TUGAS PERORANGAN

MODUL 13: FIREBASE REALTIME DATABASE

Disusun sebagai

MATA KULIAH: PBF

Oleh:

Rois Dwi Admaja/1741720193

TI – 3B / 23

PROGRAM STUDI: D-IV TEKNIK INFORMATIKA

JURUSAN: TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2020

1. Mennghubungkan reactjs ke firebase dengan menggunakan firebase config

src > firebase > config.js > ...

```
const firebaseConfig = {
    apiKey: "AIzaSyCsFkorNs1_GJOIYRr0zzpAx8GZowxtYH8",
    authDomain: "react-login-fab26.firebaseapp.com",
    databaseURL: "https://react-login-fab26.firebaseio.com",
    projectId: "react-login-fab26",
    storageBucket: "react-login-fab26.appspot.com",
    messagingSenderId: "506362191682",
    appId: "1:506362191682:web:1a4e23451f9ca5bf09f2db",
    measurementId: "G-MVQPVSD6K1"
  };

  export default firebaseConfig;
```

2. Lalu melakukan modifikasi pada statefull komponen BlogPost.jsx agar bisa melakukan CURD ke API firebase

src > container > BlogPost > BlogPost.jsx > BlogPost > handleTombolSimpan

```
import React, {Component} from "react";
import './BlogPost.css';
import Post from "../../component/BlogPost/Post";
// import API from "../../services";
import firebase from "firebase";
import firebaseConfig from "../../firebase/config.js";

class BlogPost extends Component{

    constructor(props) {
        super(props);
        firebase.initializeApp(firebaseConfig);

        this.state = {
            listArtikel: []
        }
    }
```

src > container > BlogPost > BlogPost.jsx > BlogPost > handleTombolSimpan

```
ambilDataDariServerAPI = () => {              // fungsi untuk mengambil data dari API dengan penambahan sort dan orde
  let ref = firebase.database().ref("/");
  ref.on("value", snapshot =>{
      const state = snapshot.val();
      this.setState(state);
  })
}

simpanDataKeServerAPI = () =>{ //fungsi untuk mengirim insert data ke API realtime database
    firebase.database()
    .ref("/")
    .set(this.state);
}

componentDidMount() {       // komponen untuk mengecek ketika compnent telah di-mount-ing, maka panggil API
    this.ambilDataDariServerAPI()  // ambil data dari server API lokal
}

componentDidUpdate(prevProps, prevState) {
    if (prevState !== this.state) {
        this.simpanDataKeServerAPI();
    }
}
```

```
    handleTombolSimpan = (event) => {            // fungsi untuk meng-handle tombol simpan
        let title = this.refs.judulArtikel.value;
        let body = this.refs.isiArtikel.value;
        let uid = this.refs.uid.value;

        if (uid && title && body) {
            const {listArtikel} = this.state;
            const indeksArtikel = listArtikel.findIndex(data => {
                return data.uid === uid;
            });
            listArtikel[indeksArtikel].title = title;
            listArtikel[indeksArtikel].body = body;
            this.setState({listArtikel});
        }else if (title && body) {
            const uid = new Date().getTime().toString();
            const {listArtikel} = this.state;
            listArtikel.push({uid, title, body});
            this.setState({listArtikel});
        }
        this.refs.judulArtikel.value = "";
        this.refs.isiArtikel.value = "";
        this.refs.uid.value = "";
    };
```

```
    render() {
        return(
            <div className="post-artikel">
                <div className="form pb-2 border-bottom">
                    <div className="form-group row">
                        <label htmlFor="title" className="col-sm-2 col-form-label">Judul</label>
                        <div className="col-sm-10">
                            <input type="text" className="form-control" id="title" name="title" ref="judulArtikel"/>
                        </div>
                    </div>
                    <div className="form-group row">
                        <label htmlFor="body" className="col-sm-2 col-form-label">Isi</label>
                        <div className="col-sm-10">
                            <textarea className="form-control" id="body" name="body" rows="3" ref="isiArtikel"></textarea>
                        </div>
                    </div>
                    <input type="hidden" name="uid" ref="uid"/>
                    <button type="submit" className="btn btn-primary" onClick={this.handleTombolSimpan}>Simpan</button>
                </div>
                <h2>Daftar Artikel</h2>
                {
                    this.state.listArtikel.map(artikel => {  // Looping dan masukkan untuk setiap data yang ada di listArti
                        return <Post key={artikel.uid} judul={artikel.title}
                        isi={artikel.body} idArtikel={artikel.uid}
                        hapusArtikel={this.handleHapusArtikel}/>     // mappingkan data json dari API sesuai dengan kategor
                    })
                }
            </div>
        )
    }
}

export default BlogPost;
```

3. Melakukan perubahan pada komponen post bagian tombol hapus, pada tombol hapus terdapat kondisi untuk memunculkan jendela konfirmasi

src > component > BlogPost > Post.jsx > Post

```jsx
import React from "react";

const Post = (props) => {
    return (
        <div className="artikel">
            <div className="gambar-artikel">
                <img src="http://placeimg.com/80/80/tech" alt="Gambar Tumbnail Artikel"/>
            </div>
            <div className="konten-artikel">
                <div className="judul-artikel">{props.judul}</div>
                <p className="isi-artikel">{props.isi}</p>
                <button className="btn btn-sm btn-warning"
                    onClick={() => {if (window.confirm('Apakah anda yakin menghapus artikel ini?'))
                    props.hapusArtikel(props.idArtikel)}}>
                    Hapus
                </button>
            </div>
        </div>
    )
}

export default Post;
```

4. Pertanyaan Praktikum

a. Perhatikan file BlogPost.jsx, apa saja kode program yang berubah dari BlogPost.js pada Modul-8 dengan BlogPost.jsx pada praktikum kali ini? Kenapa?

Banyak perubahan dari file BlogPost.jsx ini terutama pada state listArtikel yang di masukkan ke dalam constructor

```jsx
    constructor(props) {
        super(props);
        firebase.initializeApp(firebaseConfig);

        this.state = {
            listArtikel: []
        }
    }
```

lalu ambil data server yang semula mengambil data dari fake API diganti menjadi firebase

```jsx
    ambilDataDariServerAPI = () => {
      let ref = firebase.database().ref("/");
      ref.on("value", snapshot =>{
          const state = snapshot.val();
          this.setState(state);
      })
    }
```

Lalu pada handleHapusArtikel dan handleTombolSimpan juga berubah

```
43      handleHapusArtikel = (idArtikel) => {        // fungsi yang meng-handle button action hapus data
44          const {listArtikel} = this.state;
45          const newState = listArtikel.filter(data => {
46              return data.uid !== idArtikel;
47          });
48          this.setState({listArtikel: newState});
49      }
50
```

src > container > BlogPost > BlogPost.jsx > BlogPost > render > state.listArtikel.map() callback

```
51      handleTombolSimpan = (event) => {            // fungsi untuk meng-handle tombol simpan
52          let title = this.refs.judulArtikel.value;
53          let body = this.refs.isiArtikel.value;
54          let uid = this.refs.uid.value;
55
56          if (uid && title && body) {
57              const {listArtikel} = this.state;
58              const indeksArtikel = listArtikel.findIndex(data => {
59                  return data.uid === uid;
60              });
61              listArtikel[indeksArtikel].title = title;
62              listArtikel[indeksArtikel].body = body;
63              this.setState({listArtikel});
64          }else if (title && body) {
65              const uid = new Date().getTime().toString();
66              const {listArtikel} = this.state;
67              listArtikel.push({uid, title, body});
68              this.setState({listArtikel});
69          }
70          this.refs.judulArtikel.value = "";
71          this.refs.isiArtikel.value = "";
72          this.refs.uid.value = "";
73      };
```

Selain perubahan pada fungsi diatas juga terdapat penambahan fungsi yaitu simpanDataKeServerAPI

```
27      simpanDataKeServerAPI = () =>{ //fungsi untuk mengirim insert data ke API realtime database
28          firebase.database()
29          .ref("/")
30          .set(this.state);
31      }
```

Dan juga terdapat penambahan componentDidUpdate

```
37      componentDidUpdate(prevProps, prevState) {
38          if (prevState !== this.state) {
39              this.simpanDataKeServerAPI();
40          }
41      }
42
```

Pada form input juga terjadi perubahan yaitu fungsi onChange diganti menjadi ref.

```
75      render() {
76          return(
77              <div className="post-artikel">
78                  <div className="form pb-2 border-bottom">
79                      <div className="form-group row">
80                          <label htmlFor="title" className="col-sm-2 col-form-label">Judul</label>
81                          <div className="col-sm-10">
82                              <input type="text" className="form-control" id="title" name="title" ref="judulArtikel"/>
83                          </div>
84                      </div>
85                      <div className="form-group row">
86                          <label htmlFor="body" className="col-sm-2 col-form-label">Isi</label>
87                          <div className="col-sm-10">
88                              <textarea className="form-control" id="body" name="body" rows="3" ref="isiArtikel"></textarea>
89                          </div>
90                      </div>
91                      <input type="hidden" name="uid" ref="uid"/>
92                      <button type="submit" className="btn btn-primary" onClick={this.handleTombolSimpan}>Simpan</button>
93                  </div>
```

Dan pada state untuk menampilkan artikel , id artikel diganti menjadi uid

```
96                      this.state.listArtikel.map(artikel => {  // looping dan masukkan untuk setiap data yang ada di listArti
97                          return <Post key={artikel.uid} judul={artikel.title}
98                          isi={artikel.body} idArtikel={artikel.uid}
99                          hapusArtikel={this.handleHapusArtikel}/>     // mappingkan data json dari API sesuai dengan kategor
100                     })
101                 }
```

b. Perhatikan file Post.jsx, apa saja kode program yang berubah dari Post.js pada Modul-8 dengan Post.jsx pada praktikum kali ini? Kenapa?

Perubahan yang terjadi pada post.jsx hanya pada button hapus saja dimana terdapat fungsi untuk melakukan konfirmasi apakah user mau menghapus artikel

src > component > BlogPost > Post.jsx > Post

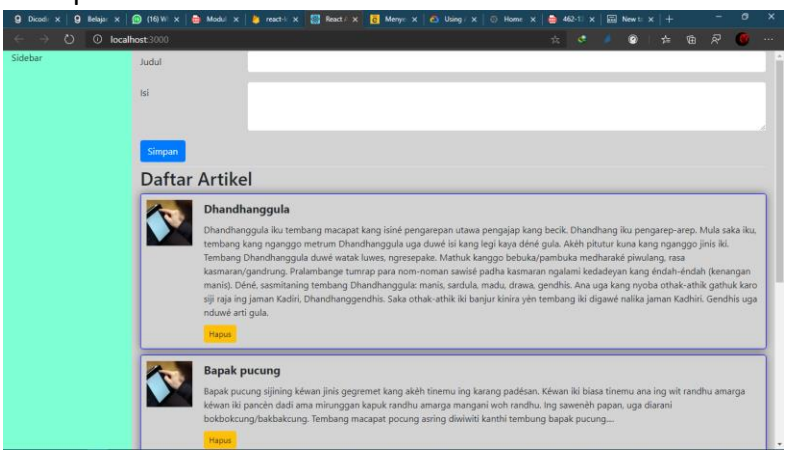
```jsx
import React from "react";

const Post = (props) => {
    return (
        <div className="artikel">
            <div className="gambar-artikel">
                <img src="http://placeimg.com/80/80/tech" alt="Gambar Tumbnail Artikel"/>
            </div>
            <div className="konten-artikel">
                <div className="judul-artikel">{props.judul}</div>
                <p className="isi-artikel">{props.isi}</p>
                <button className="btn btn-sm btn-warning"
                    onClick={() => {if (window.confirm('Apakah anda yakin menghapus artikel ini?'))
                    props.hapusArtikel(props.idArtikel)}}>
                    Hapus
                </button>
            </div>
        </div>
    )
}

export default Post;
```

c. Apakah Global API service yang kita buat pada Modul-8 kemarin kita pakai lagi pada praktikum kali ini? Kenapa alasannya?

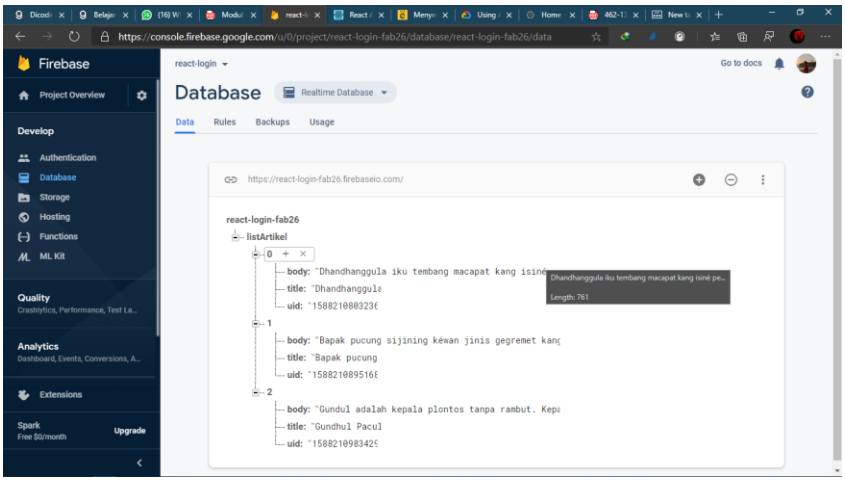
tidak digunakan lagi , karena pada praktikum ini menggunakan database firebase

d. Data yang kita insert bertambah, dan saat kita refresh browser, data masih tetap ada. Dimanakah data-data artikel tersebut disimpan? Tunjukkan hasil screenshot data disimpan tersebut.

Hasil pada halaman react



Hasil tersebut disimpan pada database firebase



e. Menurut kalian lebih mudah dan lebih praktis mana aplikasi reactjs menggunakan data API sendiri (seperti Modul-4 dan Modul-8) atau menggunakan Firebase realtime database? Berika alasannya.

Untuk pembuatannya lebih mudah menggunakan FakeAPI sendiri karena hanya melakukan sedikit konfigurasi. Tetapi kelebihan dari firebase adalah memiliki database yang realtime sehingga semua preangkat yang terhubung akan menerima update dalam waktu milidetik.