LAPORAN PRAKTIKUM MATA KULIAH
PEMROGRAMAN BERBASIS FRAMEWORK

Nama : Alenovan Wiradhika Putra

Kelas : TI-3C D-IV T.INFORMATIKA

NIM : 1741720065

No. Absen : 02

PROGRAM STUDI D-IV TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2020

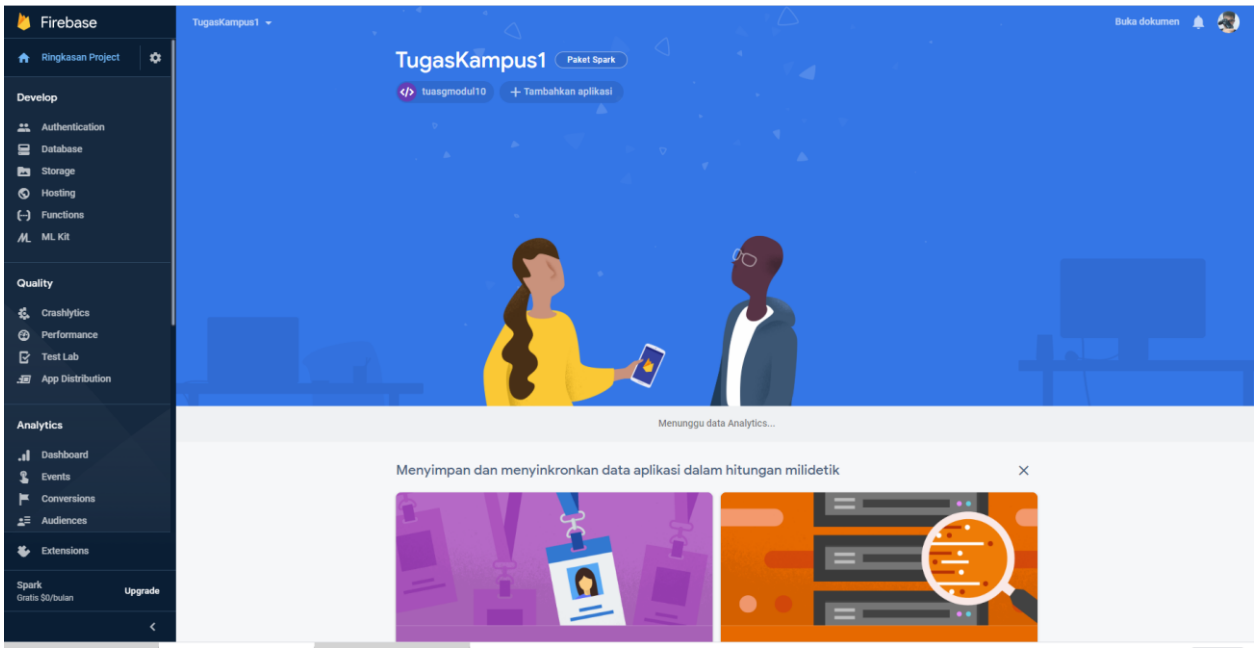
1.1 Langkah Persiapan

Dalam pembuatan Create method pada Firebase, maka perlu database yang akan digunakan untuk menyimpan hasil insert data yang kita masukkan. Langkah-langkah pada persiapan praktikum ini adalah

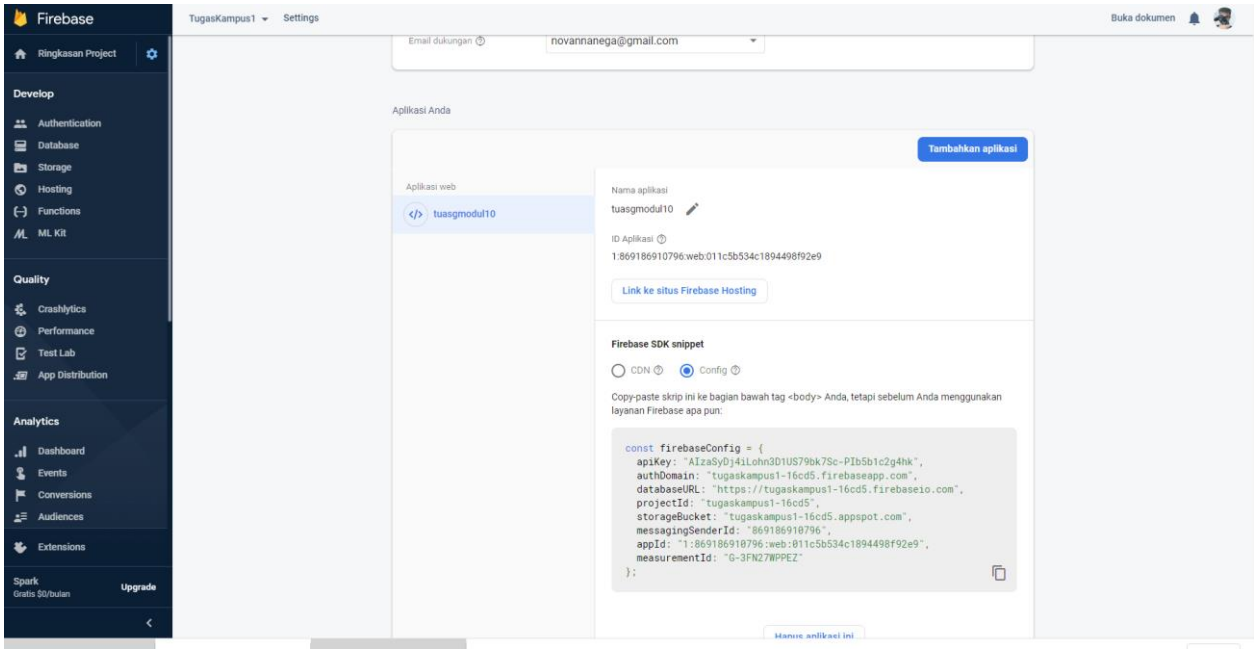
1. Buat project `reactjs` baru, atau pakai project `reactjs` yang sudah ada.
2. Backup folder `public` dan `src`, kemudian hapus semua file dan folder yang ada di dalam folder

   `public` dan `src`.

3. Ekstrak file `Sorce Modul 8.rar` ke dalam project reactjs.
4. Silahkan install beberapa *package* yang dibutuhkan melalui CMD, seperti
   a. `npm install react-router-dom`
   b. `npm install --save firebase`
   c. `npm install bootstrap`
5. Pastikan anda tidak lupa konfigurasi dari aplikasi firebase yang telah dibuat. Jika lupa bisa masuk project firebase konsol di https://console.firebase.google.com
6. Kemudian klik "`ringkasan project`", dan pilih app yang sudah dibuat. Sesuai dengan materi sebelumnya adalah app "`firebase-login-app`", kemudian klik setting.



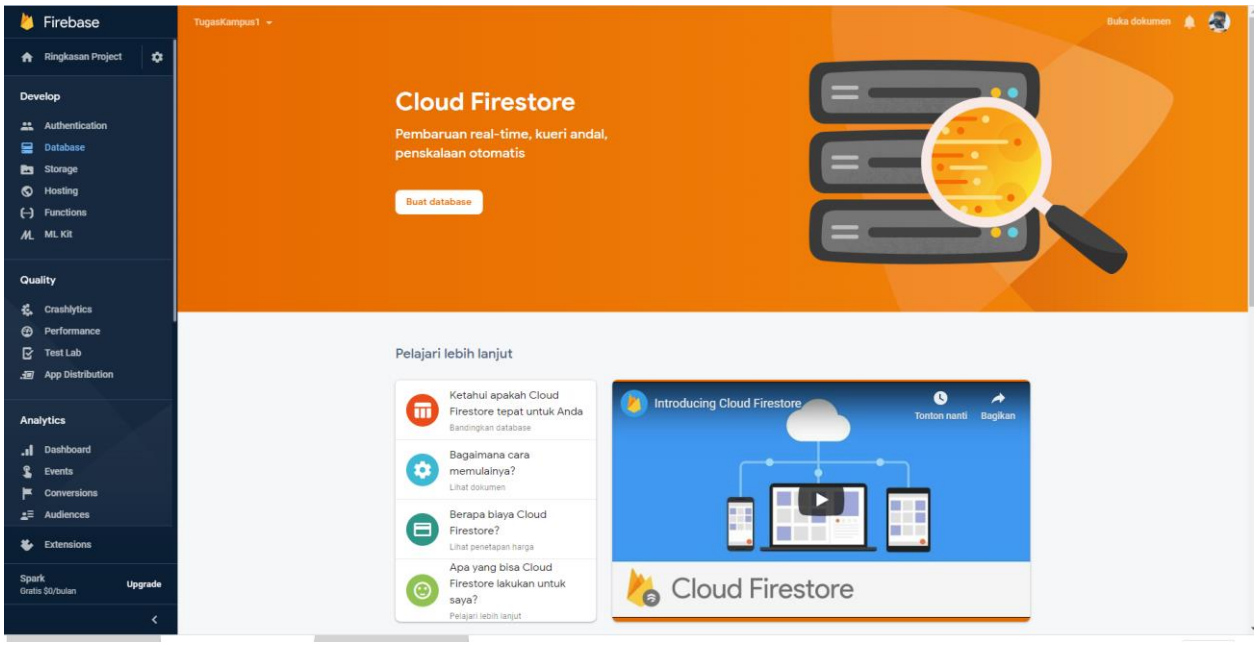
7. Pada tab `umum/common` di menu `Setting`, scroll ke bawah, dan pilih opsi `config` pada `Firebase SDK snippet`.



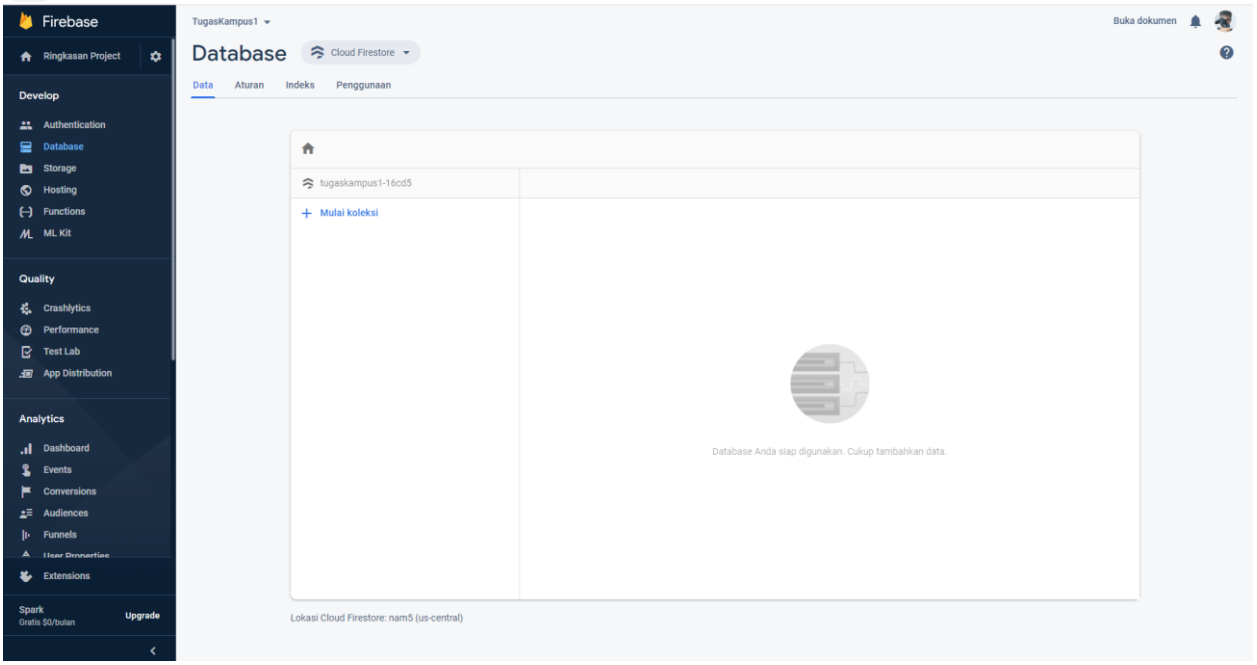
8. Konfigurasi inilah yang akan terus kita gunakan saat berinteraksi dengan API Firebase yang telah kita buat.

```
const firebaseConfig = {
  apiKey: "AIzaSyDj4iLohn3D1US79bk7Sc-PIb5b1c2g4hk",
  authDomain: "tugaskampus1-16cd5.firebaseapp.com",
  databaseURL: "https://tugaskampus1-16cd5.firebaseio.com",
  projectId: "tugaskampus1-16cd5",
  storageBucket: "tugaskampus1-16cd5.appspot.com",
  messagingSenderId: "869186910796",
  appId: "1:869186910796:web:011c5b534c1894498f92e9",
  measurementId: "G-3FN27WPPEZ"
};
```

9. Masuk pada menu `database` pada firebase. Pada materi kali ini kita menggunakan `real time database firebase` jadi pilih menu `Real time database.`



10. Seperti Gambar berikut tampilan database real time firebase yang akan kita gunakan



11. Langkah Persiapan telah selesai. Kita mulai praktikumnya.

1.2 Langkah Praktikum

Praktikum ini sedikit membahas antara praktikum Modul 8 (Global API) dengan Modul 10 dan 11 tentang Firebase. Langkah-langkah praktikum yang dikerjakan adalah

12. Pada project ReactJS kalian, buat file dan folder firebase/config.js dalam folder src dan simpan konfigurasi code program di atas (seperti Langkah 8) pada file config.js.

```
const firebaseConfig = {

  apiKey: "AIzaSyDj4iLohn3D1US79bk7Sc-PIb5b1c2g4hk",

  authDomain: "tugaskampus1-16cd5.firebaseapp.com",
  databaseURL: "https://tugaskampus1-16cd5.firebaseio.com",
  projectId: "tugaskampus1-16cd5",
  storageBucket: "tugaskampus1-16cd5.appspot.com",
  messagingSenderId: "869186910796",
  appId: "1:869186910796:web:011c5b534c1894498f92e9",
  measurementId: "G-3FN27WPPEZ"
};

export default firebaseConfig;
```

12. Kita modifikasi *statefull component* BlogPost.jsx yang dulu pernah kita kerjakan, menjadi seperti ini

```
      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/firebase";

class BlogPost extends Component{
    constructor(props){
        super(props);
        firebase.initializeApp(firebaseConfig);
        this.state = {
            listArtikel: []
        }
    }
```

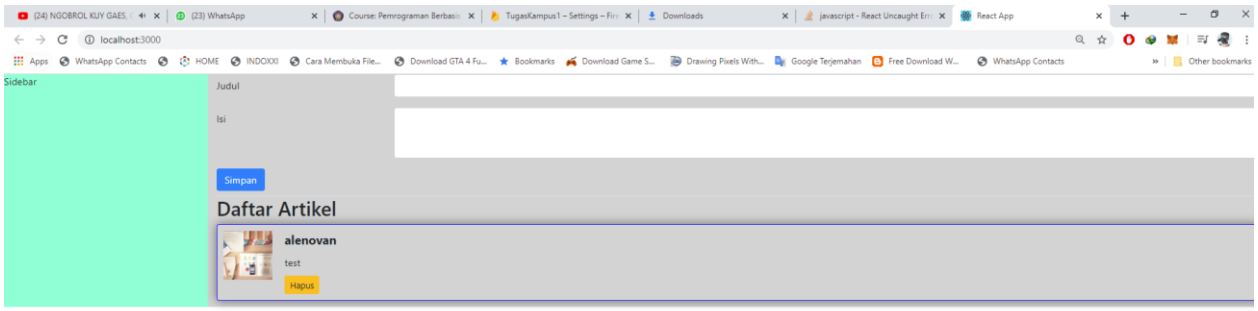
```
    ambilDataDariServerAPI = () => {                // fungsi untuk mengambil dat
a dari API dengan penambahan sort dan order
        let ref = firebase.database().ref("/");
        ref.on("value", snapshot => {
            const state = snapshot.val();
            this.setState(state);
        });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

    simpanDataKeServerAPI = () => {
        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();
        }
    }

    handleHapusArtikel = (idArtikel) => {        // fungsi yang meng-
handle button action hapus data
        const {listArtikel} = this.state;
        const newState = listArtikel.filter(data => {
            return data.uid !== idArtikel;
        });
        this.setState({listArtikel: newState});
    }

    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 masuk
kan untuk setiap data yang ada di listArtikel ke variabel artikel
```

```
                        return <Post key={artikel.uid} judul={artikel.title} isi=
{artikel.body} idArtikel={artikel.uid} hapusArtikel={this.handleHapusArtikel}/>
   // mappingkan data json dari API sesuai dengan kategorinya
                    })
                }
            </div>
        )
    }
}

export default BlogPost;
```

12. Edit pula *stateless component* **Post.jsx** menjadi seperti gambar berikut

```
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 Ar
tikel"/>
            </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 in
i??'))
                        props.hapusArtikel(props.idArtikel)
                    }}>
                    Hapus
                </button>
            </div>
        </div>
    )
}

export default Post;
```



12. Silahkan lakukan proses insert data pada browser, lihat apa yang terjadi

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?

Pada modul 8 media penyimpanan yang digunakan yaitu fake api , sedangkan di modul kali ini , penyimpanan data di sambungkan dengan firebase database

```
➔ ambilDataDariServerAPI
➔ simpanDataKeServerAPI
➔ handleTombolSimpan
➔ handleHapusArtikel
```

Script lengkapnya

```
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/firebase";

class BlogPost extends Component{
```

```
    constructor(props){
        super(props);
        firebase.initializeApp(firebaseConfig);
        this.state = {
            listArtikel: []
        }
    }

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

    simpanDataKeServerAPI = () => {
        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();
        }
    }

    handleHapusArtikel = (idArtikel) => {        // fungsi yang meng-
handle button action hapus data
        const {listArtikel} = this.state;
        const newState = listArtikel.filter(data => {
            return data.uid !== idArtikel;
        });
        this.setState({listArtikel: newState});
    }

    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 masuk
kan untuk setiap data yang ada di listArtikel ke variabel artikel
                        return <Post key={artikel.uid} judul={artikel.title} isi=
{artikel.body} idArtikel={artikel.uid} hapusArtikel={this.handleHapusArtikel}/>
   // mappingkan data json dari API sesuai dengan kategorinya
                    })
                }
            </div>
        )
    }
}

export default BlogPost;
```

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?

Di post kali ini hanya di tambahkan alert untuk hapus pada post

```
      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 Ar
tikel"/>
            </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 in
i??'))
                        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?

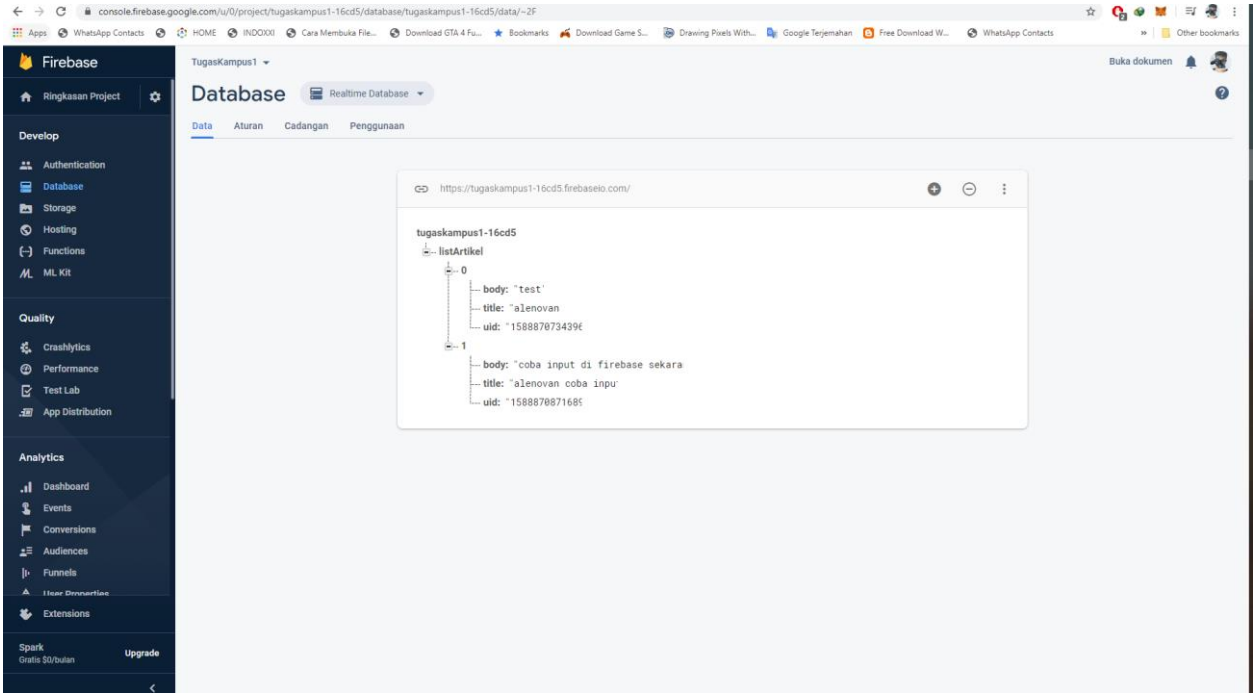
- **Jawab**

Tidak , Fungsi langusng di gabungkan di blogpost.jsx semua

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.

- **Jawab**

Data telah tersimpan di pernyimpanan firebase yang telah kita sambungkan dengan key di atas



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.

- **Jawab**

Tergantung permintaan dan kebutuhan , biasanya ada project yang membutuhkan api sendiri dan di simpan di db / server mereka sendiri (hosting pribadi)

Tetapi

Afa juga yang menggunakan firebase sebagai media penyimpanan , selain cepat juga banyak fitur" di berikan

Tetapi menurutsaya tergantung kebutuhan dan permintaan dalam pembuatan aplikasi dari awal antara team dan client