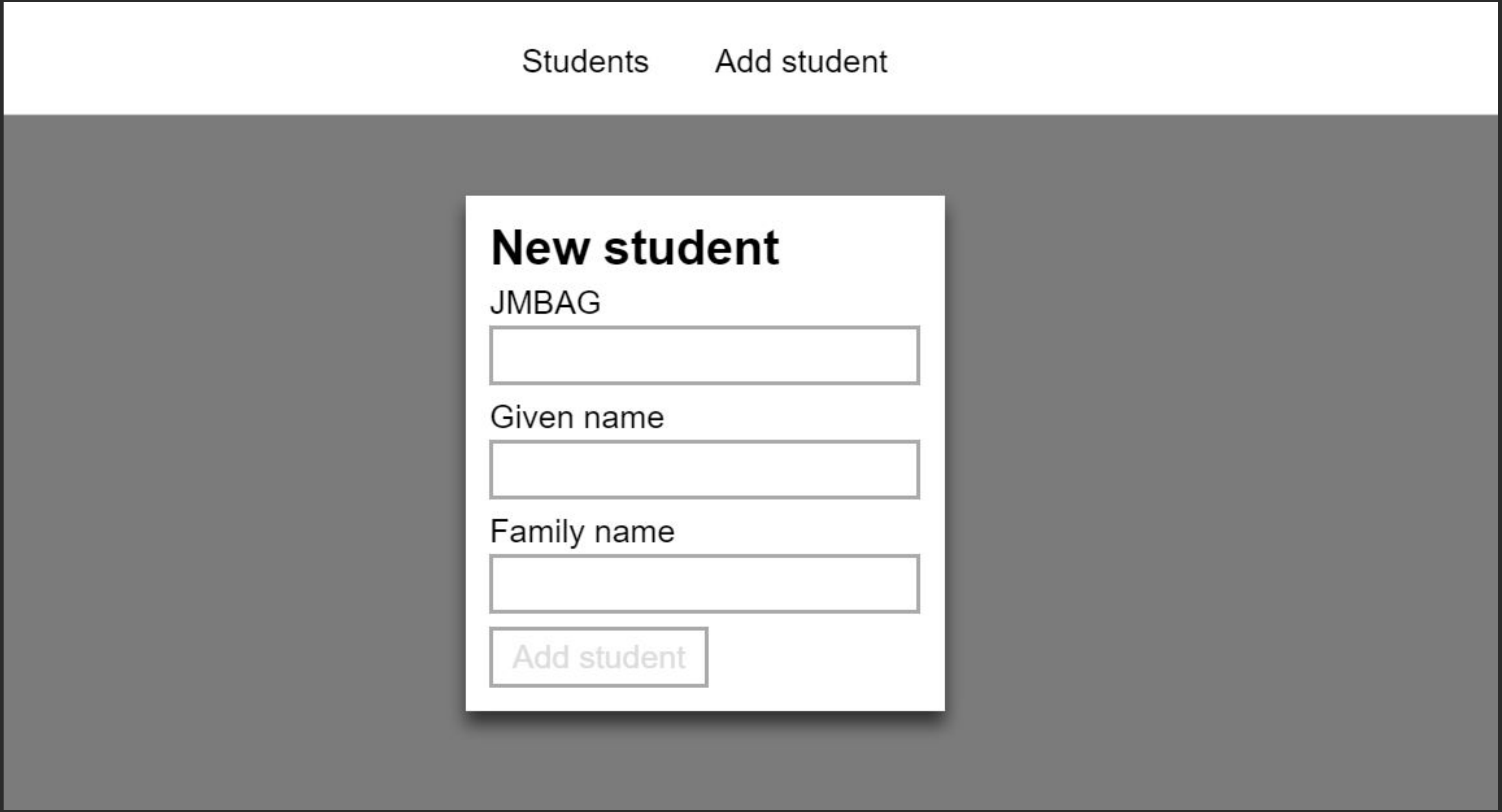


Linkovi

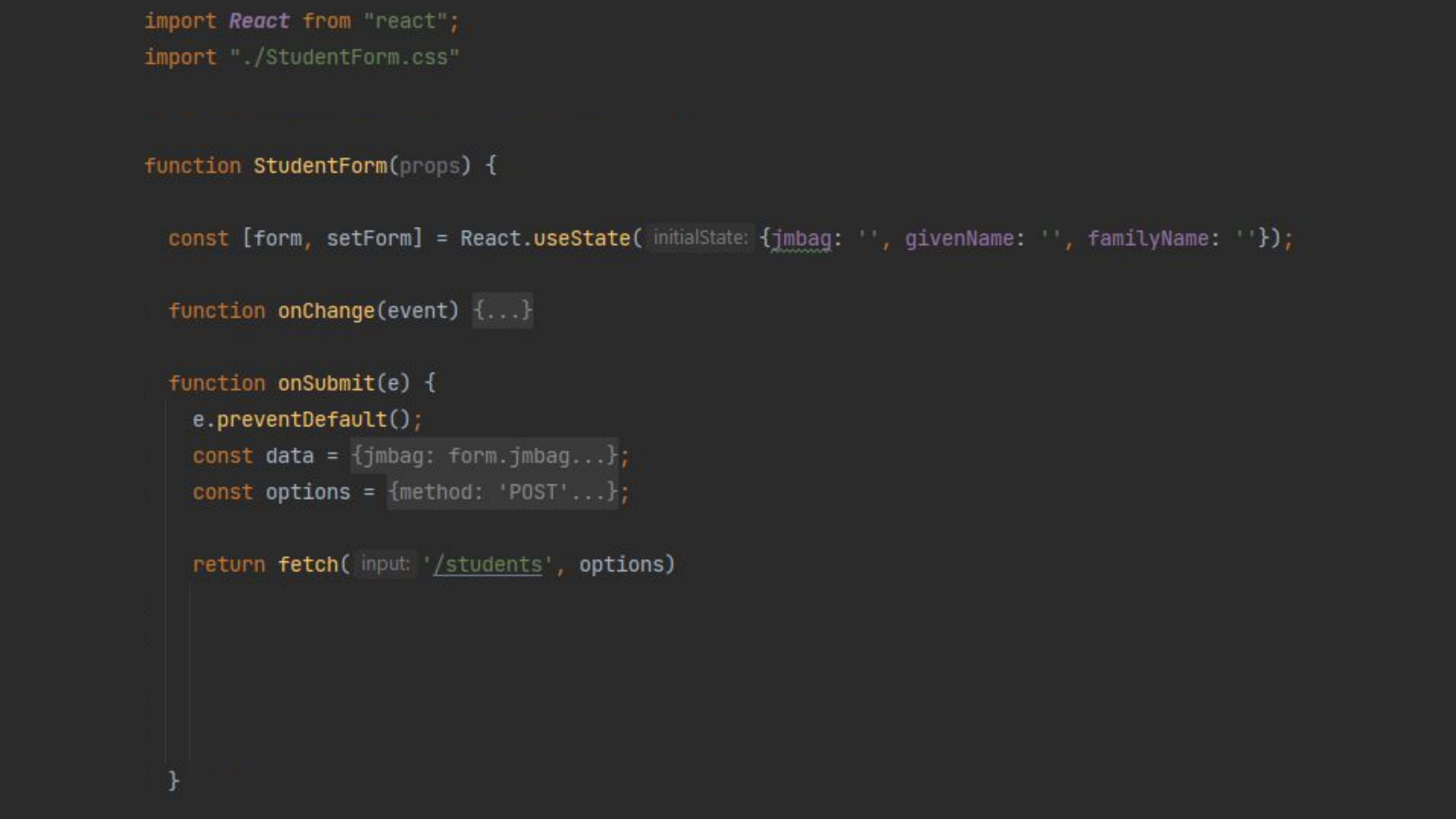
Header

Header.js

```
import React from 'react';
import {Link} from "react-router-dom";
import "./Header.css";

function Header() {
  return (
    <header className="Header">
      <Link to='/students'>Students</Link>
      <Link to='/students/new'>Add student</Link>
    </header>
  )
}

export default Header;
```


Students
Add student
New student
JMBAG
Given name
Family name
Add student

Programatska promjena rute



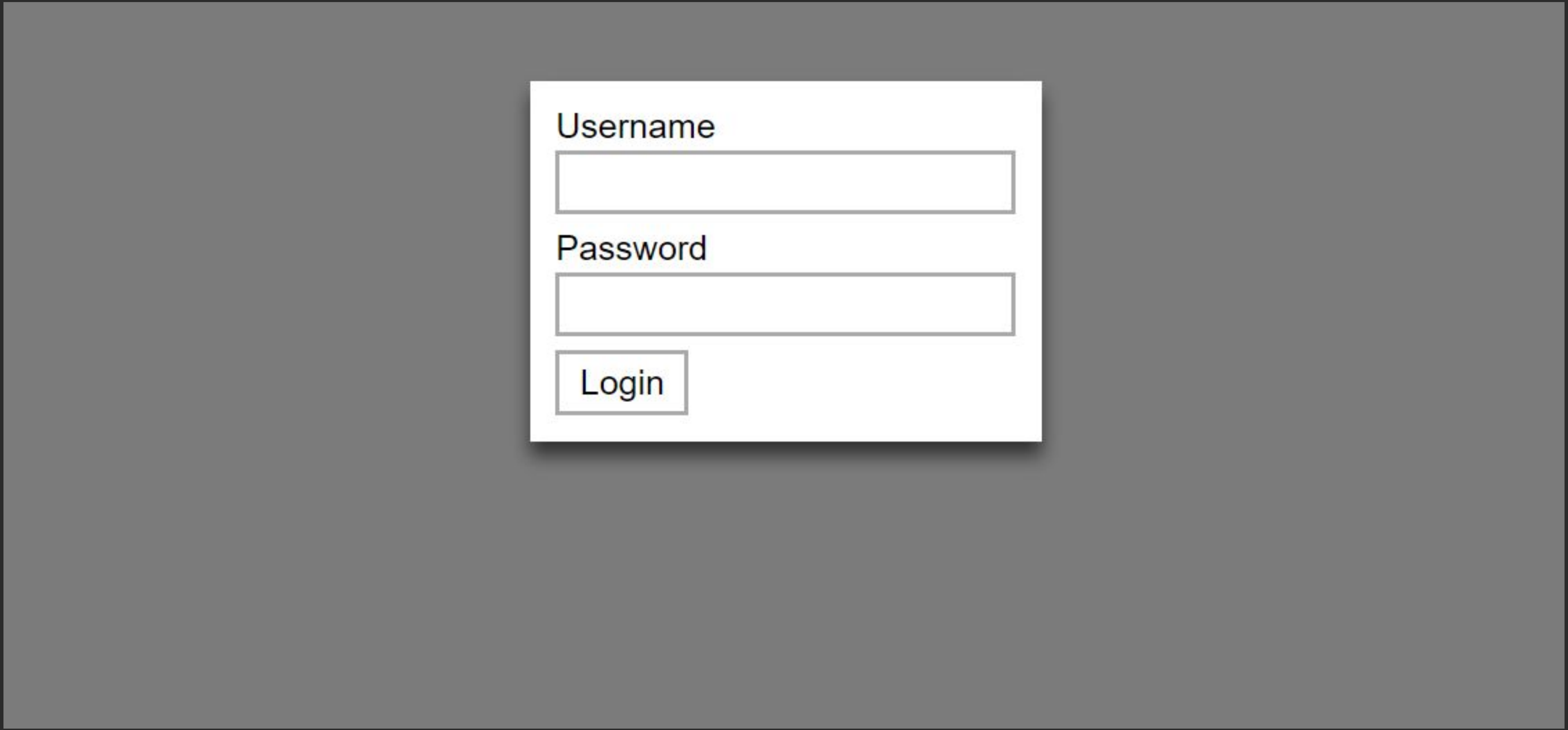
```
import React from "react";
import "./StudentForm.css"


function StudentForm(props) {

  const [form, setForm] = React.useState( initialState: {jmbag: '', givenName: '', familyName: ''});

  function onChange(event) {...}

  function onSubmit(e) {
    e.preventDefault();
    const data = {jmbag: form.jmbag...};
    const options = {method: 'POST'...};

    return fetch( input: '/students', options)



  }
```

Security

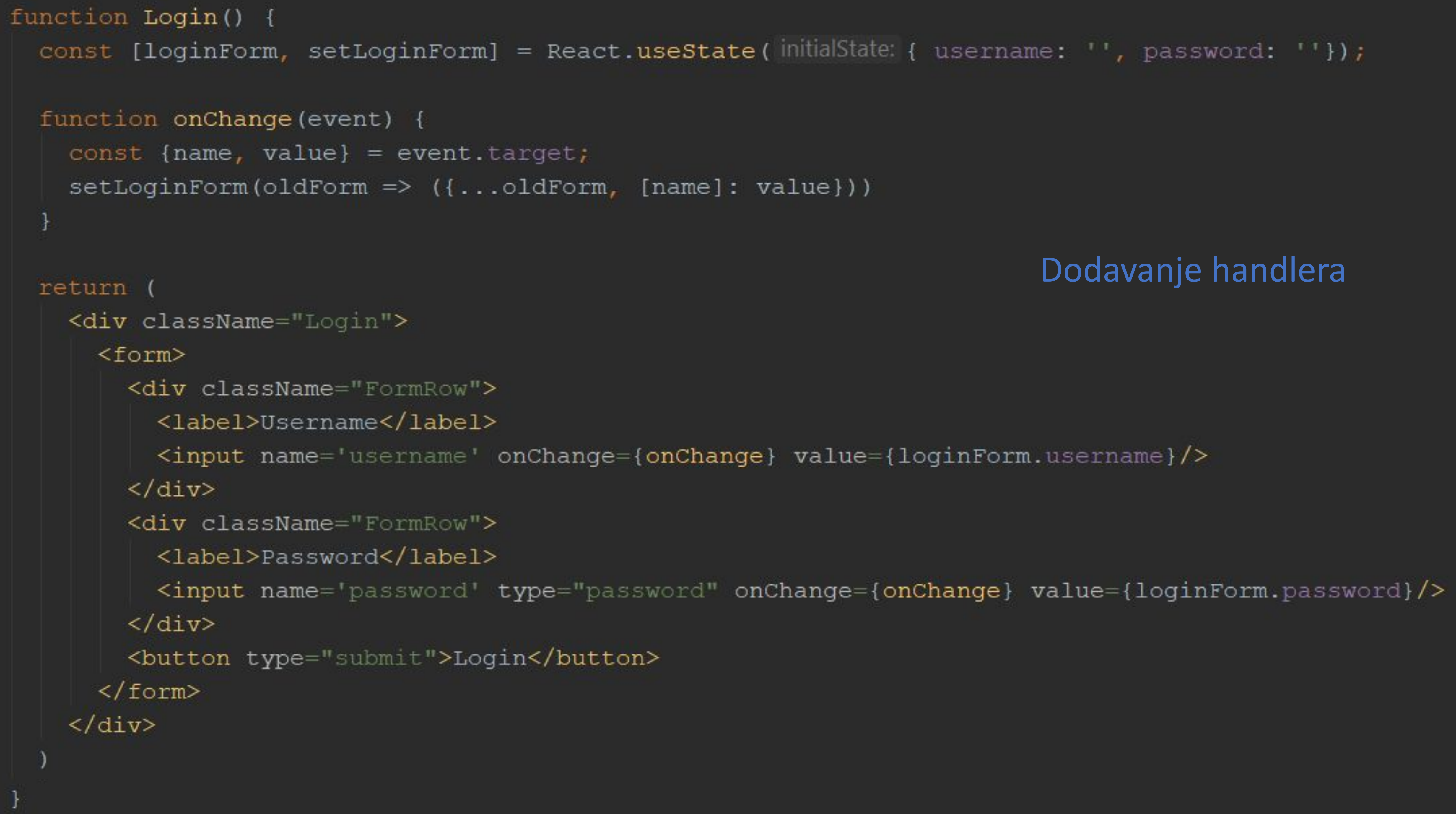
Login komponenta

Login.js

```
function Login() {

  return (
    <div className="Login">
      <form>
        <div className="FormRow">
          <label>Username</label>
          <input name='username'/>
        </div>
        <div className="FormRow">
          <label>Password</label>
          <input name='password' type="password"/>
        </div>
        <button type="submit">Login</button>
      </form>
    </div>
  )
}
```


Username
Password
Login

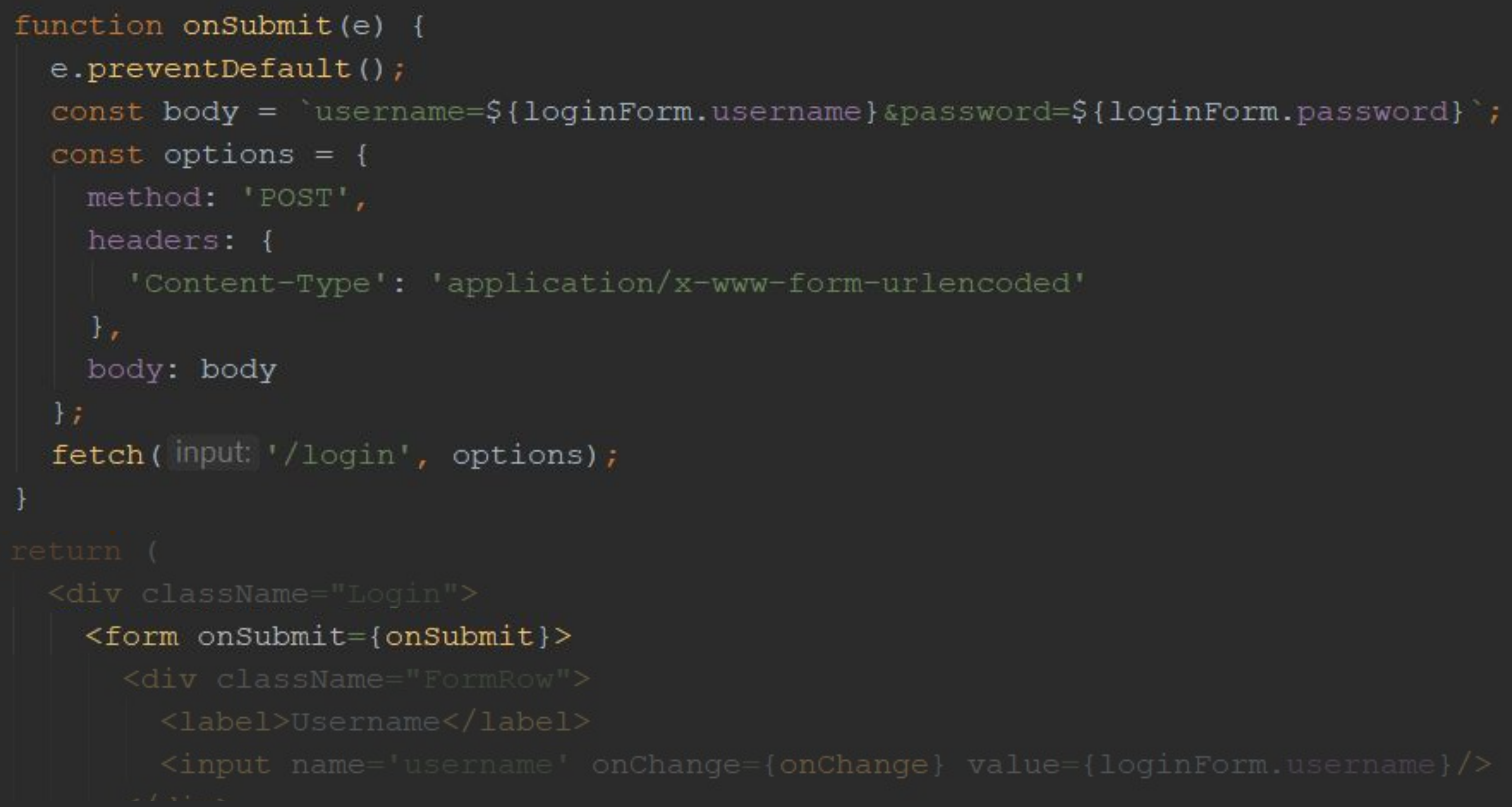
Dodavanje handlera



```
function Login() {
  const [loginForm, setLoginForm] = React.useState( initialState: { username: '', password: ''});

  function onChange(event) {
    const {name, value} = event.target;
    setLoginForm(oldForm => ({...oldForm, [name]: value}))
  }

  return (
    <div className="Login">
      <form>
        <div className="FormRow">
          <label>Username</label>
          <input name='username' onChange={onChange} value={loginForm.username}/>
        </div>
        <div className="FormRow">
          <label>Password</label>
          <input name='password' type="password" onChange={onChange} value={loginForm.password}/>
        </div>
        <button type="submit">Login</button>
      </form>
    </div>
  )
}
```

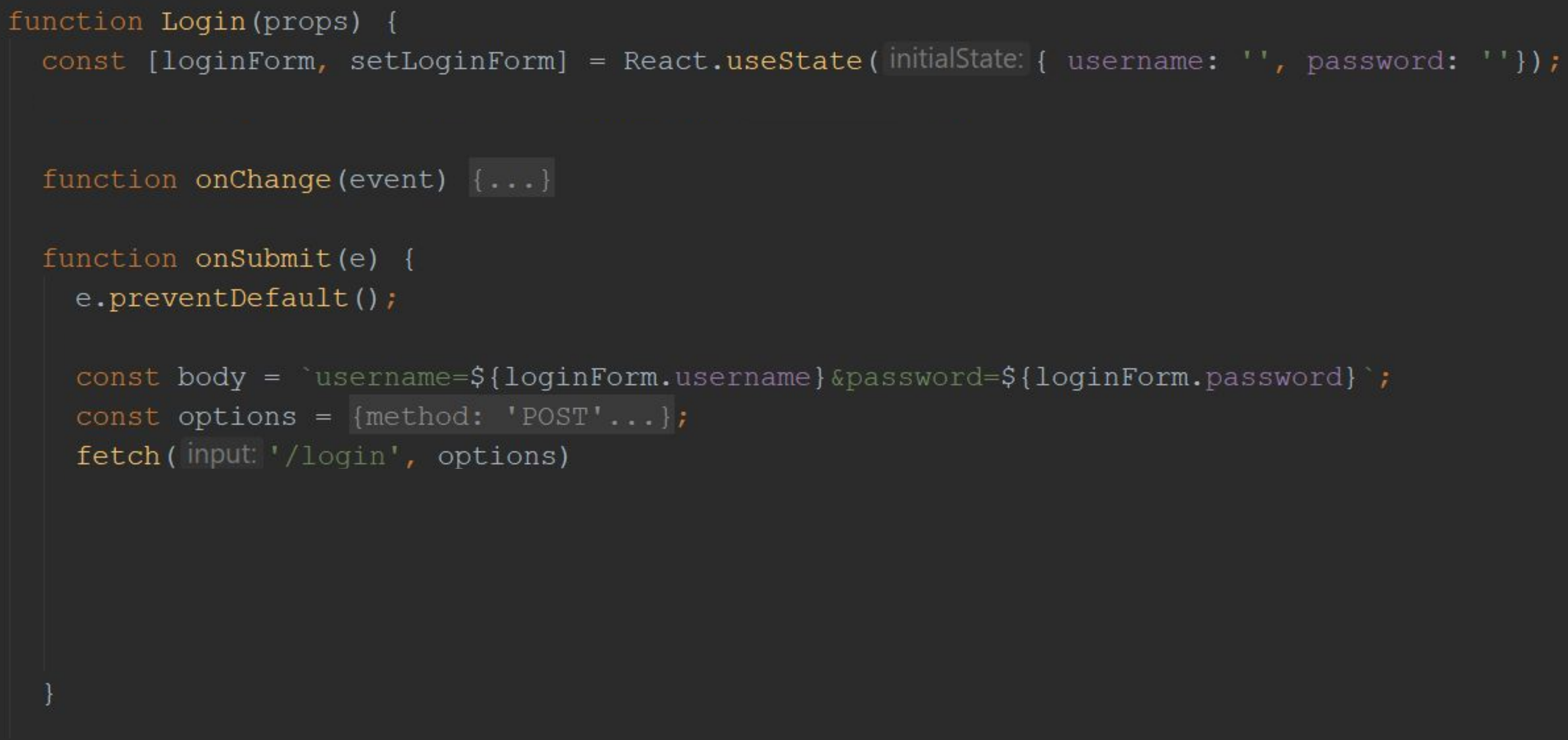
Dostupno na backendu

Putanja:	/login
Metoda:	POST
Body:	Username i password
Content-Type:	x-www-form-urlencoded
Vraća:	200 ili 401

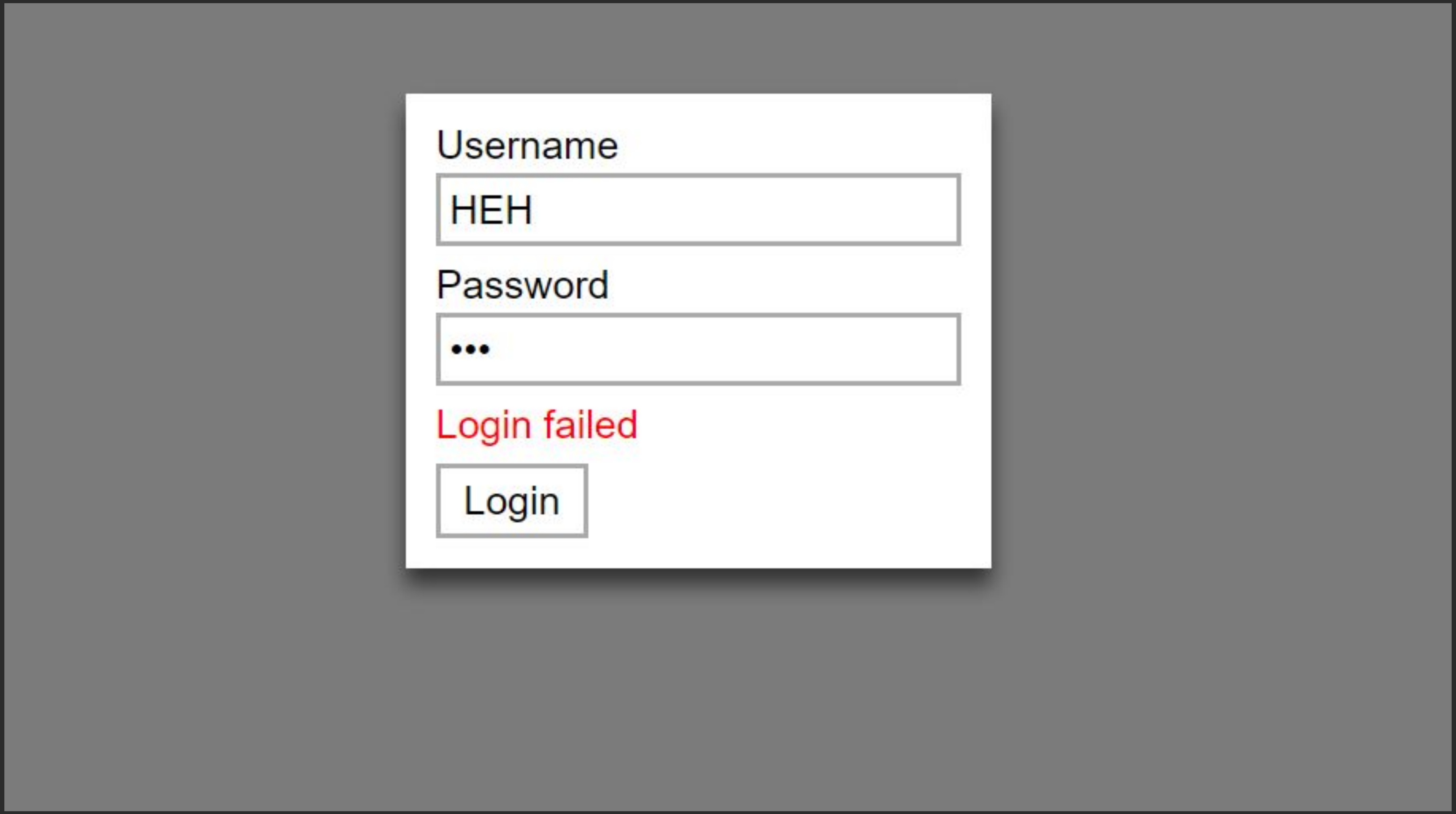


```
function onSubmit(e) {
  e.preventDefault();
  const body = `username=${loginForm.username}&password=${loginForm.password}`;
  const options = {
    method: 'POST',
    headers: {
      'Content-Type': 'application/x-www-form-urlencoded'
    },
    body: body
  };
  fetch( input: '/login', options);
}
return (
  <div className="Login">
    <form onSubmit={onSubmit}>
      <div className="FormRow">
        <label>Username</label>
        <input name='username' onChange={onChange} value={loginForm.username}/>
```

Error handling



```
function Login(props) {
  const [loginForm, setLoginForm] = React.useState( initialState: { username: '', password: ''});


  function onChange(event) {...}

  function onSubmit(e) {
    e.preventDefault();

    const body = `username=${loginForm.username}&password=${loginForm.password}`;
    const options = {method: 'POST'...};
    fetch( input: '/login', options)




  }
```


Username

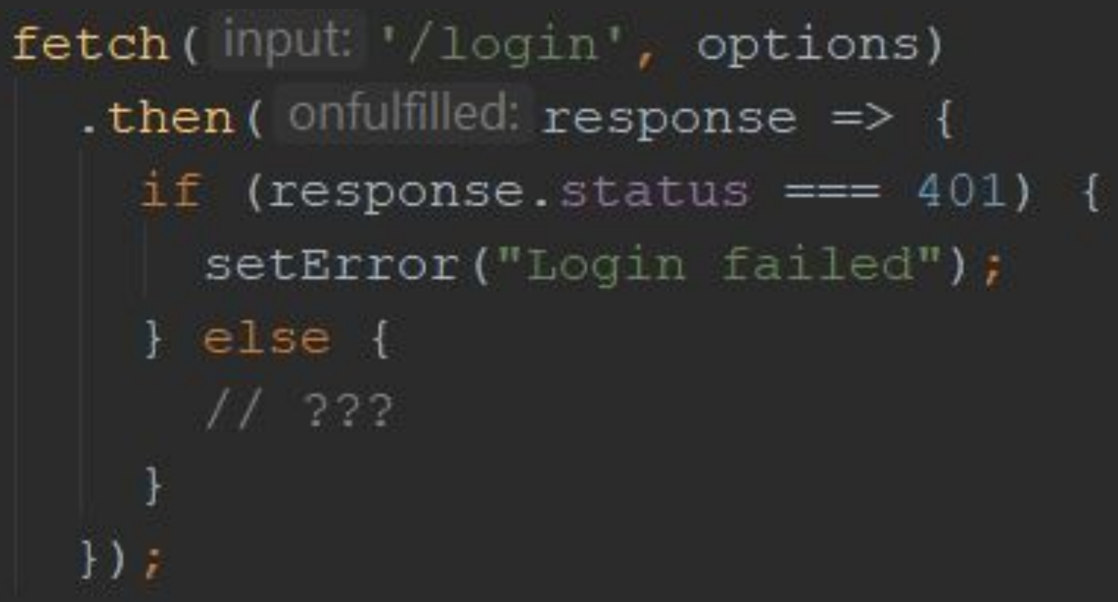
HEH

Password

•••

Login failed

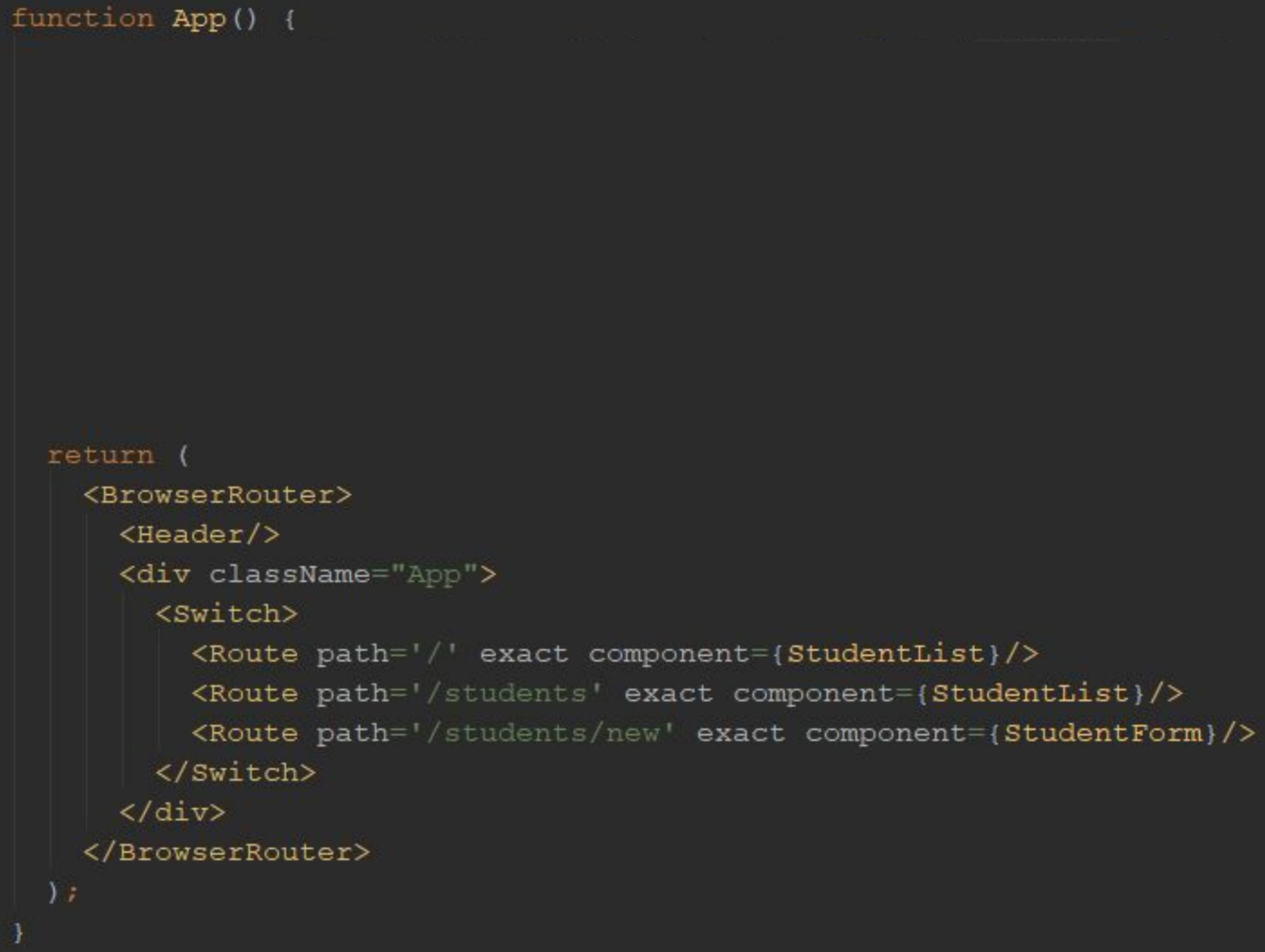
Što ćemo odraditi ako nema greške i login prođe dobro?



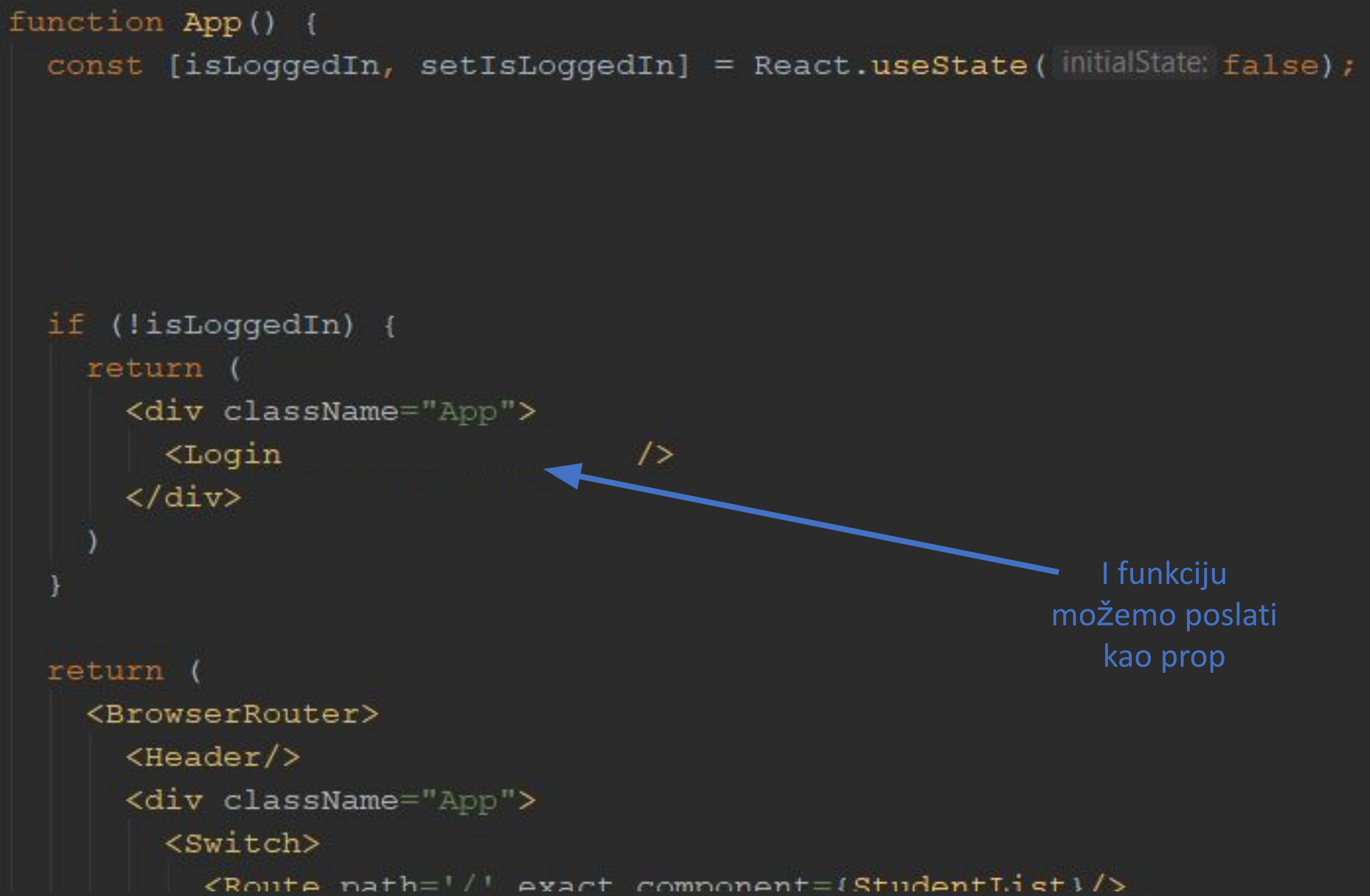
```
fetch( input: '/login', options)
  .then( onfulfilled: response => {
    if (response.status === 401) {
      setError("Login failed");
    } else {
      // ???
    }
  });
```

Pogledajmo gdje ćemo i kako koristiti Login komponentu

App.js



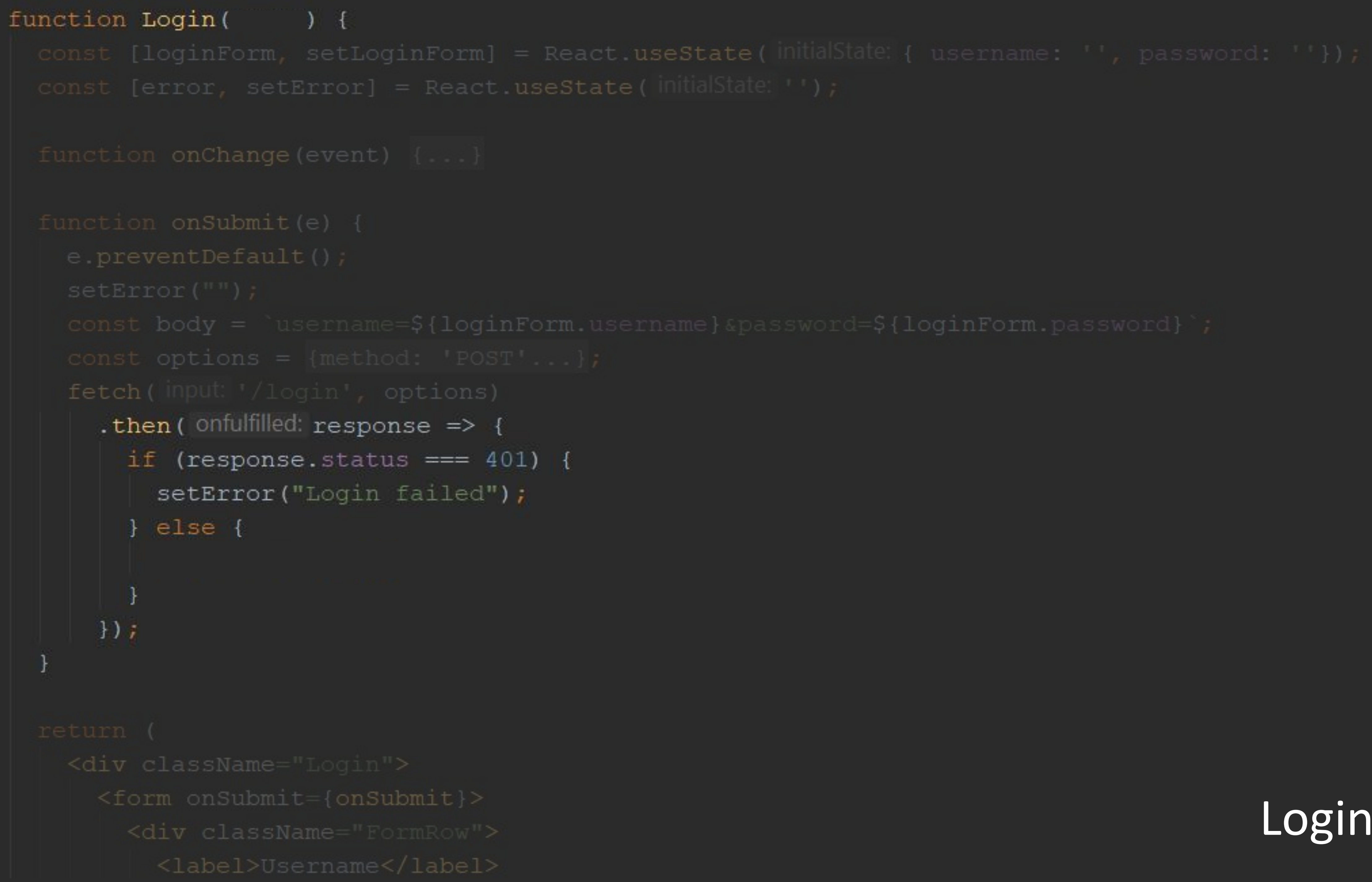
```
function App() {










  return (
    <BrowserRouter>
      <Header/>
      <div className="App">
        <Switch>
          <Route path='/' exact component={StudentList}/>
          <Route path='/students' exact component={StudentList}/>
          <Route path='/students/new' exact component={StudentForm}/>
        </Switch>
      </div>
    </BrowserRouter>
  );
}
```



```
function App() {
  const [isLoggedIn, setIsLoggedIn] = React.useState( initialState: false);




  if (!isLoggedIn) {
    return (
      <div className="App">
        <Login              />
      </div>
    )
  }

  return (
    <BrowserRouter>
      <Header/>
      <div className="App">
        <Switch>
          <Route path='/' exact component={StudentList}/>
```

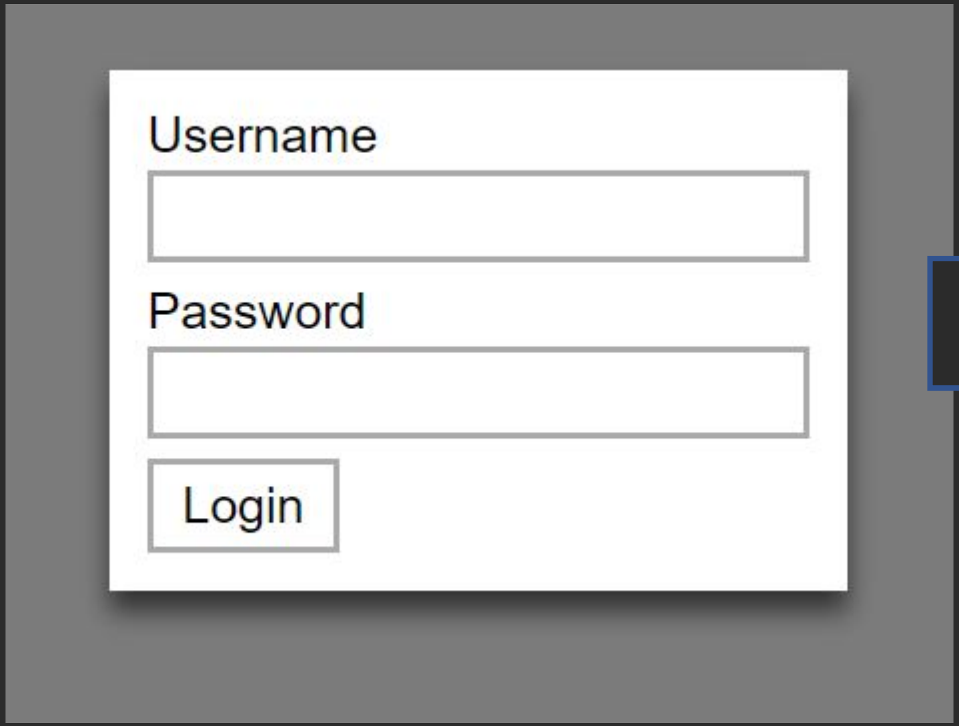
App.js

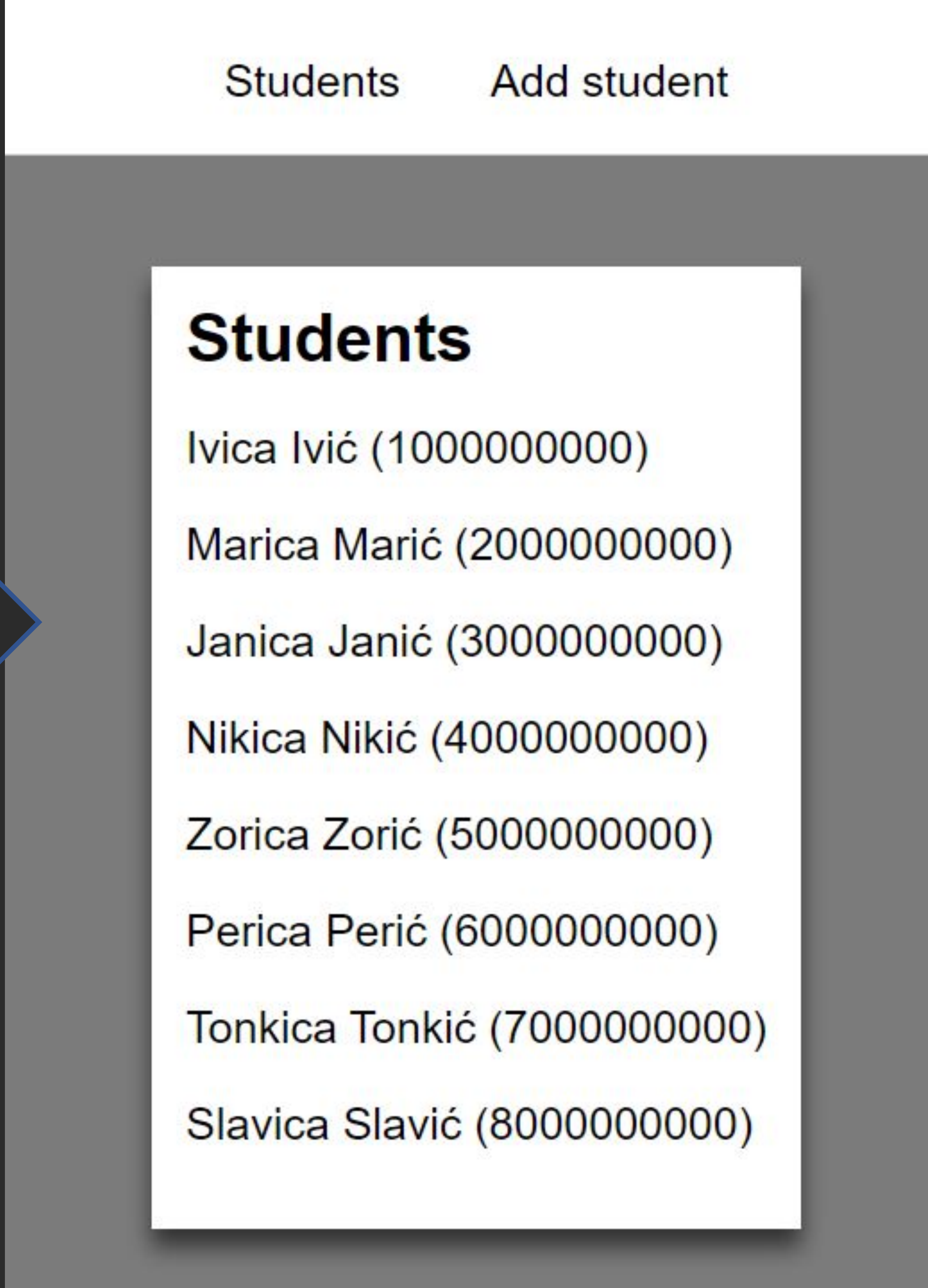
I funkciju
možemo poslati
kao prop



```
function Login(      ) {
  const [loginForm, setLoginForm] = React.useState( initialState: { username: '', password: ''});
  const [error, setError] = React.useState( initialState: '');

  function onChange(event) {...}

  function onSubmit(e) {
    e.preventDefault();
    setError("");
    const body = `username=${loginForm.username}&password=${loginForm.password}`;
    const options = {method: 'POST'...};
    fetch( input: '/login', options)
      .then( onfulfilled: response => {
        if (response.status === 401) {
          setError("Login failed");
        } else {

        }
      });
  }

  return (
    <div className="Login">
      <form onSubmit={onSubmit}>
        <div className="FormRow">
          <label>Username</label>
```

Login.js





Students

Ivica Ivić (1000000000)

Marica Marić (2000000000)

Janica Janić (3000000000)

Nikica Nikić (4000000000)

Zorica Zorić (5000000000)

Perica Perić (6000000000)

Tonkica Tonkić (7000000000)

Slavica Slavić (8000000000)

Dostupno na backendu

Putanja:	/logout
Metoda:	GET
Vraća:	200 OK

Header.js

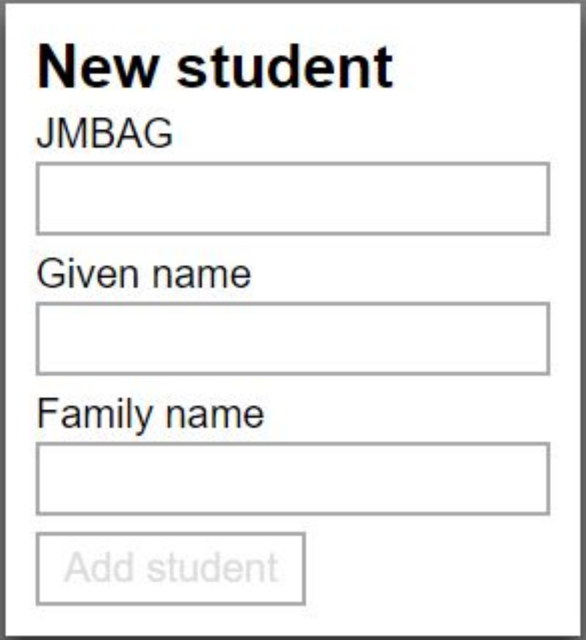
```
import React from 'react';
import {Link} from "react-router-dom";
import "./Header.css";

function Header(      ) {








  return (
    <header className="Header">
      <Link to='/students'>Students</Link>
      <Link to='/students/new'>Add student</Link>

    </header>
  )
}

export default Header;
```

And that’s all folks!

Ali samo što se tiče aplikacije

Izvlačenje duplicirane stilizacije

Vizualno podjednake komponente



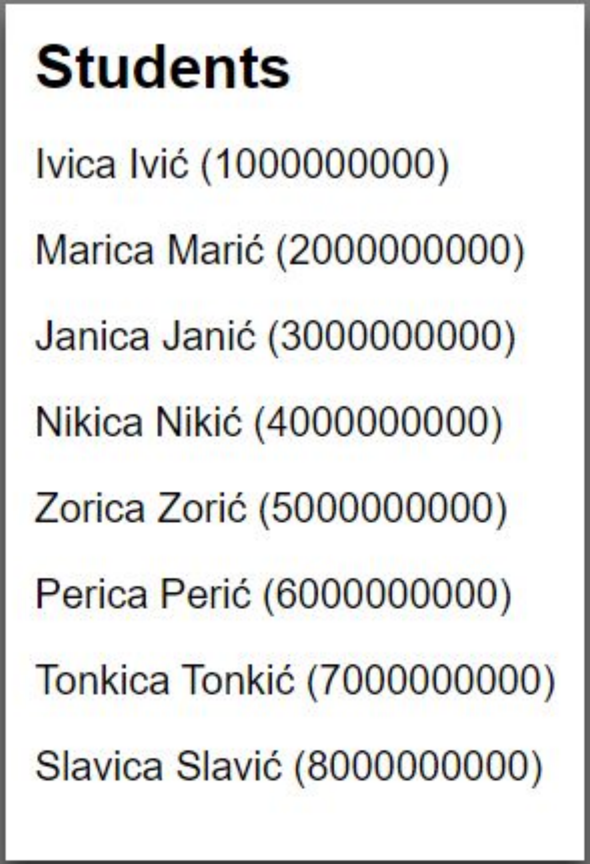
New student

JMBAG

Given name

Family name

Add student



Students

Ivica Ivić (1000000000)

Marica Marić (2000000000)

Janica Janić (3000000000)

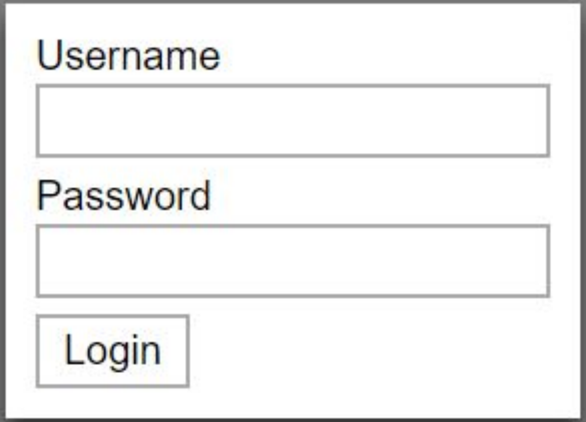
Nikica Nikić (4000000000)

Zorica Zorić (5000000000)

Perica Perić (6000000000)

Tonkica Tonkić (7000000000)

Slavica Slavić (8000000000)

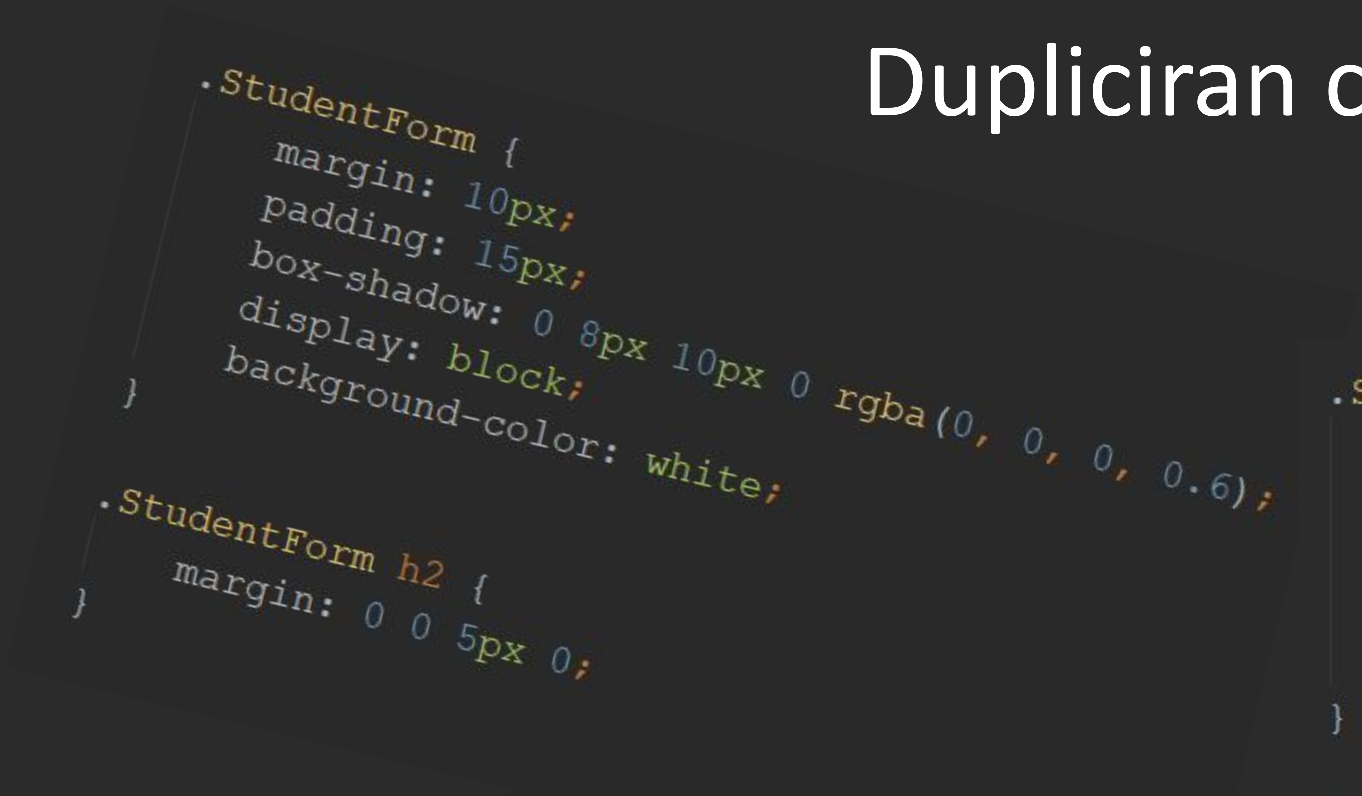


Username

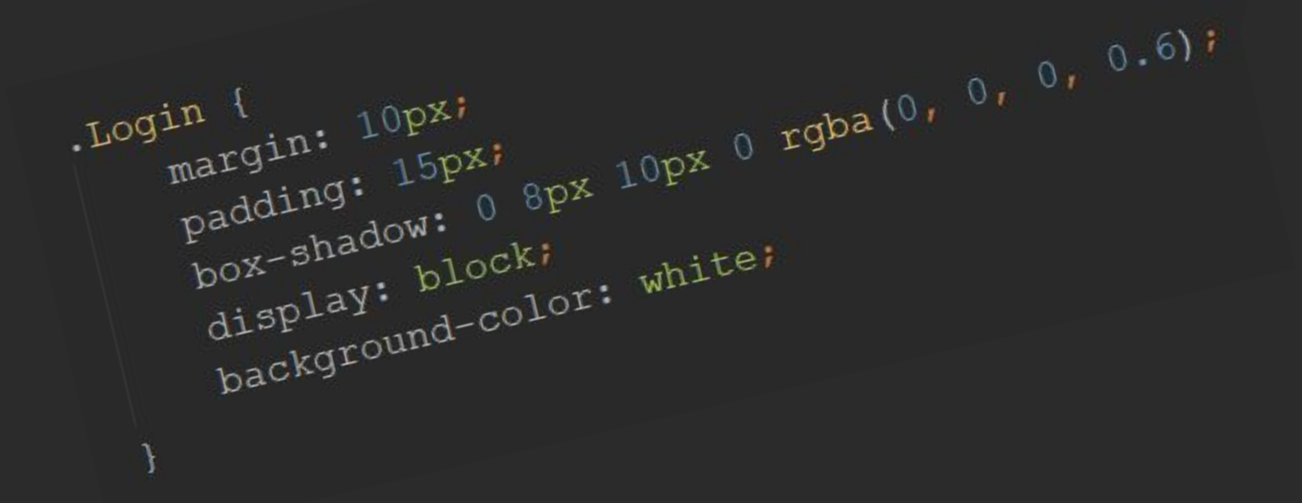
Password

Login

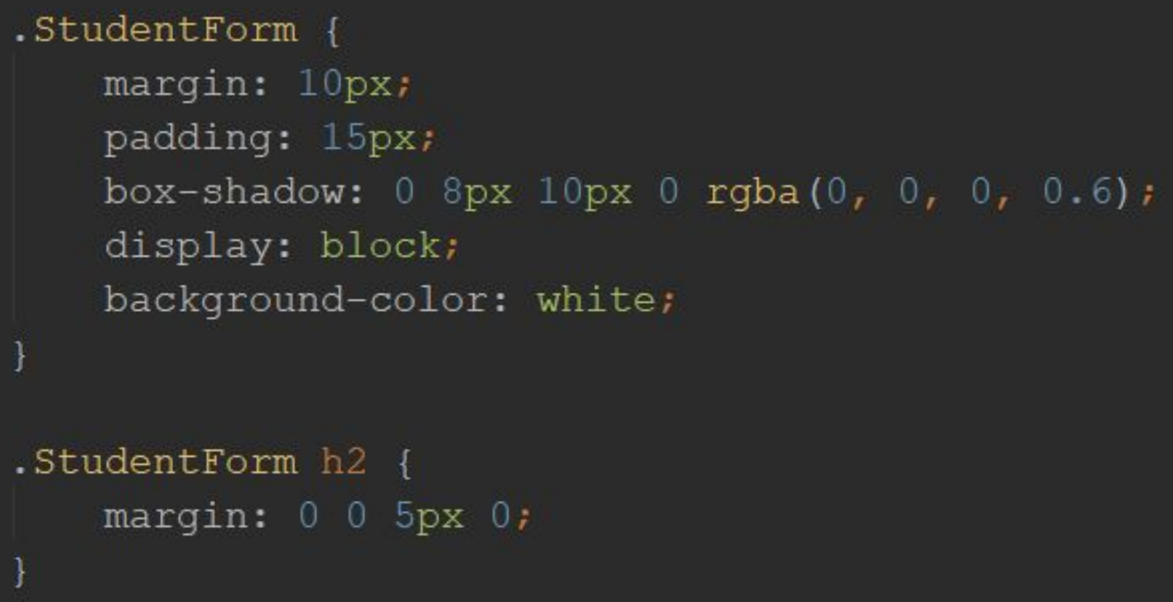
Dupliciran css???



```css
.StudentForm {
    margin: 10px;
    padding: 15px;
    box-shadow: 0 8px 10px 0 rgba(0, 0, 0, 0.6);
    display: block;
    background-color: white;
}

.StudentForm h2 {
    margin: 0 0 5px 0;
}
```

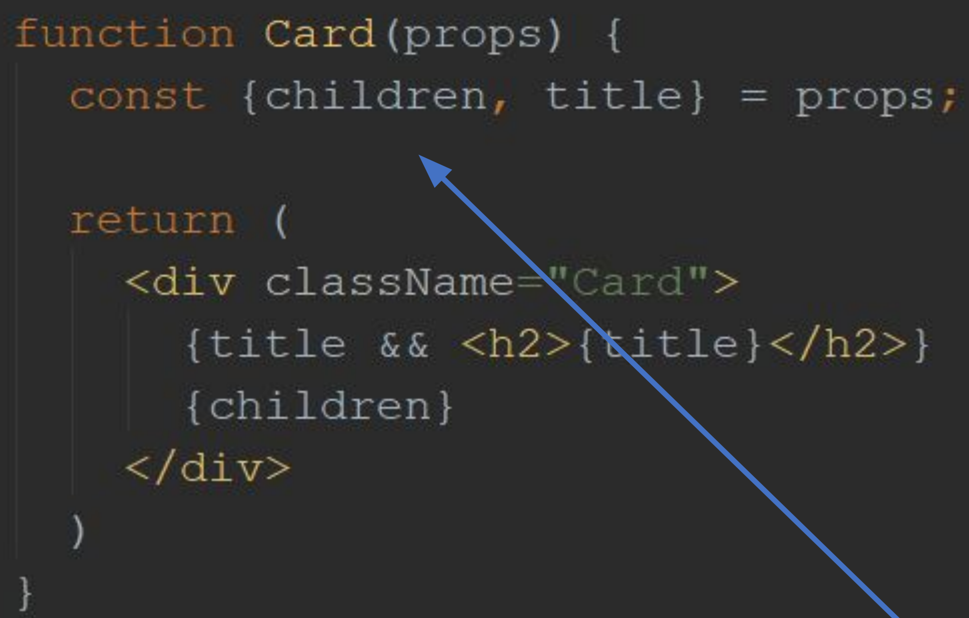


```css
.Login {
    margin: 10px;
    padding: 15px;
    box-shadow: 0 8px 10px 0 rgba(0, 0, 0, 0.6);
    display: block;
    background-color: white;

}
```



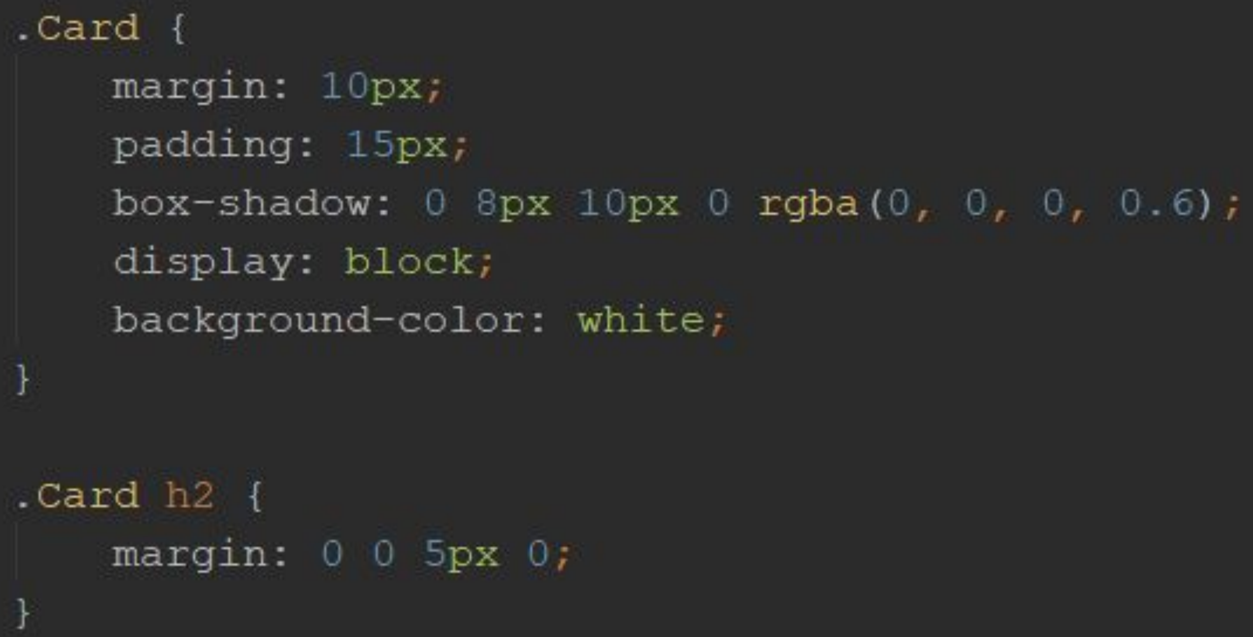
```css
.StudentForm {
    margin: 10px;
    padding: 15px;
    box-shadow: 0 8px 10px 0 rgba(0, 0, 0, 0.6);
    display: block;
    background-color: white;
}

.StudentForm h2 {
    margin: 0 0 5px 0;
}
```

Card.js



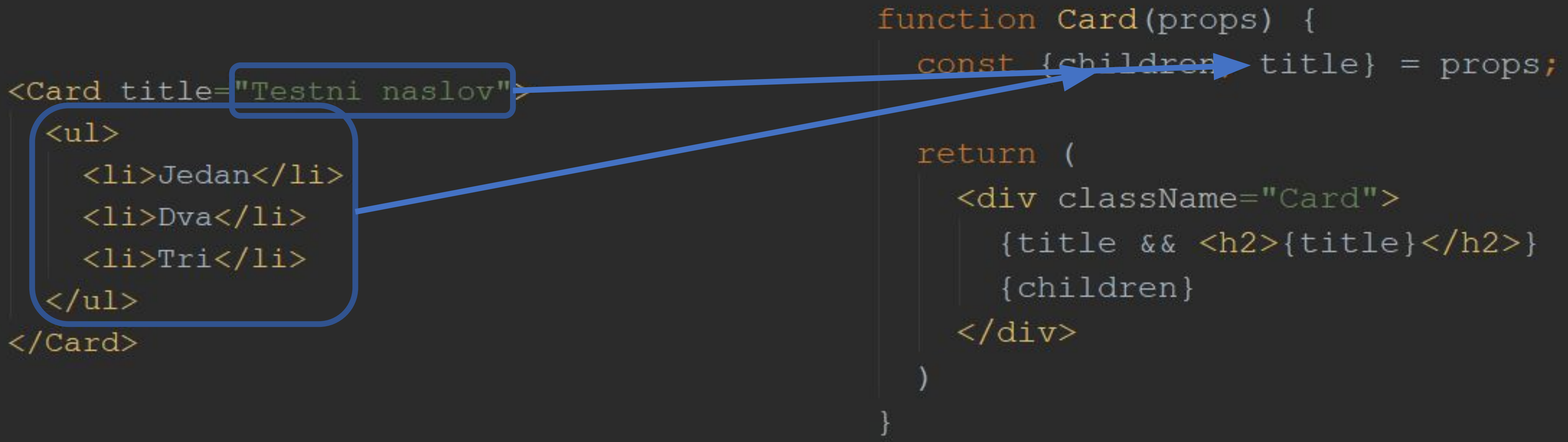
```
function Card(props) {
  const {children, title} = props;

  return (
    <div className="Card">
      {title && <h2>{title}</h2>}
      {children}
    </div>
  )
}
```

Card.css



```
.Card {
    margin: 10px;
    padding: 15px;
    box-shadow: 0 8px 10px 0 rgba(0, 0, 0, 0.6);
    display: block;
    background-color: white;
}

.Card h2 {
    margin: 0 0 5px 0;
}
```

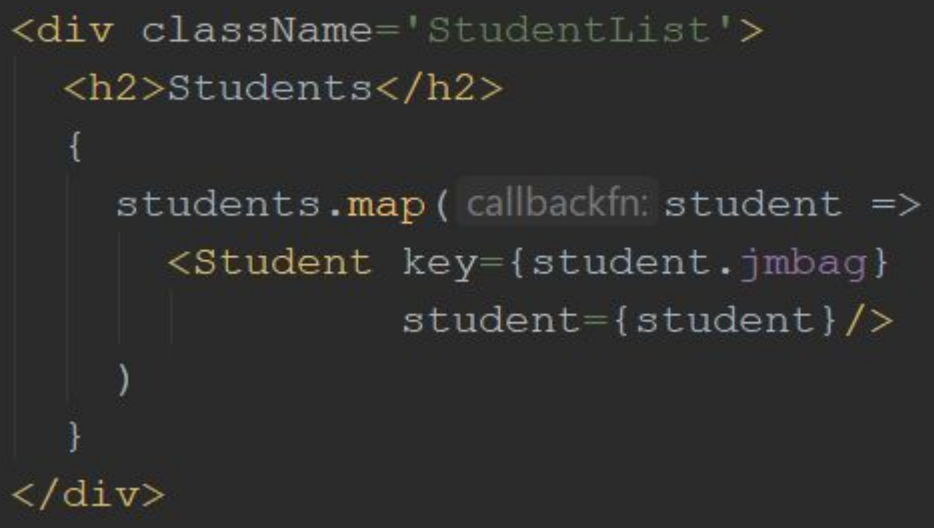
Poseban prop koji predstavlja podelemente u DOM-u



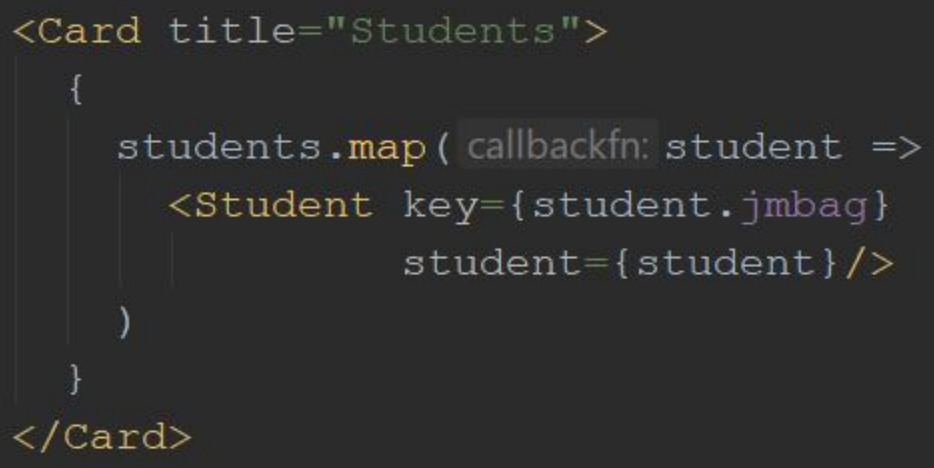
```
<Card title="Testni naslov">
  <ul>
    <li>Jedan</li>
    <li>Dva</li>
    <li>Tri</li>
  </ul>
</Card>
```

```
function Card(props) {
  const {children, title} = props;

  return (
    <div className="Card">
      {title && <h2>{title}</h2>}
      {children}
    </div>
  )
}
```

StudentList.js



```
<div className='StudentList'>
  <h2>Students</h2>
  {
    students.map( callbackfn: student =>
      <Student key={student.jmbag}
               student={student}/>
    )
  }
</div>
```



```
<Card title="Students">
  {
    students.map( callbackfn: student =>
      <Student key={student.jmbag}
               student={student}/>
    )
  }
</Card>
```

Dodatni materijali

Gotove komponente:

Material-UI
Reactstrap
Semantic UI

Upravljanje globalnim stanjem:

Redux
MobX

Sve za vrijeme:

Moment.js

Pregled u izolaciji:

Storybook

Mobilni razvoj:

React Native