

TUGAS INDIVIDU

Jobsheet 12

Firestore Realtime Database

Pemrograman Berbasis Framework



Oleh :

Dzikri Alif Abdillah (1741720052)

TI – 3A

PROGRAM STUDI – D4 TEKNIK INFORMATIKA

JURUSAN – TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

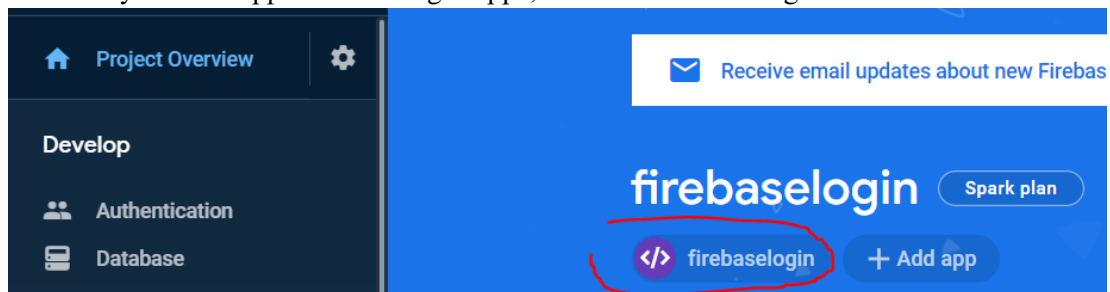
2020/2021

1.1 Langkah persiapan

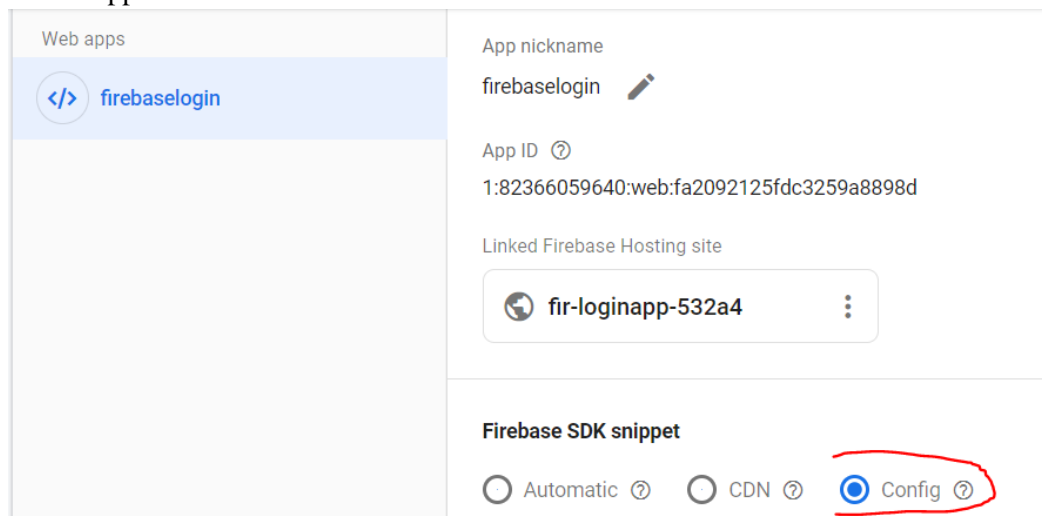
1. Membuat project

```
HP@DESKTOP-DZ MINGW64 /d/DZ/KULIAH/2019-2020/Semester 6/Pemrograman Framework/belajar-react
$ npx create-react-app pertemuan-12
```

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. X



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.

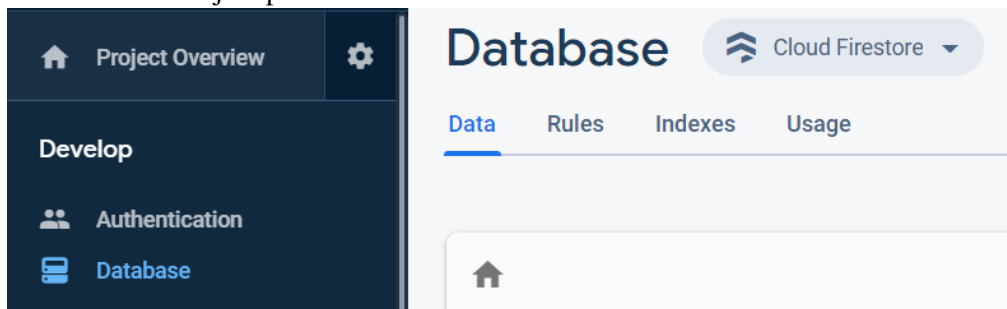
Firebase SDK snippet

☐ Automatic ☐ CDN ☒ Config

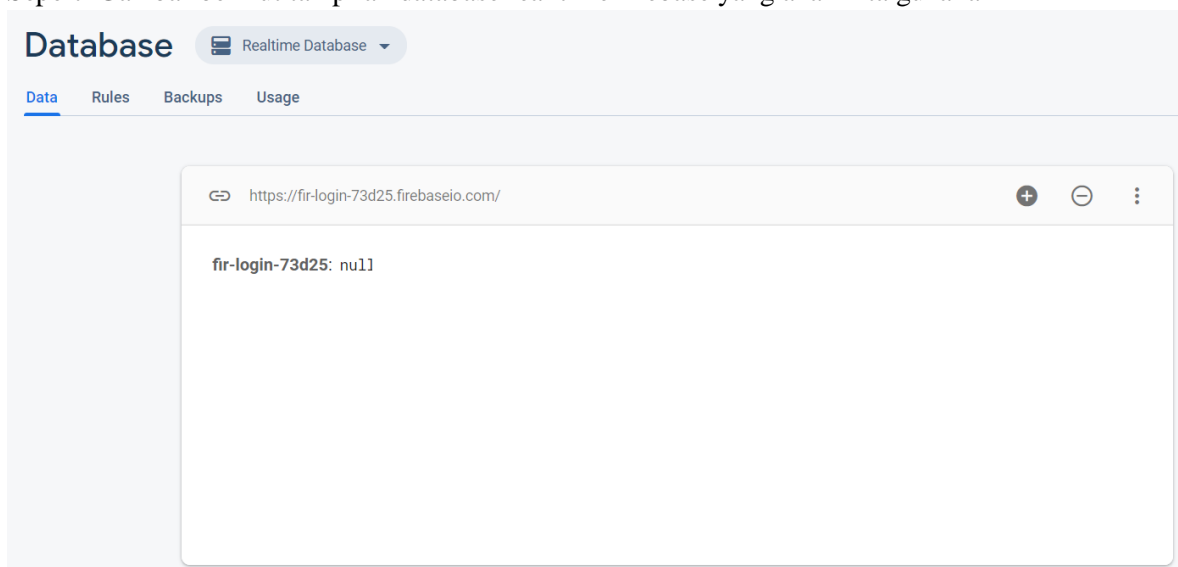
Copy and paste these scripts into the bottom of your <body> tag, but before you use any Firebase services:

```
const firebaseConfig = {
  apiKey: "AIzaSyDwGUcXEyUaqn-KdcUn7XcJbSyJ6RwgcE",
  authDomain: "fir-login-73d25.firebaseio.com",
  databaseURL: "https://fir-login-73d25.firebaseio.com",
  projectId: "fir-login-73d25",
  storageBucket: "fir-login-73d25.appspot.com",
  messagingSenderId: "82366059640",
  appId: "1:82366059640:web:fa2092125fdc3259a8898d",
  measurementId: "G-ZN1EBZFR8F"
};
```

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



1.2 Langkah praktikum

1. 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.

```
firebase > JS Config.js > default
const firebaseConfig = {
  apiKey: "AIzaSyDwGUcXEyUaqqn-KdcUn7XcJbSyJ6RwgcE",
  authDomain: "fir-login-73d25.firebaseio.com",
  databaseURL: "https://fir-login-73d25.firebaseio.com",
  projectId: "fir-login-73d25",
  storageBucket: "fir-login-73d25.appspot.com",
  messagingSenderId: "82366059640",
  appId: "1:82366059640:web:fa2092125fdc3259a8898d",
  measurementId: "G-ZN1EBZFR8F"
};
export default firebaseConfig;
```

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

```
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";

class BlogPost extends Component{
  constructor(props){
    super(props);
    firebase.initializeApp(firebaseConfig);

    this.state = {
      listArtikel: []
    }
  }

  ambilDataDariServerAPI = () => {
    // fungsi untuk mengambil data 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();
    }
  }
}
```

```

container > BlogPost > BlogPost.jsx > BlogPost > handleTombolSimpan

    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) { // cek apakah tdk null
        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) { // jika data blm ada diserver
        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 = "";
    }
  }
}

```

```

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

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 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;

```

3. Edit pula stateless component Post.jsx menjadi seperti gambar berikut

```
component > BlogPost > Post.jsx > Post
import React from "react";

const Post = (props) => {
  return (
    <div className="artikel">
      <div className="gambar-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={() => props.hapusArtikel(props.idArtikel)}>Hapus</button>
        <button onClick={() => {if (window.confirm('Apakah anda yakin menghapus artikel ini?')) props.hapusArtikel(props.idArtikel)}}></button>
      </div>
    </div>
  )
}

export default Post;
```

4. Silahkan lakukan proses insert data pada browser, lihat apa yang terjadi
Output browser ketika menambah data

← → ↺ ⓘ localhost:3000

☆ 🔍 📷 📁 📄 📱 📺 ⋮

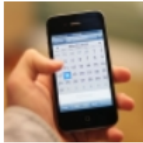
Sidebar

Judul

Isi

Simpan

Daftar Artikel



firebase crud

dzikri test

Hapus

1.3 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?
 - Kode program yang berubah yaitu fungsi ambilDataDariServerAPI, simpanDataKeServerAPI, componentDidUpdate, handleHapusArtikel, HandleTombolSimpan dan render HTML. Serta penambahan constructor
 - Di beberapa fungsi untuk menyambungkan program dengan database firebase, sedangkan di fungsi render menambahkan atribut **ref** untuk mempassing data dari input ke fungsi simpanDataKeServerAPI
- 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?
 - Yang berubah hanya button dengan penambahan popup konfirmasi penghapusan data
- c. Apakah Global API service yang kita buat pada Modul-8 kemarin kita pakai lagi pada praktikum kali ini? Kenapa alasannya?
 - Tidak, karena pada project ini kita terhubung ke firebase tidak dengan API buatan kita
- 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.
 - Data disimpan didalam 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
 - Lebih mudah dan praktis dengan firebase, karena jika menggunakan API sendiri akan memberikan waktu lama