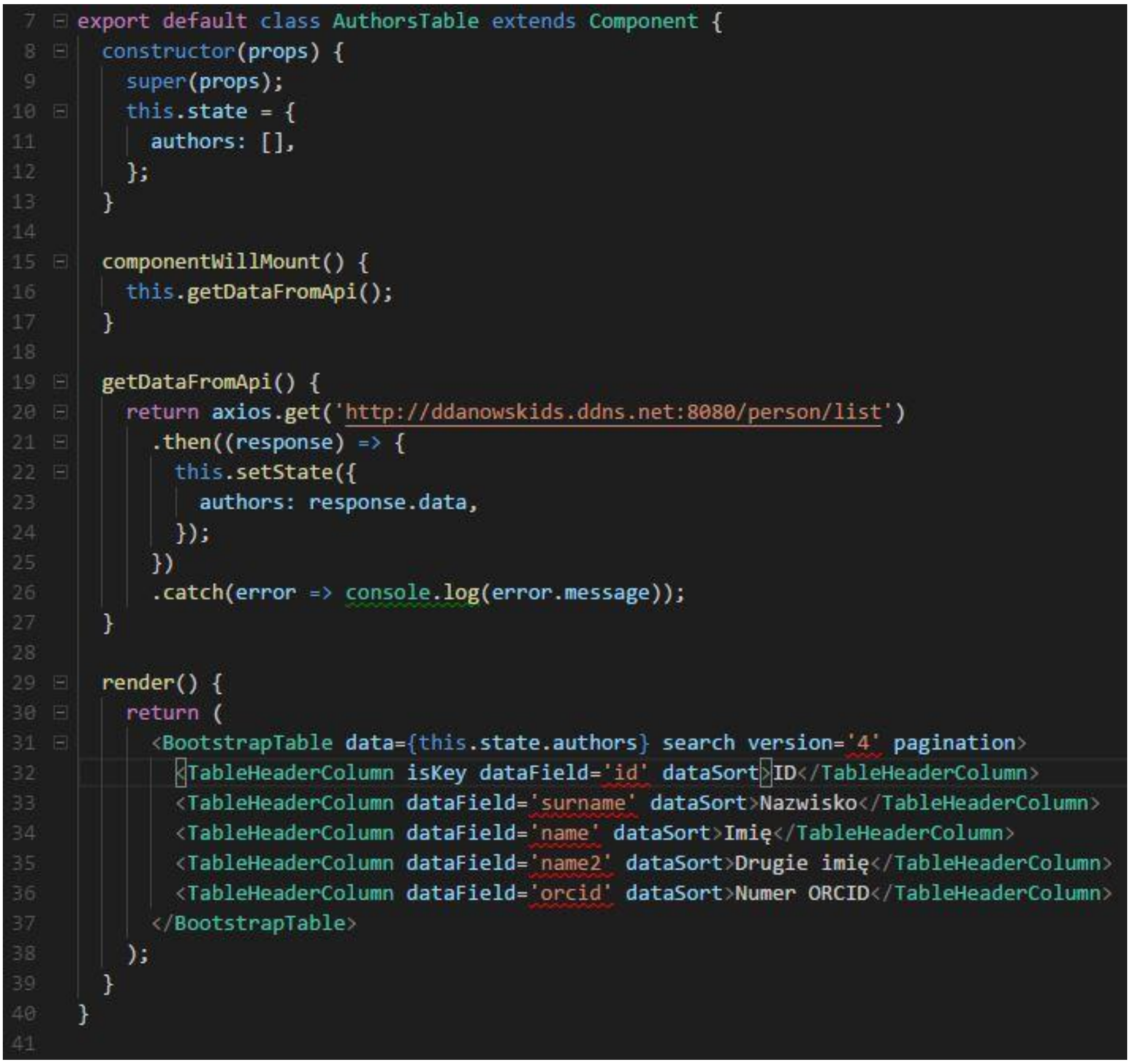
FRONTEND

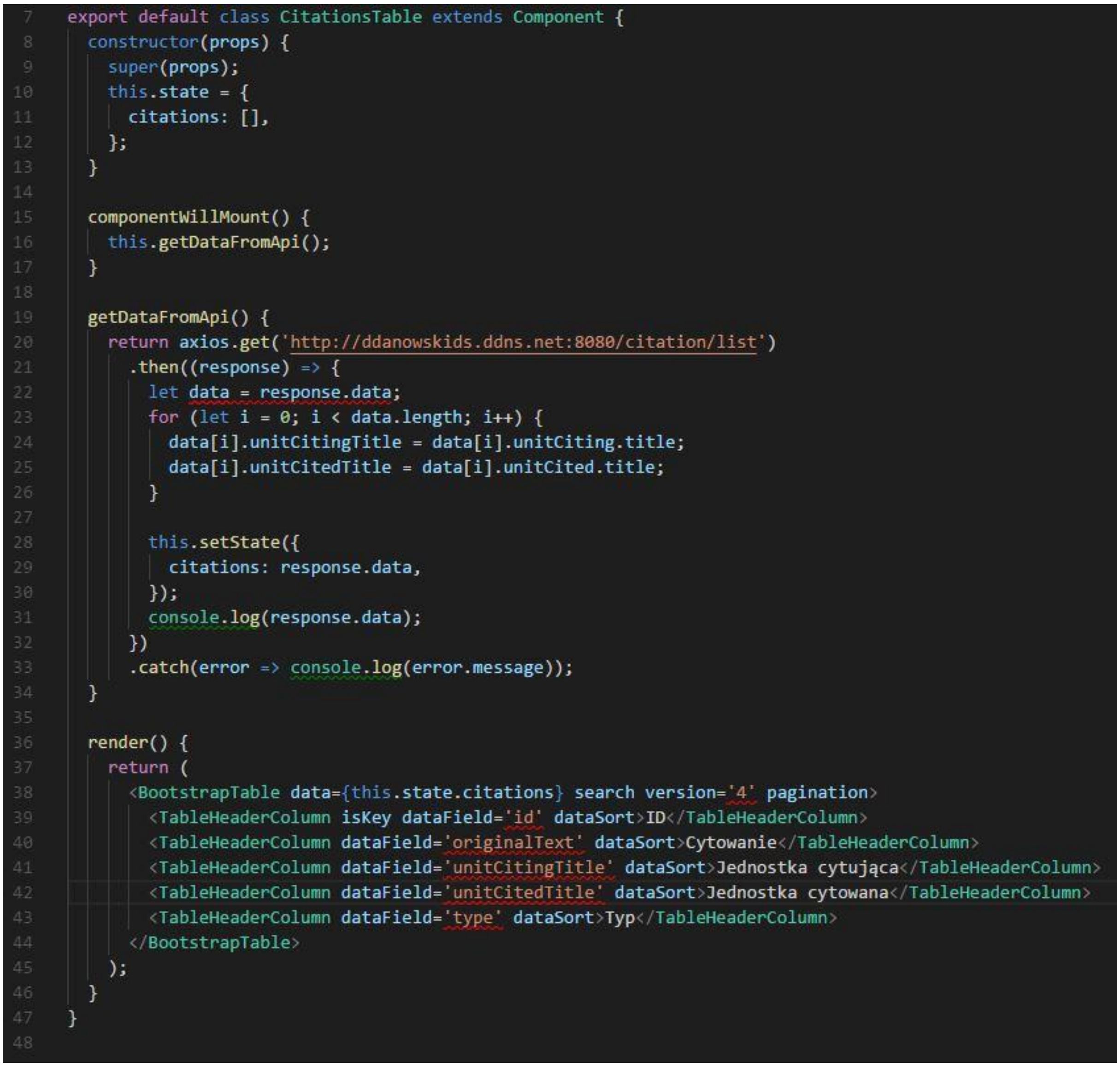
1. Pobieranie danych z bazy danych i wyświetlanie w formie tabel (do przeglądania):

   a) Tabela z autorami



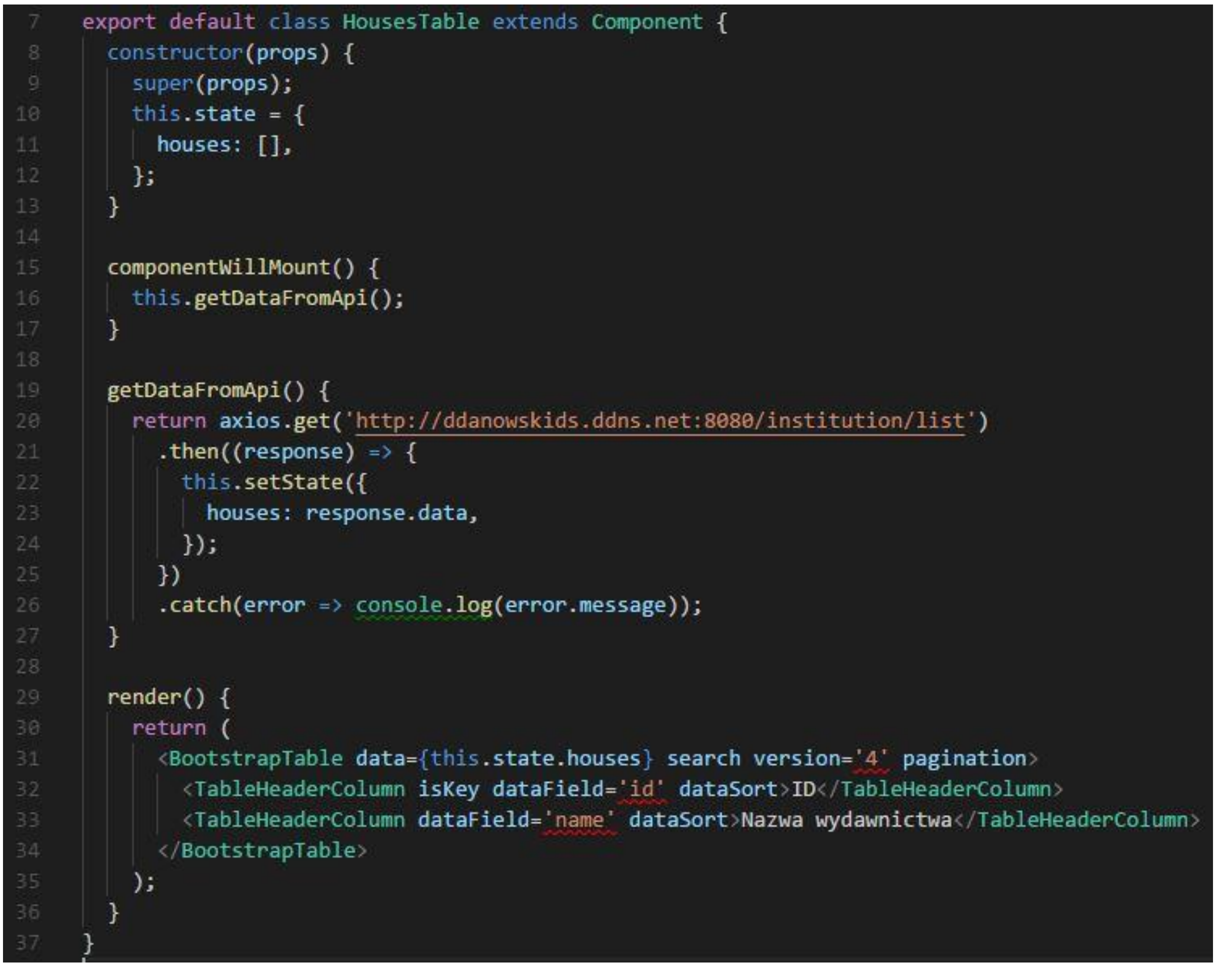
```
export default class AuthorsTable extends Component {
  constructor(props) {
    super(props);
    this.state = {
      authors: [],
    };
  }

  componentWillMount() {
    this.getDataFromApi();
  }

  getDataFromApi() {
    return axios.get('http://ddanowskids.ddns.net:8080/person/list')
      .then((response) => {
        this.setState({
          authors: response.data,
        });
      })
      .catch(error => console.log(error.message));
  }

  render() {
    return (
      <BootstrapTable data={this.state.authors} search version='4' pagination>
        <TableHeaderColumn isKey dataField='id' dataSort>ID</TableHeaderColumn>
        <TableHeaderColumn dataField='surname' dataSort>Nazwisko</TableHeaderColumn>
        <TableHeaderColumn dataField='name' dataSort>Imię</TableHeaderColumn>
        <TableHeaderColumn dataField='name2' dataSort>Drugie imię</TableHeaderColumn>
        <TableHeaderColumn dataField='orcid' dataSort>Numer ORCID</TableHeaderColumn>
      </BootstrapTable>
    );
  }
}
```

b) Tabela z cytowaniami



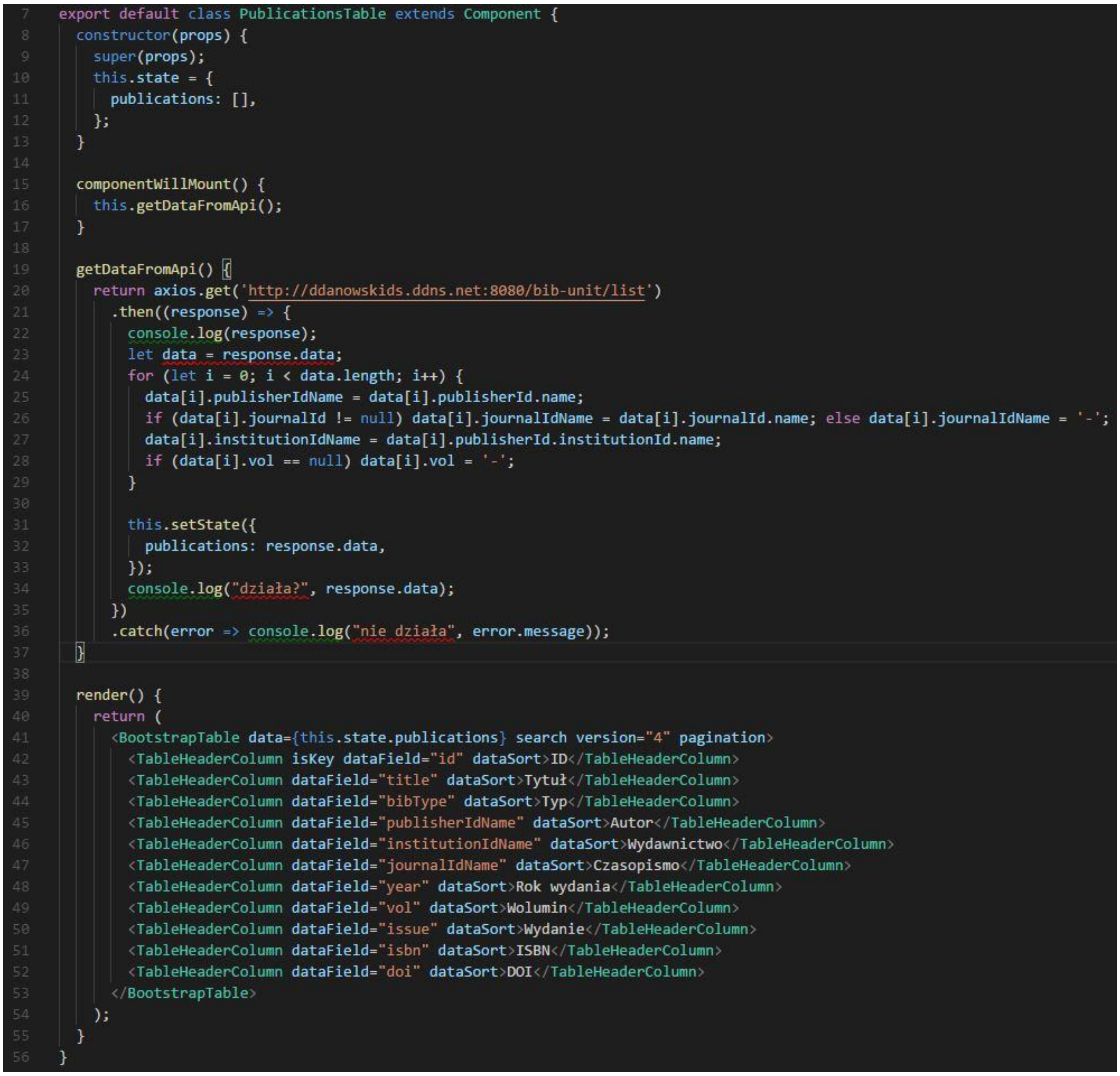
```
export default class CitationsTable extends Component {
  constructor(props) {
    super(props);
    this.state = {
      citations: [],
    };
  }

  componentWillMount() {
    this.getDataFromApi();
  }

  getDataFromApi() {
    return axios.get('http://ddanowskids.ddns.net:8080/citation/list')
      .then((response) => {
        let data = response.data;
        for (let i = 0; i < data.length; i++) {
          data[i].unitCitingTitle = data[i].unitCiting.title;
          data[i].unitCitedTitle = data[i].unitCited.title;
        }

        this.setState({
          citations: response.data,
        });
        console.log(response.data);
      })
      .catch(error => console.log(error.message));
  }

  render() {
    return (
      <BootstrapTable data={this.state.citations} search version='4' pagination>
        <TableHeaderColumn isKey dataField='id' dataSort>ID</TableHeaderColumn>
        <TableHeaderColumn dataField='originalText' dataSort>Cytowanie</TableHeaderColumn>
        <TableHeaderColumn dataField='unitCitingTitle' dataSort>Jednostka cytująca</TableHeaderColumn>
        <TableHeaderColumn dataField='unitCitedTitle' dataSort>Jednostka cytowana</TableHeaderColumn>
        <TableHeaderColumn dataField='type' dataSort>Typ</TableHeaderColumn>
      </BootstrapTable>
    );
  }
}
```

c) Tabela z wydawnictwami



```
export default class HousesTable extends Component {
  constructor(props) {
    super(props);
    this.state = {
      houses: [],
    };
  }

  componentWillMount() {
    this.getDataFromApi();
  }

  getDataFromApi() {
    return axios.get('http://ddanowskids.ddns.net:8080/institution/list')
      .then((response) => {
        this.setState({
          houses: response.data,
        });
      })
      .catch(error => console.log(error.message));
  }

  render() {
    return (
      <BootstrapTable data={this.state.houses} search version='4' pagination>
        <TableHeaderColumn isKey dataField='id' dataSort>ID</TableHeaderColumn>
        <TableHeaderColumn dataField='name' dataSort>Nazwa wydawnictwa</TableHeaderColumn>
      </BootstrapTable>
    );
  }
}
```

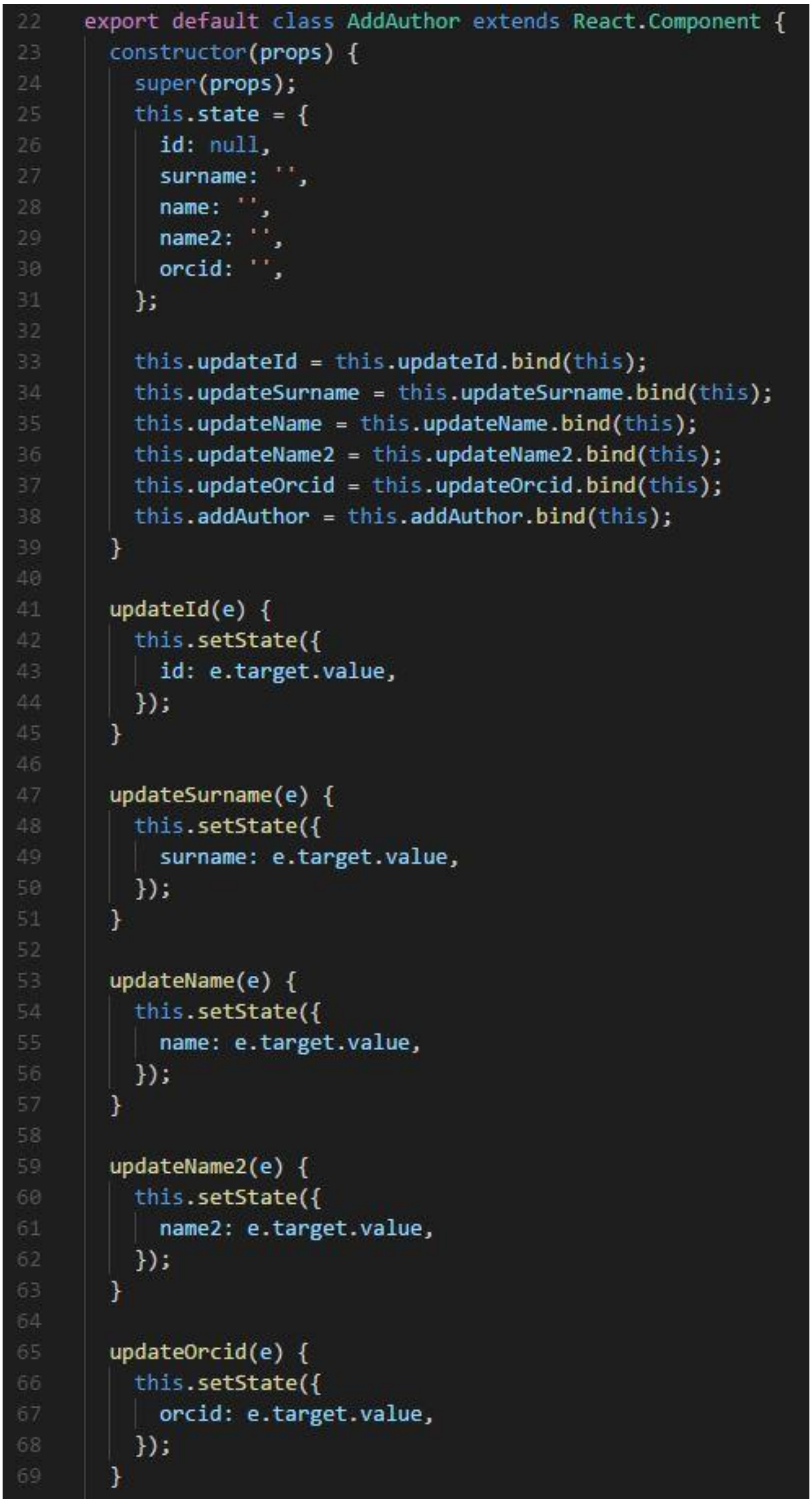
d) Tabela z publikacjami



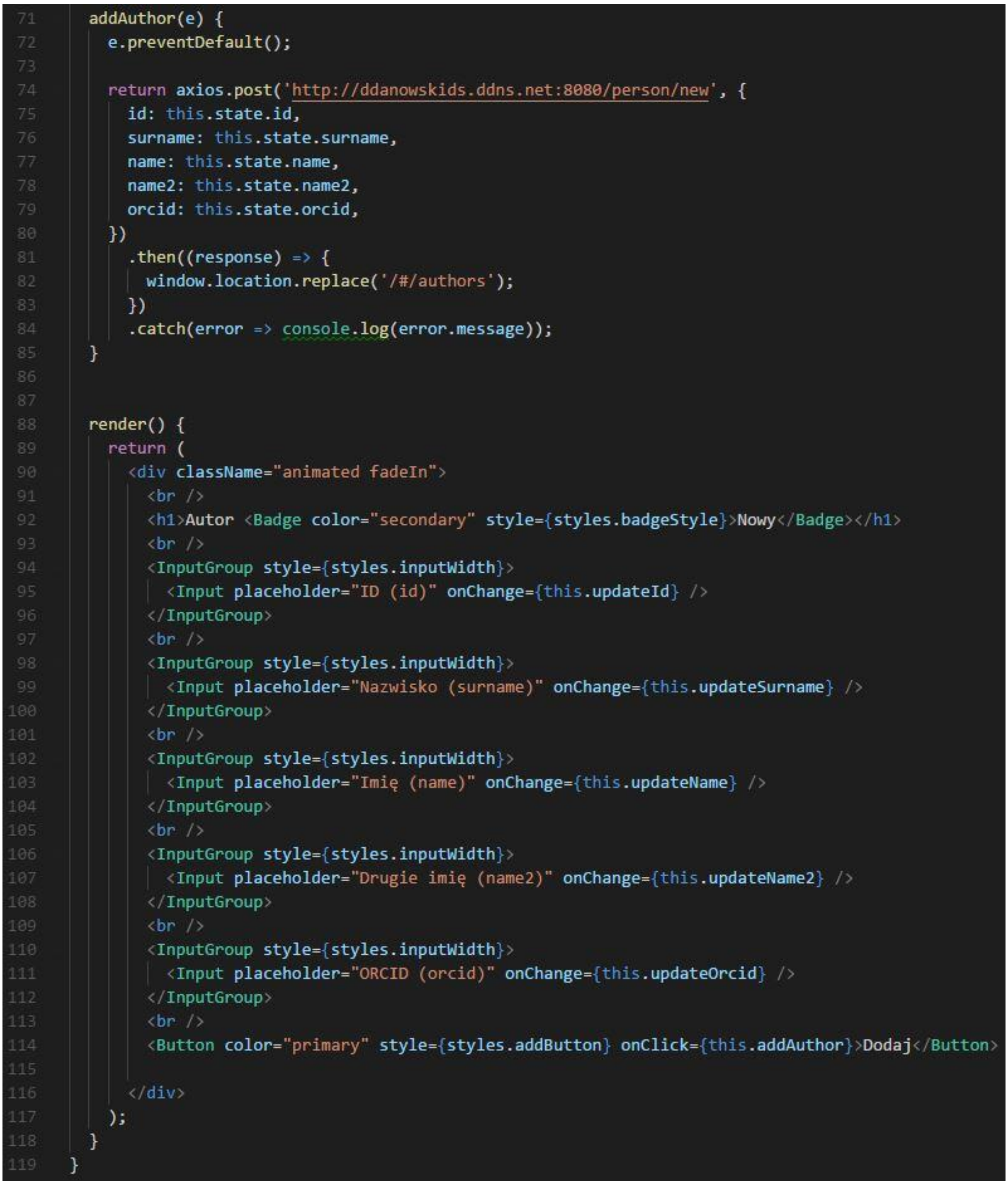
```
export default class PublicationsTable extends Component {
  constructor(props) {
    super(props);
    this.state = {
      publications: [],
    };
  }

  componentWillMount() {
    this.getDataFromApi();
  }

  getDataFromApi() {
    return axios.get('http://ddanowskids.ddns.net:8080/bib-unit/list')
      .then((response) => {
        console.log(response);
        let data = response.data;
        for (let i = 0; i < data.length; i++) {
          data[i].publisherIdName = data[i].publisherId.name;
          if (data[i].journalId != null) data[i].journalIdName = data[i].journalId.name; else data[i].journalIdName = '-';
          data[i].institutionIdName = data[i].publisherId.institutionId.name;
          if (data[i].vol == null) data[i].vol = '-';
        }

        this.setState({
          publications: response.data,
        });
        console.log("działa?", response.data);
      })
      .catch(error => console.log("nie działa", error.message));
  }

  render() {
    return (
      <BootstrapTable data={this.state.publications} search version="4" pagination>
        <TableHeaderColumn isKey dataField="id" dataSort>ID</TableHeaderColumn>
        <TableHeaderColumn dataField="title" dataSort>Tytuł</TableHeaderColumn>
        <TableHeaderColumn dataField="bibType" dataSort>Typ</TableHeaderColumn>
        <TableHeaderColumn dataField="publisherIdName" dataSort>Autor</TableHeaderColumn>
        <TableHeaderColumn dataField="institutionIdName" dataSort>Wydawnictwo</TableHeaderColumn>
        <TableHeaderColumn dataField="journalIdName" dataSort>Czasopismo</TableHeaderColumn>
        <TableHeaderColumn dataField="year" dataSort>Rok wydania</TableHeaderColumn>
        <TableHeaderColumn dataField="vol" dataSort>Wolumin</TableHeaderColumn>
        <TableHeaderColumn dataField="issue" dataSort>Wydanie</TableHeaderColumn>
        <TableHeaderColumn dataField="isbn" dataSort>ISBN</TableHeaderColumn>
        <TableHeaderColumn dataField="doi" dataSort>DOI</TableHeaderColumn>
      </BootstrapTable>
    );
  }
}
```

2. Dodawanie nowych rekordów:

   a) Dodawanie autora

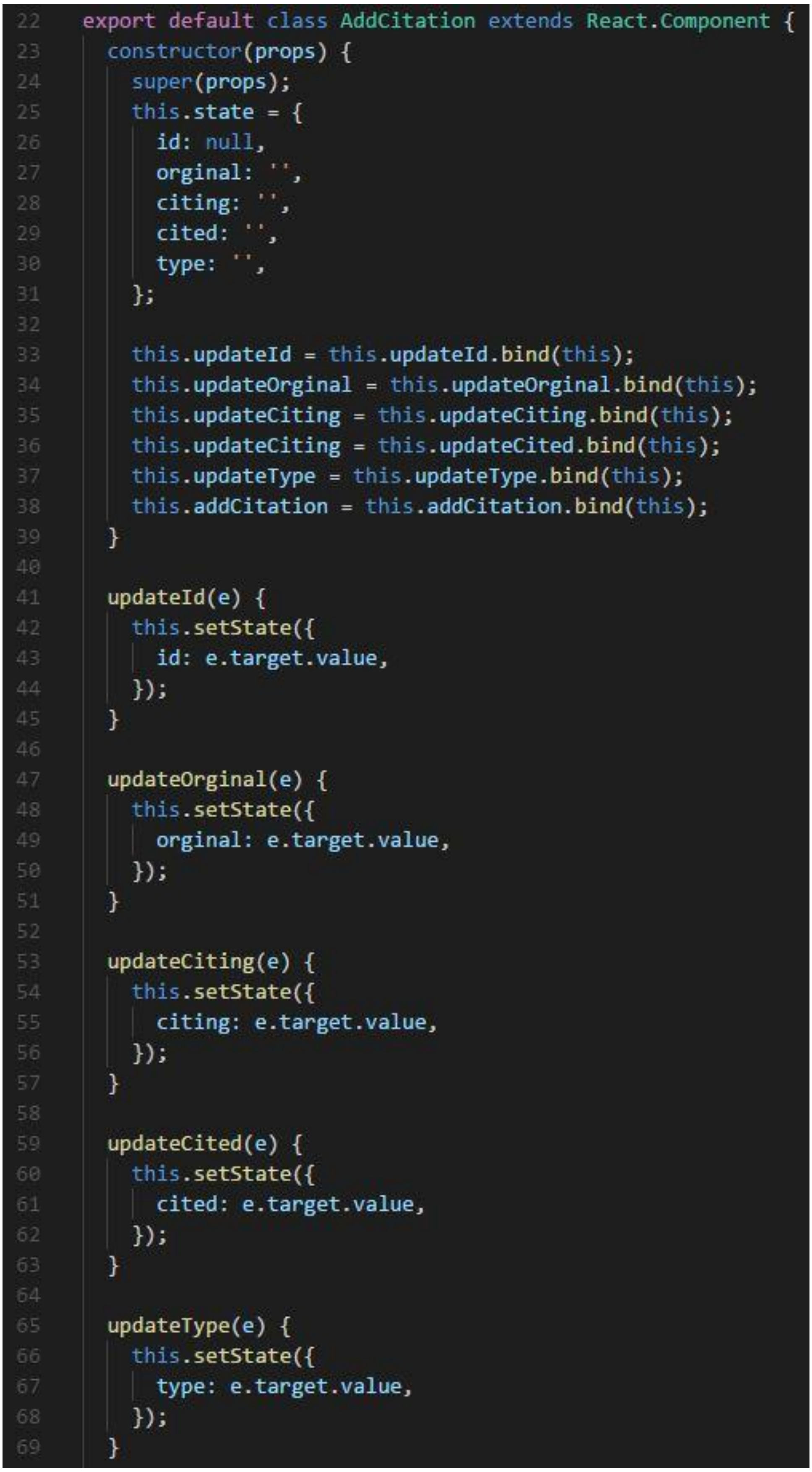


```
export default class AddAuthor extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      id: null,
      surname: '',
      name: '',
      name2: '',
      orcid: '',
    };

    this.updateId = this.updateId.bind(this);
    this.updateSurname = this.updateSurname.bind(this);
    this.updateName = this.updateName.bind(this);
    this.updateName2 = this.updateName2.bind(this);
    this.updateOrcid = this.updateOrcid.bind(this);
    this.addAuthor = this.addAuthor.bind(this);
  }

  updateId(e) {
    this.setState({
      id: e.target.value,
    });
  }

  updateSurname(e) {
    this.setState({
      surname: e.target.value,
    });
  }

  updateName(e) {
    this.setState({
      name: e.target.value,
    });
  }

  updateName2(e) {
    this.setState({
      name2: e.target.value,
    });
  }

  updateOrcid(e) {
    this.setState({
      orcid: e.target.value,
    });
  }
```

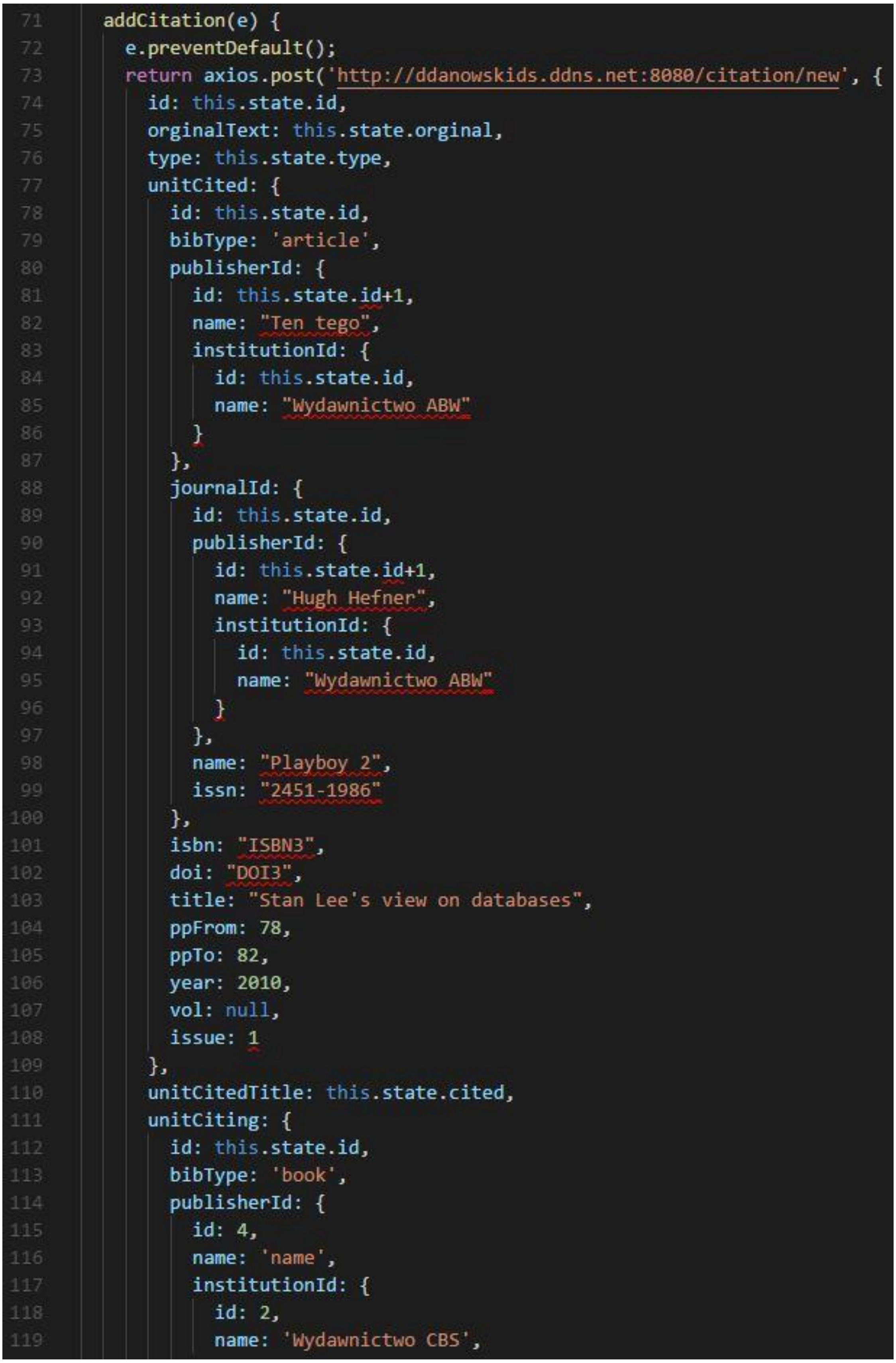


```
  addAuthor(e) {
    e.preventDefault();

    return axios.post('http://ddanowskids.ddns.net:8080/person/new', {
      id: this.state.id,
      surname: this.state.surname,
      name: this.state.name,
      name2: this.state.name2,
      orcid: this.state.orcid,
    })
      .then((response) => {
        window.location.replace('/#/authors');
      })
      .catch(error => console.log(error.message));
  }


  render() {
    return (
      <div className="animated fadeIn">
        <br />
        <h1>Autor <Badge color="secondary" style={styles.badgeStyle}>Nowy</Badge></h1>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="ID (id)" onChange={this.updateId} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Nazwisko (surname)" onChange={this.updateSurname} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Imię (name)" onChange={this.updateName} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Drugie imię (name2)" onChange={this.updateName2} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="ORCID (orcid)" onChange={this.updateOrcid} />
        </InputGroup>
        <br />
        <Button color="primary" style={styles.addButton} onClick={this.addAuthor}>Dodaj</Button>

      </div>
    );
  }
}
```

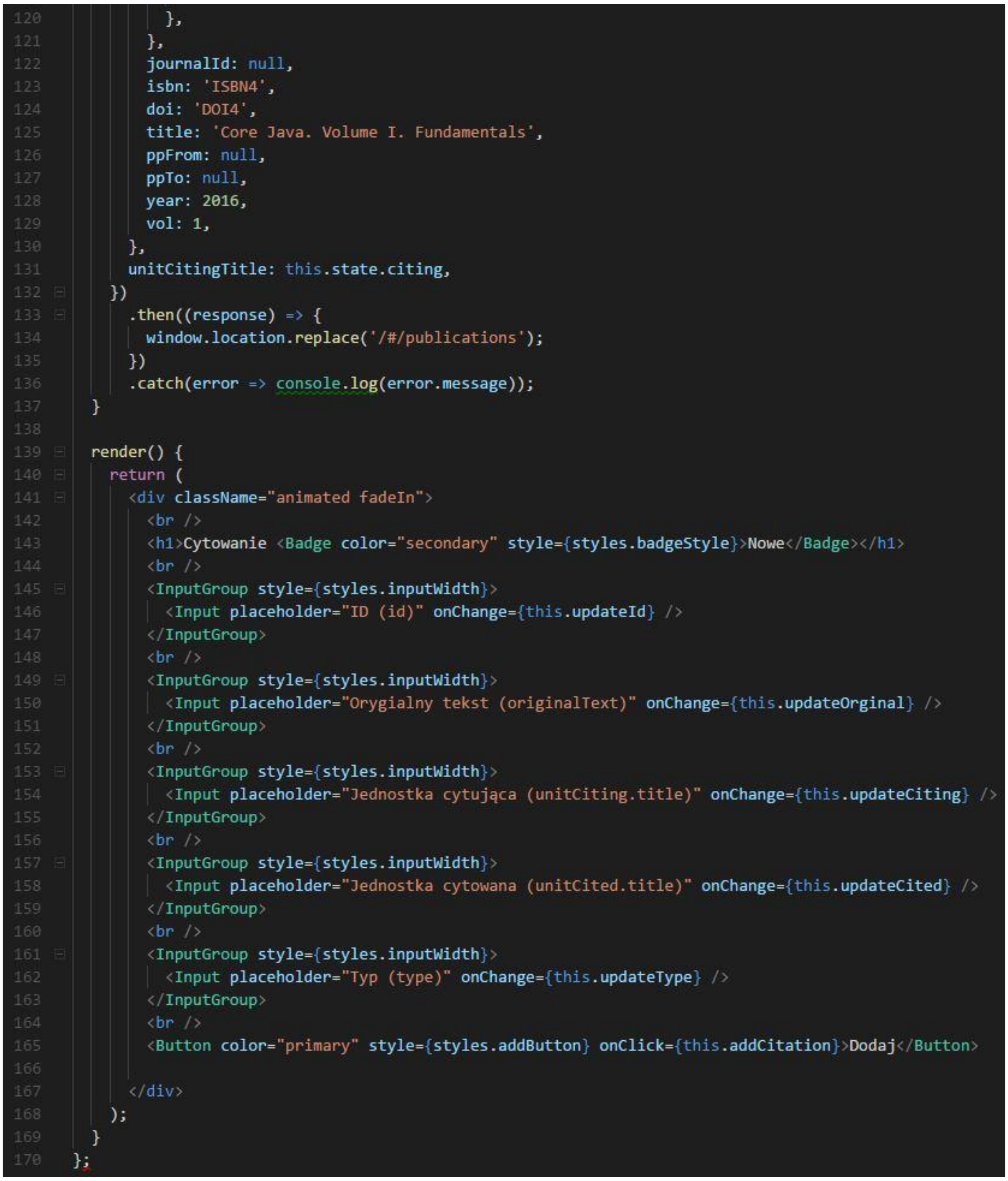
b) Dodawanie cytowania



```
export default class AddCitation extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      id: null,
      orginal: '',
      citing: '',
      cited: '',
      type: '',
    };

    this.updateId = this.updateId.bind(this);
    this.updateOrginal = this.updateOrginal.bind(this);
    this.updateCiting = this.updateCiting.bind(this);
    this.updateCiting = this.updateCited.bind(this);
    this.updateType = this.updateType.bind(this);
    this.addCitation = this.addCitation.bind(this);
  }

  updateId(e) {
    this.setState({
      id: e.target.value,
    });
  }

  updateOrginal(e) {
    this.setState({
      orginal: e.target.value,
    });
  }

  updateCiting(e) {
    this.setState({
      citing: e.target.value,
    });
  }

  updateCited(e) {
    this.setState({
      cited: e.target.value,
    });
  }

  updateType(e) {
    this.setState({
      type: e.target.value,
    });
  }
```

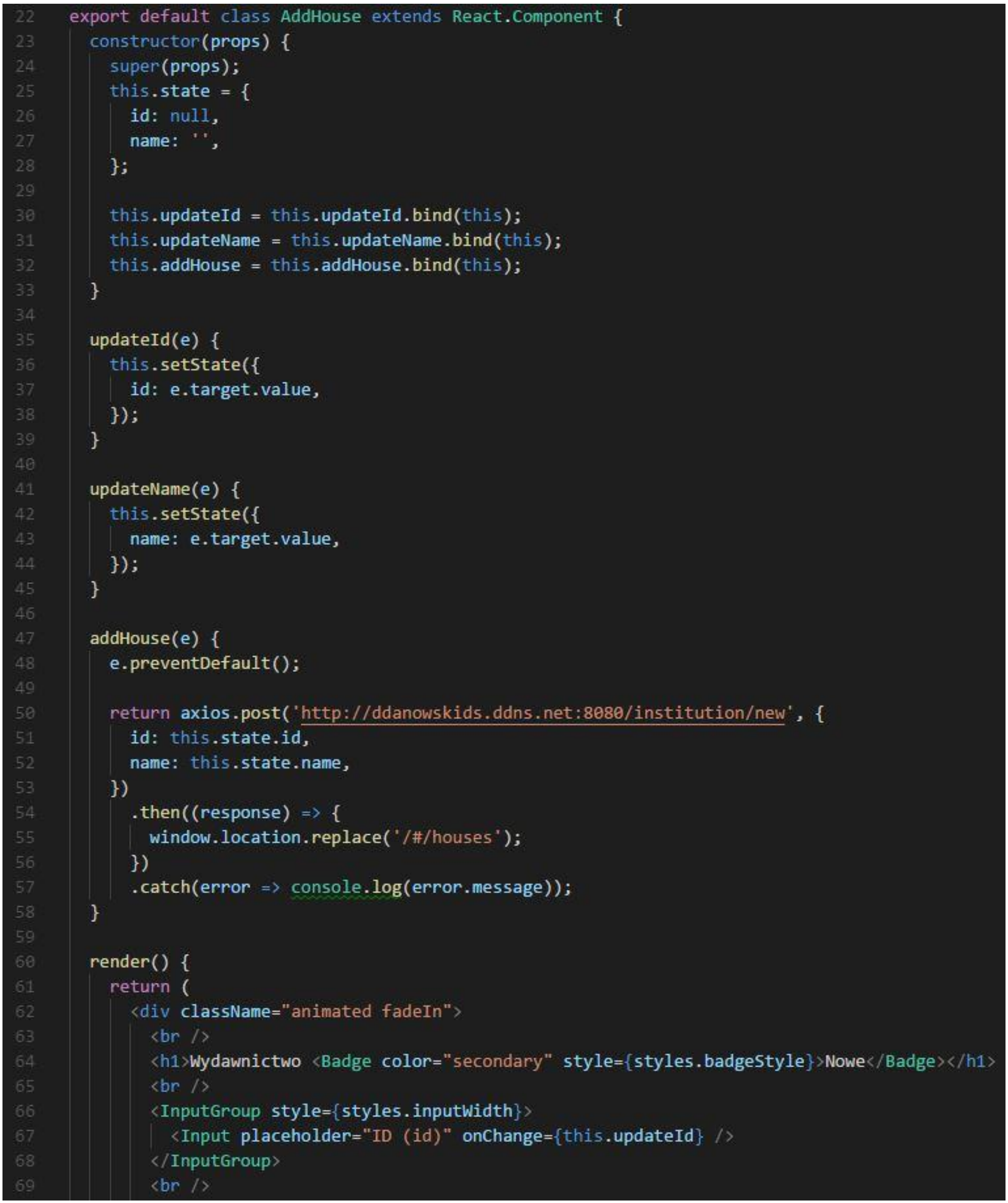


```
  addCitation(e) {
    e.preventDefault();
    return axios.post('http://ddanowskids.ddns.net:8080/citation/new', {
      id: this.state.id,
      orginalText: this.state.orginal,
      type: this.state.type,
      unitCited: {
        id: this.state.id,
        bibType: 'article',
        publisherId: {
          id: this.state.id+1,
          name: "Ten tego",
          institutionId: {
            id: this.state.id,
            name: "Wydawnictwo ABW"
          }
        },
        journalId: {
          id: this.state.id,
          publisherId: {
            id: this.state.id+1,
            name: "Hugh Hefner",
            institutionId: {
              id: this.state.id,
              name: "Wydawnictwo ABW"
            }
          },
          name: "Playboy 2",
          issn: "2451-1986"
        },
        isbn: "ISBN3",
        doi: "DOI3",
        title: "Stan Lee's view on databases",
        ppFrom: 78,
        ppTo: 82,
        year: 2010,
        vol: null,
        issue: 1
      },
      unitCitedTitle: this.state.cited,
      unitCiting: {
        id: this.state.id,
        bibType: 'book',
        publisherId: {
          id: 4,
          name: 'name',
          institutionId: {
            id: 2,
            name: 'Wydawnictwo CBS',
```

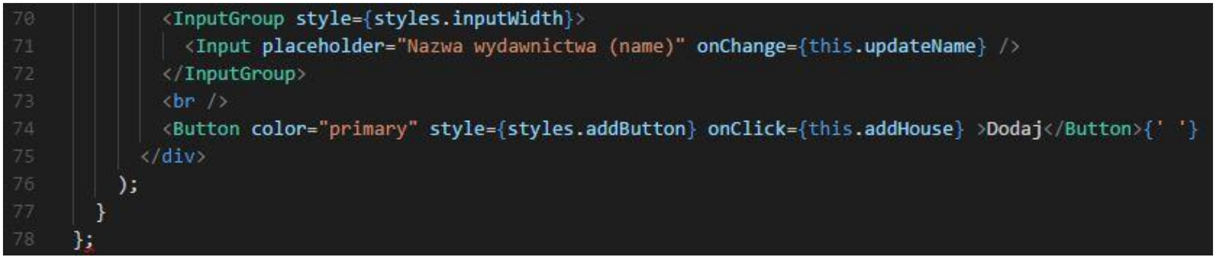


```
          },
        },
        journalId: null,
        isbn: 'ISBN4',
        doi: 'DOI4',
        title: 'Core Java. Volume I. Fundamentals',
        ppFrom: null,
        ppTo: null,
        year: 2016,
        vol: 1,
      },
      unitCitingTitle: this.state.citing,
    })
      .then((response) => {
        window.location.replace('/#/publications');
      })
      .catch(error => console.log(error.message));
  }

  render() {
    return (
      <div className="animated fadeIn">
        <br />
        <h1>Cytowanie <Badge color="secondary" style={styles.badgeStyle}>Nowe</Badge></h1>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="ID (id)" onChange={this.updateId} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Orygialny tekst (originalText)" onChange={this.updateOrginal} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Jednostka cytująca (unitCiting.title)" onChange={this.updateCiting} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Jednostka cytowana (unitCited.title)" onChange={this.updateCited} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Typ (type)" onChange={this.updateType} />
        </InputGroup>
        <br />
        <Button color="primary" style={styles.addButton} onClick={this.addCitation}>Dodaj</Button>

      </div>
    );
  }
};
```

c) Dodawanie wydawnictwa

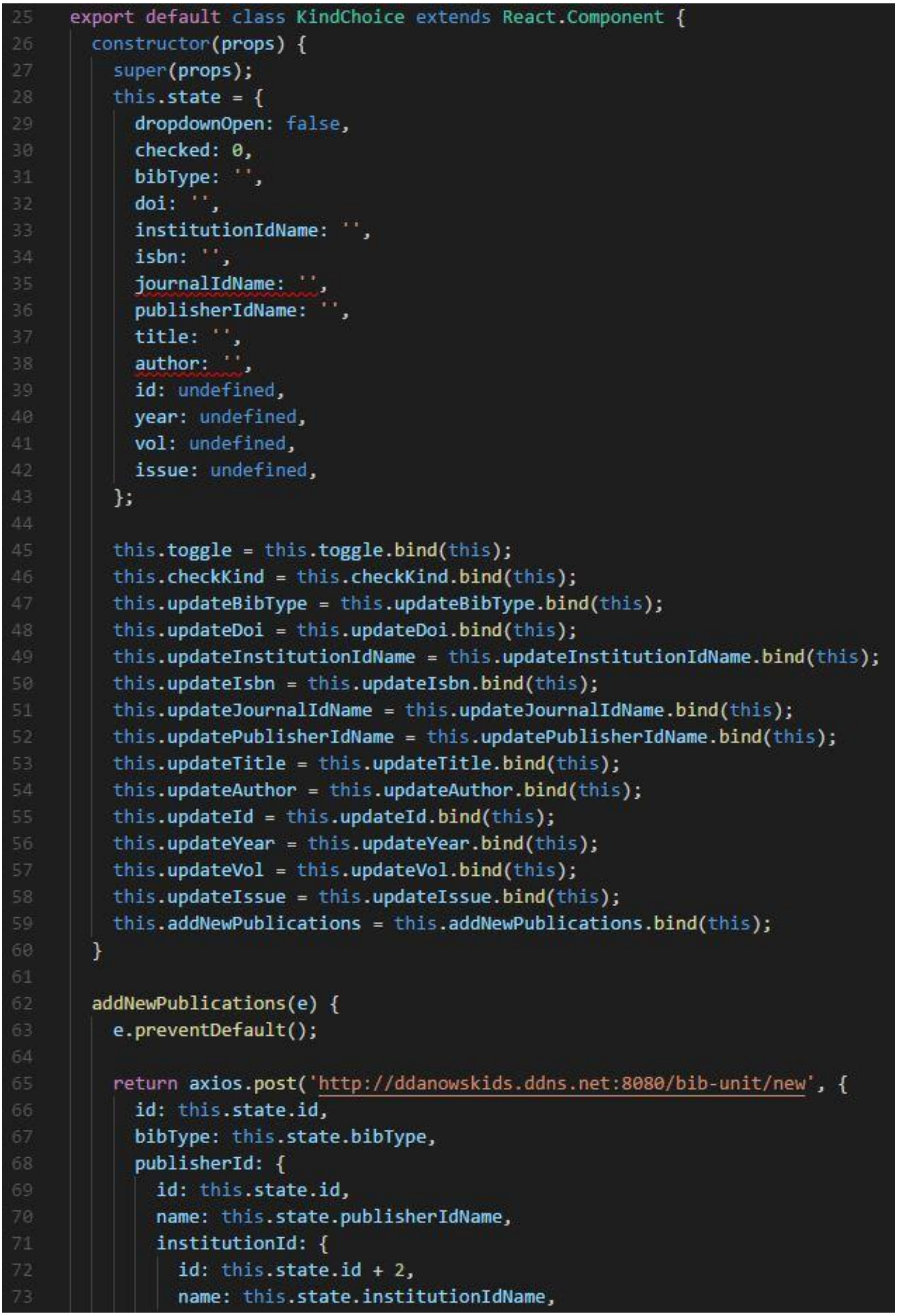


```
export default class AddHouse extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      id: null,
      name: '',
    };

    this.updateId = this.updateId.bind(this);
    this.updateName = this.updateName.bind(this);
    this.addHouse = this.addHouse.bind(this);
  }

  updateId(e) {
    this.setState({
      id: e.target.value,
    });
  }

  updateName(e) {
    this.setState({
      name: e.target.value,
    });
  }

  addHouse(e) {
    e.preventDefault();

    return axios.post('http://ddanowskids.ddns.net:8080/institution/new', {
      id: this.state.id,
      name: this.state.name,
    })
      .then((response) => {
        window.location.replace('/#/houses');
      })
      .catch(error => console.log(error.message));
  }

  render() {
    return (
      <div className="animated fadeIn">
        <br />
        <h1>Wydawnictwo <Badge color="secondary" style={styles.badgeStyle}>Nowe</Badge></h1>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="ID (id)" onChange={this.updateId} />
        </InputGroup>
        <br />
```

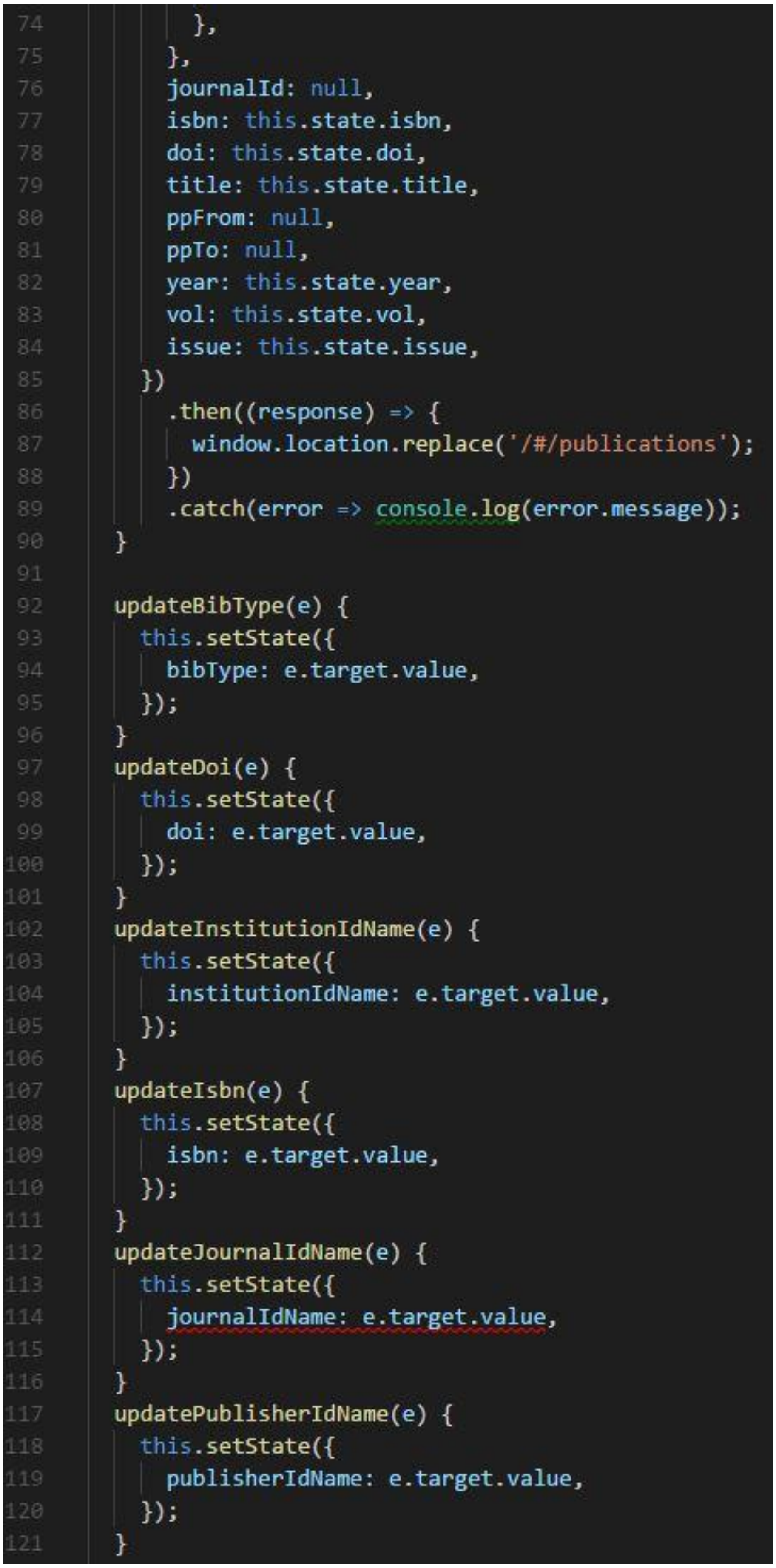


```
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Nazwa wydawnictwa (name)" onChange={this.updateName} />
        </InputGroup>
        <br />
        <Button color="primary" style={styles.addButton} onClick={this.addHouse} >Dodaj</Button>{' '}
      </div>
    );
  }
};
```

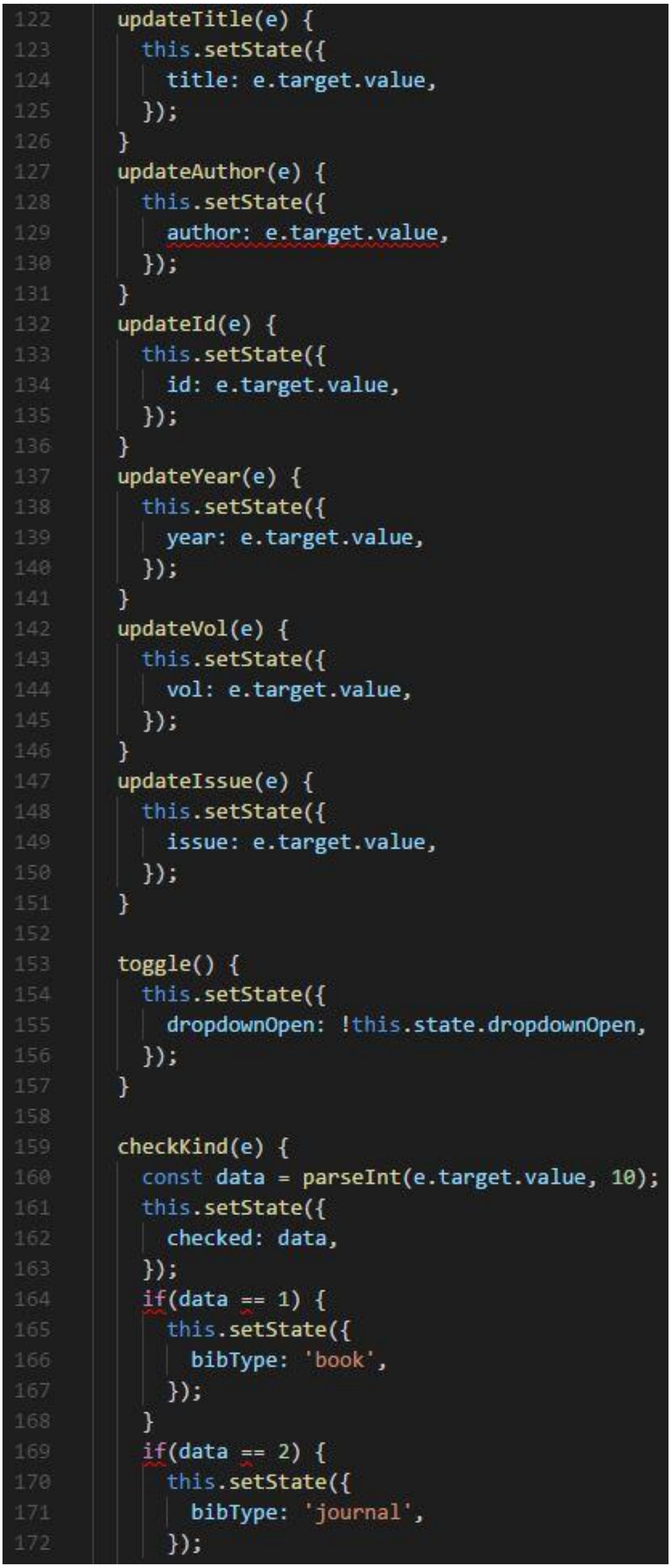
d) Dodawanie publikacji



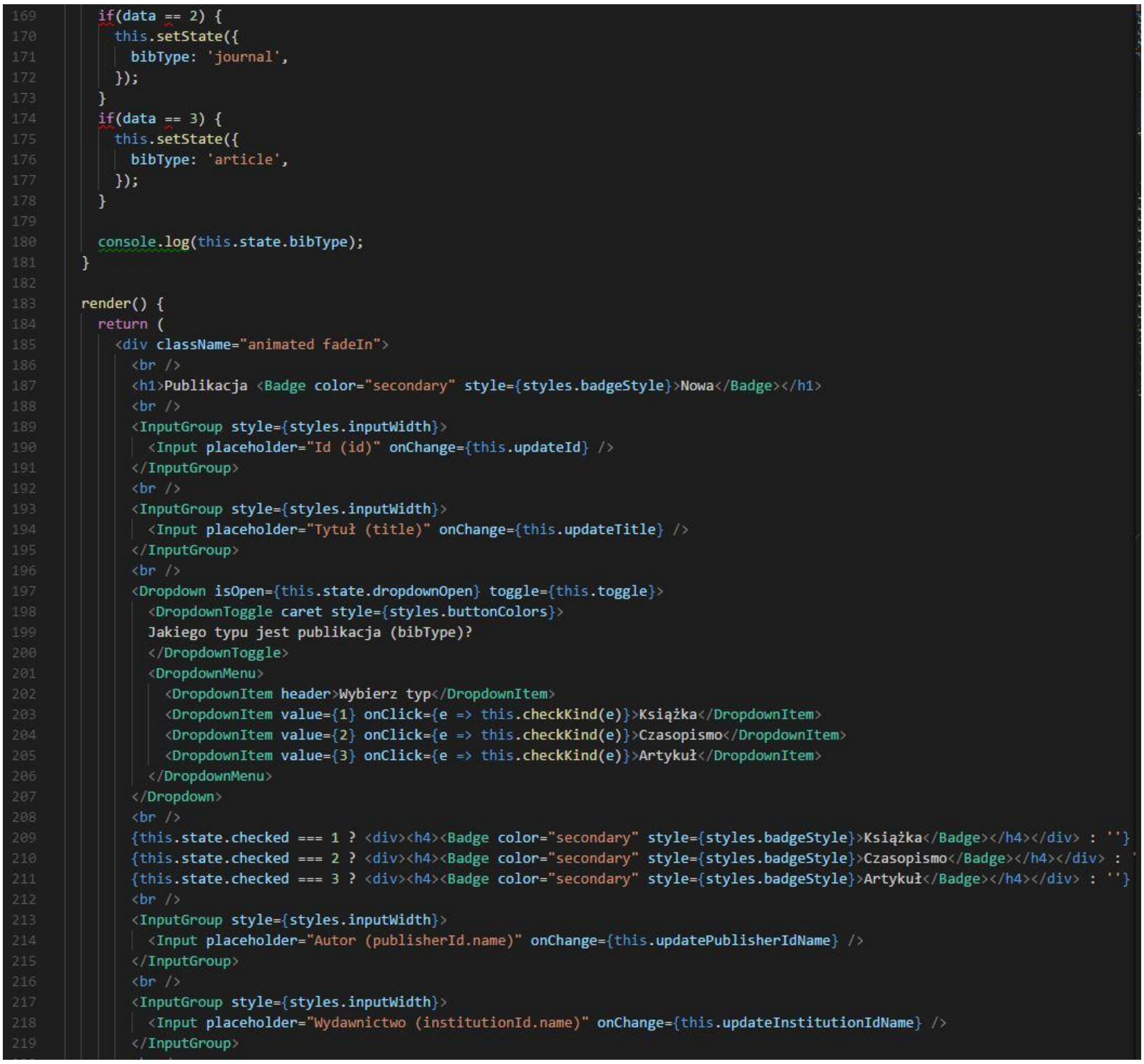
```
export default class KindChoice extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      dropdownOpen: false,
      checked: 0,
      bibType: '',
      doi: '',
      institutionIdName: '',
      isbn: '',
      journalIdName: '',
      publisherIdName: '',
      title: '',
      author: '',
      id: undefined,
      year: undefined,
      vol: undefined,
      issue: undefined,
    };

    this.toggle = this.toggle.bind(this);
    this.checkKind = this.checkKind.bind(this);
    this.updateBibType = this.updateBibType.bind(this);
    this.updateDoi = this.updateDoi.bind(this);
    this.updateInstitutionIdName = this.updateInstitutionIdName.bind(this);
    this.updateIsbn = this.updateIsbn.bind(this);
    this.updateJournalIdName = this.updateJournalIdName.bind(this);
    this.updatePublisherIdName = this.updatePublisherIdName.bind(this);
    this.updateTitle = this.updateTitle.bind(this);
    this.updateAuthor = this.updateAuthor.bind(this);
    this.updateId = this.updateId.bind(this);
    this.updateYear = this.updateYear.bind(this);
    this.updateVol = this.updateVol.bind(this);
    this.updateIssue = this.updateIssue.bind(this);
    this.addNewPublications = this.addNewPublications.bind(this);
  }

  addNewPublications(e) {
    e.preventDefault();

    return axios.post('http://ddanowskids.ddns.net:8080/bib-unit/new', {
      id: this.state.id,
      bibType: this.state.bibType,
      publisherId: {
        id: this.state.id,
        name: this.state.publisherIdName,
        institutionId: {
          id: this.state.id + 2,
          name: this.state.institutionIdName,
```



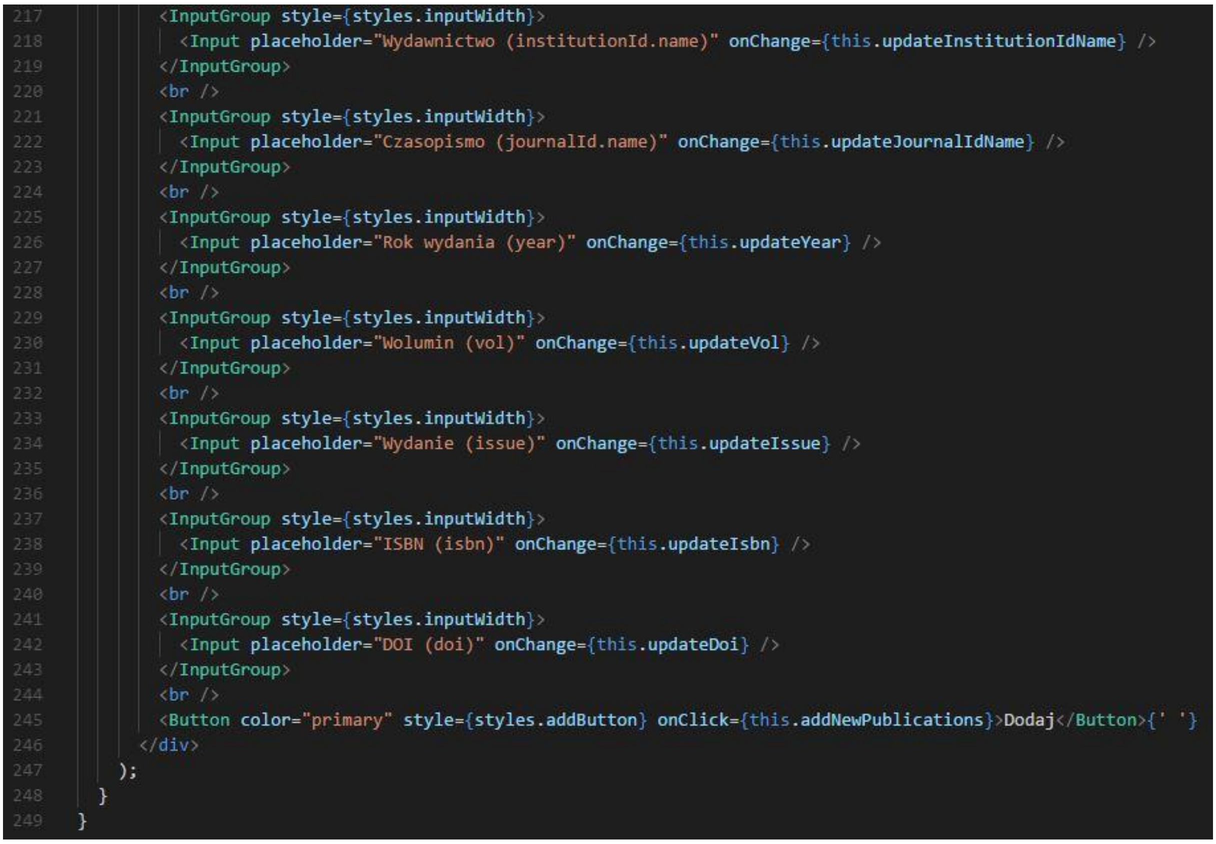
```
        },
      },
      journalId: null,
      isbn: this.state.isbn,
      doi: this.state.doi,
      title: this.state.title,
      ppFrom: null,
      ppTo: null,
      year: this.state.year,
      vol: this.state.vol,
      issue: this.state.issue,
    })
      .then((response) => {
        window.location.replace('/#/publications');
      })
      .catch(error => console.log(error.message));
  }

  updateBibType(e) {
    this.setState({
      bibType: e.target.value,
    });
  }
  updateDoi(e) {
    this.setState({
      doi: e.target.value,
    });
  }
  updateInstitutionIdName(e) {
    this.setState({
      institutionIdName: e.target.value,
    });
  }
  updateIsbn(e) {
    this.setState({
      isbn: e.target.value,
    });
  }
  updateJournalIdName(e) {
    this.setState({
      journalIdName: e.target.value,
    });
  }
  updatePublisherIdName(e) {
    this.setState({
      publisherIdName: e.target.value,
    });
  }
```



```
  updateTitle(e) {
    this.setState({
      title: e.target.value,
    });
  }
  updateAuthor(e) {
    this.setState({
      author: e.target.value,
    });
  }
  updateId(e) {
    this.setState({
      id: e.target.value,
    });
  }
  updateYear(e) {
    this.setState({
      year: e.target.value,
    });
  }
  updateVol(e) {
    this.setState({
      vol: e.target.value,
    });
  }
  updateIssue(e) {
    this.setState({
      issue: e.target.value,
    });
  }

  toggle() {
    this.setState({
      dropdownOpen: !this.state.dropdownOpen,
    });
  }

  checkKind(e) {
    const data = parseInt(e.target.value, 10);
    this.setState({
      checked: data,
    });
    if(data == 1) {
      this.setState({
        bibType: 'book',
      });
    }
    if(data == 2) {
      this.setState({
        bibType: 'journal',
      });
```

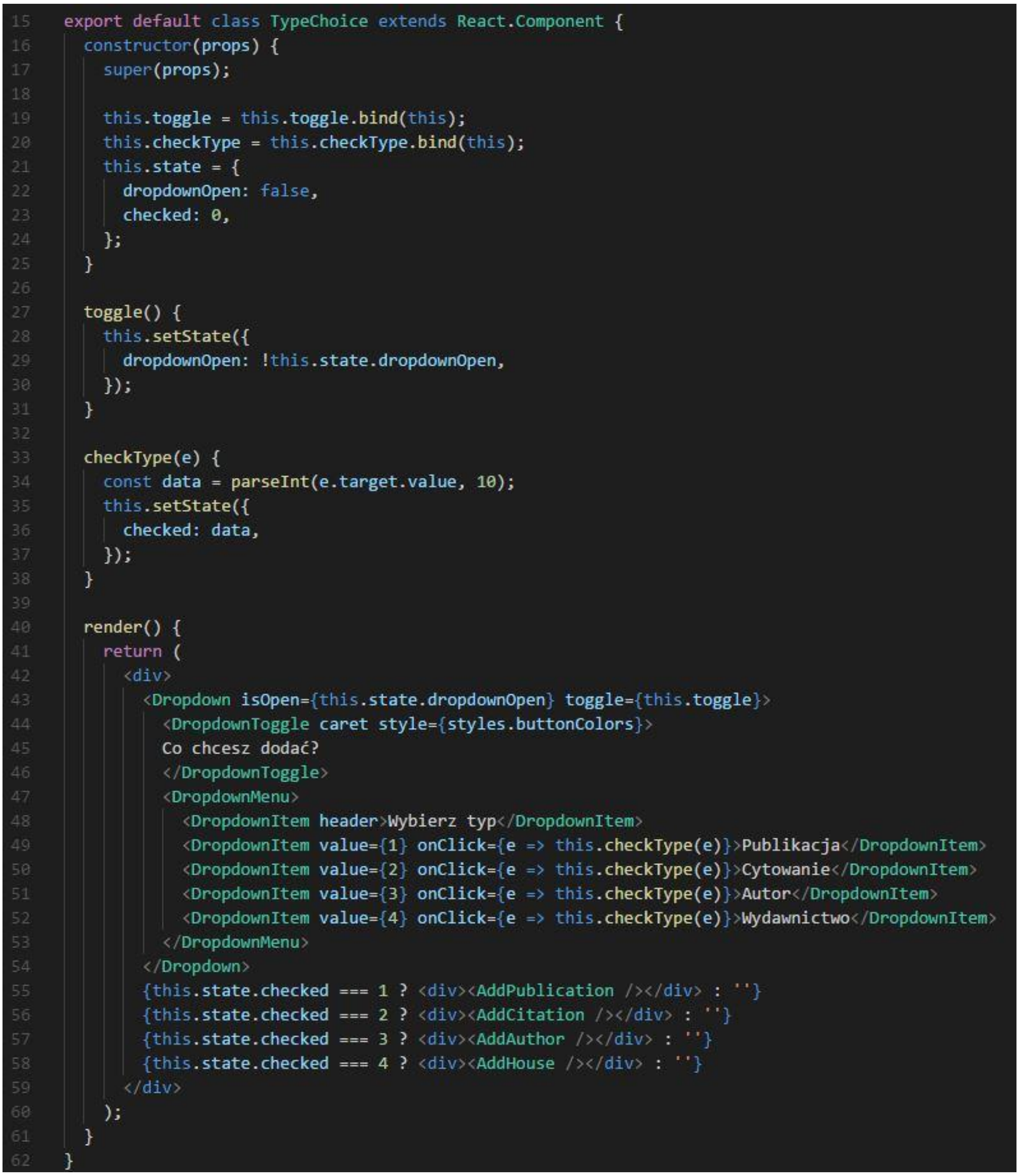


```
    if(data == 2) {
      this.setState({
        bibType: 'journal',
      });
    }
    if(data == 3) {
      this.setState({
        bibType: 'article',
      });
    }

    console.log(this.state.bibType);
  }

  render() {
    return (
      <div className="animated fadeIn">
        <br />
        <h1>Publikacja <Badge color="secondary" style={styles.badgeStyle}>Nowa</Badge></h1>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Id (id)" onChange={this.updateId} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Tytuł (title)" onChange={this.updateTitle} />
        </InputGroup>
        <br />
        <Dropdown isOpen={this.state.dropdownOpen} toggle={this.toggle}>
          <DropdownToggle caret style={styles.buttonColors}>
          Jakiego typu jest publikacja (bibType)?
          </DropdownToggle>
          <DropdownMenu>
            <DropdownItem header>Wybierz typ</DropdownItem>
            <DropdownItem value={1} onClick={e => this.checkKind(e)}>Książka</DropdownItem>
            <DropdownItem value={2} onClick={e => this.checkKind(e)}>Czasopismo</DropdownItem>
            <DropdownItem value={3} onClick={e => this.checkKind(e)}>Artykuł</DropdownItem>
          </DropdownMenu>
        </Dropdown>
        <br />
        {this.state.checked === 1 ? <div><h4><Badge color="secondary" style={styles.badgeStyle}>Książka</Badge></h4></div> : ''}
        {this.state.checked === 2 ? <div><h4><Badge color="secondary" style={styles.badgeStyle}>Czasopismo</Badge></h4></div> : '
        {this.state.checked === 3 ? <div><h4><Badge color="secondary" style={styles.badgeStyle}>Artykuł</Badge></h4></div> : ''}
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Autor (publisherId.name)" onChange={this.updatePublisherIdName} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Wydawnictwo (institutionId.name)" onChange={this.updateInstitutionIdName} />
        </InputGroup>
```



```
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Wydawnictwo (institutionId.name)" onChange={this.updateInstitutionIdName} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Czasopismo (journalId.name)" onChange={this.updateJournalIdName} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Rok wydania (year)" onChange={this.updateYear} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Wolumin (vol)" onChange={this.updateVol} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="Wydanie (issue)" onChange={this.updateIssue} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="ISBN (isbn)" onChange={this.updateIsbn} />
        </InputGroup>
        <br />
        <InputGroup style={styles.inputWidth}>
          <Input placeholder="DOI (doi)" onChange={this.updateDoi} />
        </InputGroup>
        <br />
        <Button color="primary" style={styles.addButton} onClick={this.addNewPublications}>Dodaj</Button>{' '}
      </div>
    );
  }
}
```

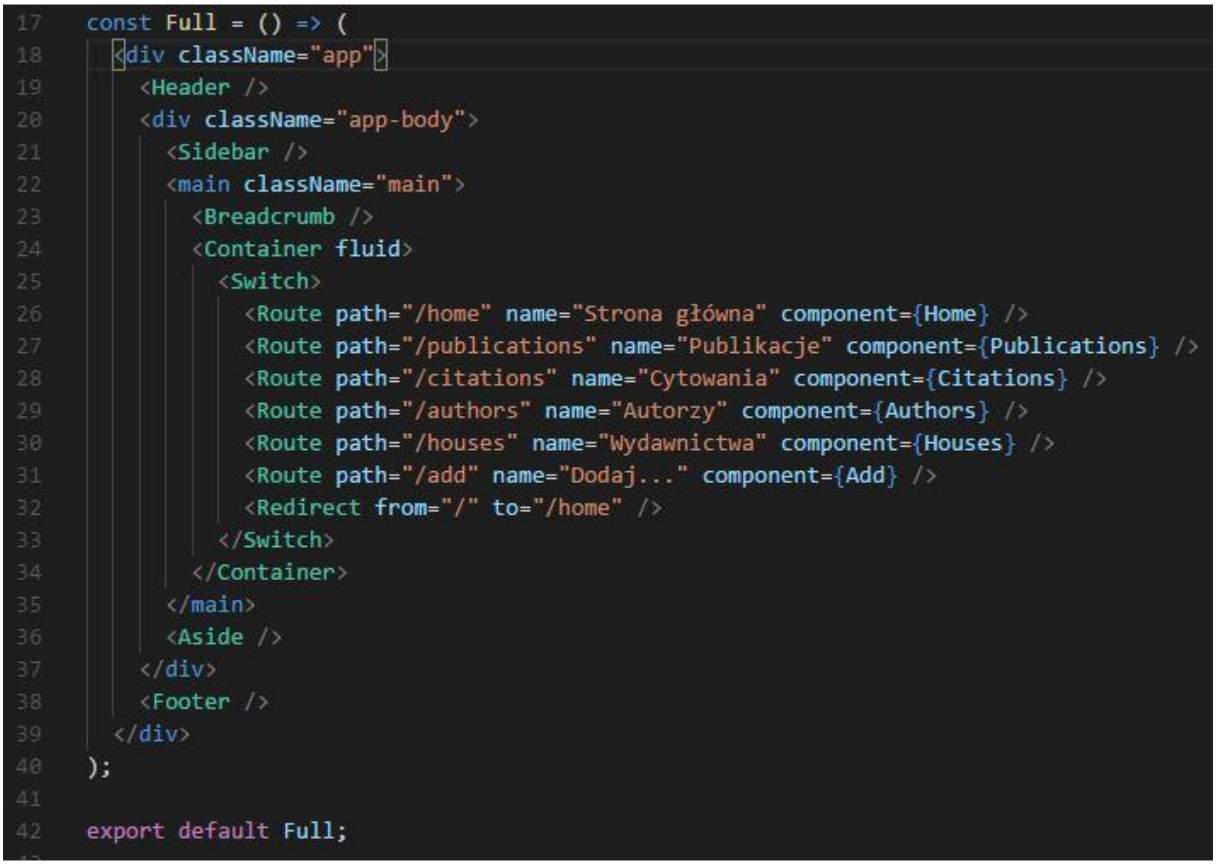
e) Wybór dodawania



```
export default class TypeChoice extends React.Component {
  constructor(props) {
    super(props);

    this.toggle = this.toggle.bind(this);
    this.checkType = this.checkType.bind(this);
    this.state = {
      dropdownOpen: false,
      checked: 0,
    };
  }

  toggle() {
    this.setState({
      dropdownOpen: !this.state.dropdownOpen,
    });
  }

  checkType(e) {
    const data = parseInt(e.target.value, 10);
    this.setState({
      checked: data,
    });
  }

  render() {
    return (
      <div>
        <Dropdown isOpen={this.state.dropdownOpen} toggle={this.toggle}>
          <DropdownToggle caret style={styles.buttonColors}>
          Co chcesz dodać?
          </DropdownToggle>
          <DropdownMenu>
            <DropdownItem header>Wybierz typ</DropdownItem>
            <DropdownItem value={1} onClick={e => this.checkType(e)}>Publikacja</DropdownItem>
            <DropdownItem value={2} onClick={e => this.checkType(e)}>Cytowanie</DropdownItem>
            <DropdownItem value={3} onClick={e => this.checkType(e)}>Autor</DropdownItem>
            <DropdownItem value={4} onClick={e => this.checkType(e)}>Wydawnictwo</DropdownItem>
          </DropdownMenu>
        </Dropdown>
        {this.state.checked === 1 ? <div><AddPublication /></div> : ''}
        {this.state.checked === 2 ? <div><AddCitation /></div> : ''}
        {this.state.checked === 3 ? <div><AddAuthor /></div> : ''}
        {this.state.checked === 4 ? <div><AddHouse /></div> : ''}
      </div>
    );
  }
}
```

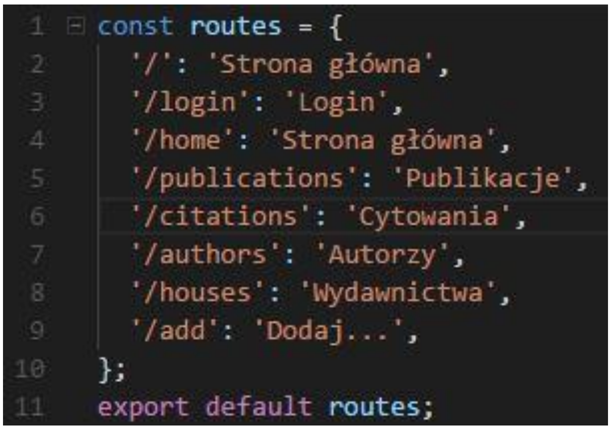
3. System zakładek

a) Ścieżki



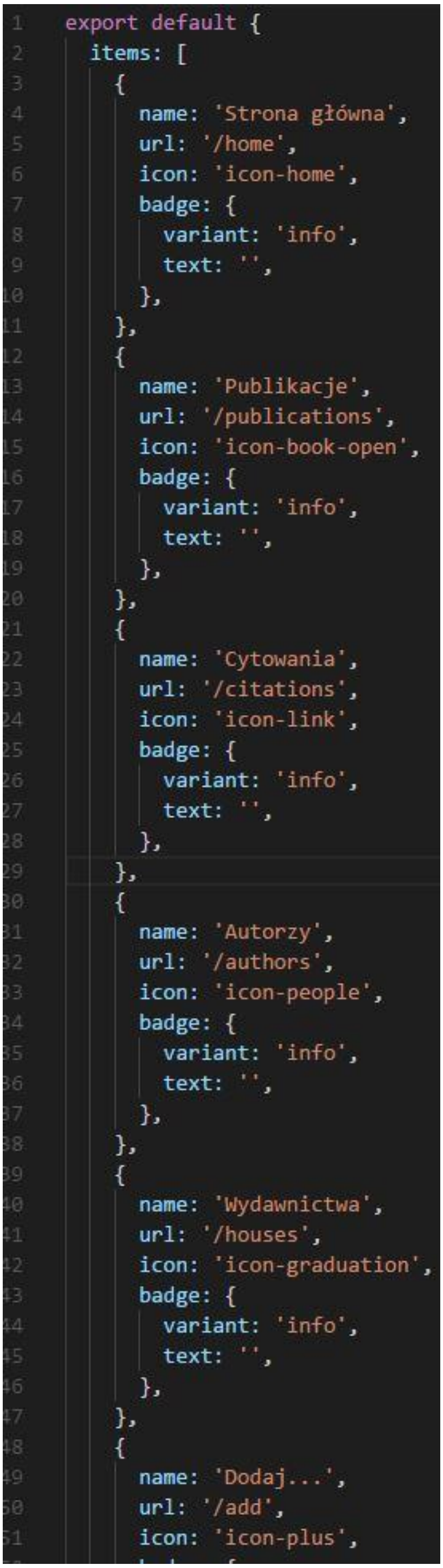
```
const Full = () => (
  <div className="app">
    <Header />
    <div className="app-body">
      <Sidebar />
      <main className="main">
        <Breadcrumb />
        <Container fluid>
          <Switch>
            <Route path="/home" name="Strona główna" component={Home} />
            <Route path="/publications" name="Publikacje" component={Publications} />
            <Route path="/citations" name="Cytowania" component={Citations} />
            <Route path="/authors" name="Autorzy" component={Authors} />
            <Route path="/houses" name="Wydawnictwa" component={Houses} />
            <Route path="/add" name="Dodaj..." component={Add} />
            <Redirect from="/" to="/home" />
          </Switch>
        </Container>
      </main>
      <Aside />
    </div>
    <Footer />
  </div>
);

export default Full;
```

b) Route



```
const routes = {
  '/': 'Strona główna',
  '/login': 'Login',
  '/home': 'Strona główna',
  '/publications': 'Publikacje',
  '/citations': 'Cytowania',
  '/authors': 'Autorzy',
  '/houses': 'Wydawnictwa',
  '/add': 'Dodaj...',
};
export default routes;
```

c) Nawigacja w panelu bocznym (menu)



```
export default {
  items: [
    {
      name: 'Strona główna',
      url: '/home',
      icon: 'icon-home',
      badge: {
        variant: 'info',
        text: '',
      },
    },
    {
      name: 'Publikacje',
      url: '/publications',
      icon: 'icon-book-open',
      badge: {
        variant: 'info',
        text: '',
      },
    },
    {
      name: 'Cytowania',
      url: '/citations',
      icon: 'icon-link',
      badge: {
        variant: 'info',
        text: '',
      },
    },
    {
      name: 'Autorzy',
      url: '/authors',
      icon: 'icon-people',
      badge: {
        variant: 'info',
        text: '',
      },
    },
    {
      name: 'Wydawnictwa',
      url: '/houses',
      icon: 'icon-graduation',
      badge: {
        variant: 'info',
        text: '',
      },
    },
    {
      name: 'Dodaj...',
      url: '/add',
      icon: 'icon-plus',
```