

**LAPORAN HASIL PRAKTIKUM
PEMROGRAMAN WEB DAN MOBILE I**



NAMA : M.FAHRIYAN MAHMUDI

NIM : 193020503018

KELAS : A

MODUL : V (React Native)

**JURUSAN/PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PALANGKARAYA
2021**

BAB I

LANDASAN TEORI

A. Landasan Teori

a. React Native

React Native adalah *framework mobile app development* yang memungkinkan pengembangan aplikasi secara multi-*platform* yaitu Android dan iOS. Sederhananya, React Native adalah *framework* yang memungkinkan *developer* membuat aplikasi untuk berbagai *platform* dengan menggunakan basis kode yang sama.

Penggabungan antara *native mobile app* dan React juga bisa dilakukan dengan mudah. Jadi, pengembang bisa membuat aplikasi Android dan iOS dengan lebih cepat. React Native sendiri pertama kali dirilis pada tahun 2015 oleh Facebook dan menjadi bagian dari proyek *open source* mereka. Kemudian, hanya dalam beberapa tahun saja, React Native telah menjadi andalan bagi banyak *developer* untuk mengembangkan aplikasi seluler.

Beberapa contoh aplikasi seluler yang menggunakan *framework* React Native adalah Instagram, Facebook, Pinterest, hingga Skype.

Ada beberapa alasan lain mengapa *React Native* begitu populer, antara lain:

1. *Developer* hanya perlu membuat satu kode untuk mengembangkan aplikasi berbasis Android atau iOS. Jadi, *framework* ini benar-benar menghemat waktu.
2. React Native adalah *framework* yang dibangun berdasarkan React, yaitu *library* dari JavaScript yang sudah populer.
3. *Framework* ini mendorong *frontend developer* yang sebelumnya hanya dapat bekerja dengan teknologi berbasis web saja. Kini, mereka bisa dengan mudah untuk mengembangkan *mobile platform*.

Selain beberapa alasan di atas, *React Native* juga memiliki keunggulan lain yaitu tidak akan *re-render webviews* dalam kodenya. Jadi program akan dijalankan sesuai dengan tampilan dan komponen dari *native* yang awalnya digunakan.

b. Cara Kerja React Native

Setelah mengetahui pengertian dan beberapa alasan mengapa *React Native* begitu populer. Kini, coba ketahui dulu bagaimana kerjanya. *React Native* adalah *framework* yang ditulis dengan campuran bahasa JavaScript dan JXL, sebuah kode *markup* khusus yang menyerupai XML. *Framework* ini memiliki kemampuan untuk berkomunikasi dengan kedua ranah sekaligus, yaitu *threads* yang berbasis JavaScript dan *threads* dari *native app*.

Jadi, bagaimana caranya untuk berkomunikasi? Rupanya *React Native* menggunakan apa yang disebut dengan “*bridge*” atau jembatan. JavaScript dan *threads native* memang ditulis dengan dua bahasa pemrograman yang berbeda. Namun, fitur *bridging* dari *React Native* tetap memungkinkan untuk komunikasi dua arah.

Itu artinya saat pengembang sudah memiliki aplikasi Android atau iOS, mereka masih tetap bisa menggunakan komponennya saat menggunakan *React Native*.

c. Flatlist

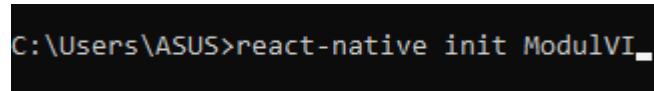
FlatList menampilkan data terstruktur yang serupa dalam daftar yang dapat di-scroll. Ini berfungsi dengan baik untuk daftar data yang besar di mana jumlah item daftar dapat berubah seiring waktu. *FlatList* hanya menampilkan elemen render yang saat ini ditampilkan di layar, tidak semua elemen daftar sekaligus.

Komponen *FlatList* membutuhkan dua props yang diperlukan: data dan *renderItem*. Data adalah sumber elemen untuk daftar, dan *renderItem* mengambil satu item dari sumber dan mengembalikan komponen yang diformat untuk dirender.

BAB II

PEMBAHASAN

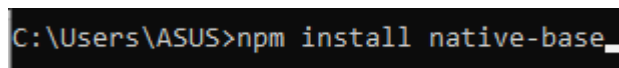
Pada praktikum ini, diminta untuk membuat sebuah program aplikasi mobile, dimana dalam program tersebut terdapat program search menggunakan flatlist react-native. Pertama, buka cmd terlebih dahulu lalu buat projectnya dengan menggunakan perintah pada cmd.



```
C:\Users\ASUS>react-native init ModulVI_
```

Gambar 2.1 Create Project

Tunggu perintah tersebut berjalan hingga selesai. Setelah selesai, maka akan ada folder project baru dengan nama ModulVI. Lalu pada project ModulVI tersebut akan diinstall native-base, caranya dengan menuliskan perintah pada cmd, perintahnya adalah,



```
C:\Users\ASUS>npm install native-base_
```

Gambar 2.2 Installasi

Tunggu hingga selesai. Setelah selesai, buka folder ModulVI tersebut lalu buka file app.js di visual studio code. Hapus semua code yang terdapat dalam file tersebut. Lalu diganti dengan kode yang dibagi menjadi beberapa bagian sebagai berikut, Merupakan bagian import, yang digunakan untuk dapat menggunakan komponen-komponen yang terdapat pada library react dan juga library native-base. Bagian kedua, terdapat deklarasi variabel dengan nama helperArray, dimana variabel tersebut berisi require, require digunakan untuk dapat menghubungkan file app.js tersebut dengan file nama.json. nama.json merupakan file yang berisi daftar nama-nama yang akan digunakan pada program ini.

```

1  import React, {Component} from 'react';
2  import{
3    Container,
4    Header,
5    Content,
6    Left,
7    Right,
8    Body,
9    Icon,
10   Text,
11   ListItem,
12   Item,
13   Input,
14 } from 'native-base';
15

```

Gambar 2.3 import (bagian 1)

```

16  let helperArray = require('./nama.json');
17  export default class App extends Component {
18    constructor(props){
19      super(props);
20      this.state = {
21        allUsers: helperArray,
22        usersFiltered:helperArray,
23      };
24    }
25

```

Gambar 2.4 HelperArray (bagian 2)

```
1  [  
2    {  
3      "name": "Radif",  
4      "address": "Jalan Tembalu 1, pangkalan Bun"  
5    },  
6  
7    {  
8      "name": "Naya",  
9      "address": "Jalan Bamban, Pangkalan Bun"  
10   },  
11  
12   {  
13     "name": "Nara",  
14     "address": "Jalan Tembalu 3, pangkalan Bun"  
15   },  
16  
17   {  
18     "name": "Nabil",  
19     "address": "Jalan Tembalu 5, pangkalan Bun"  
20   },  
21  
22   {  
23     "name": "razzi",  
24     "address": "Jalan Tembalu 2, pangkalan Bun"  
25   },  
26  
27   {  
28     "name": "irfan",  
29     "address": "Jalan Tembalu 6, pangkalan Bun"  
30   },  
31 ]
```

Gambar 2.5 Nama.json Part 1 (bagian 3)

```

32  {
33      "name": "Henry",
34      "address": "Jalan Hasanuddin, pangkalan Bun"
35  },
36
37  {
38      "name": "Dandy",
39      "address": "Jalan Seroja, pangkalan Bun"
40  },
41
42  {
43      "name": "Reza",
44      "address": "Jalan Kampung, pangkalan Bun"
45  },
46
47  {
48      "name": "Ihsan",
49      "address": "Jalan Baru, pangkalan Bun"
50  },
51
52  {
53      "name": "Agus",
54      "address": "Jalan Tenggara, pangkalan Bun"
55  },
56
57  {
58      "name": "Deddy",
59      "address": "Jalan Smantri, pangkalan Bun"
60  },
61

```

Gambar 2.6 Nama.json Part 2 (bagian 3)

```

61
62  {
63      "name": "Messi",
64      "address": "Jalan Mendawai, pangkalan Bun"
65  }
66
67  ]

```

Gambar 2.7 Nama.json Part 3 (bagian 3)

Setelah itu, terdapat kelas dengan nama App, yang berisi, konstruktor, dimana konstruktor tersebut berisi inisiasi atribut dengan nilai sama dengan variabel helperArray. Atribut tersebut akan mengambil data-data dari variabel helperArray.

```
✓ searchUser(textToSearch) {  
✓   this.setState({  
✓     usersFiltered: this.state.allUsers.filter(i =>  
      i.name.toLowerCase().includes(textToSearch.toLowerCase()),  
    ),  
  });  
}
```

Gambar 2.8 searchUser (bagian 4)

Bagian 4 merupakan fungsi yang terdapat dalam kelas App, merupakan fungsi searchUser, yang mana fungsi tersebut akan digunakan untuk mencarinama berdasarkan kata yang diinputkan pada kolom pencarian. Fungsi searchUser ini menggunakan filter, dimana akan menyeleksi sesuai dari kata yang diinputkan. Lalu terdapat toLowerCase() dimana digunakan untuk merubah nilai string menjadi string yang terdiri dari huruf kecil semua.


```

render() {
  return (
    <Container>
      <Header searchBar rounded>
        <Item>
          <Icon name="search"/>
          <Input
            placeholder="Search User"
            onChangeText={text => {
              this.searchUser(text);
            }}
          />
        </Item>
      </Header>
      <Content>
        {this.state.usersFiltered.map((item,index)=>(
          <ListItem>
            <Left></Left>
            <Body>
              <Text>{item.name}</Text>
              <Text>{item.address}</Text>
            </Body>
            <Right />
          </ListItem>
        ))}
      </Content>
    </Container>
  );
}
}

```

Gambar 2.9 Render (bagian 5)

Bagian kelima, merupakan bagian fungsi render, dimana fungsi ini berisi perintah-perintah untuk menampilkan data-data dalam bentuk list. Dimana data-data yang ditampilkan adalah data yang terdapat pada variabel helperArray. Berikut adalah tampilan dari data tersebut serta implementasi dari pencarian kata pada aplikasi tersebut.

BAB III

KESIMPULAN

Dapat ditarik Kesimpulan, bahwa React Native ialah kombinasi antara React dan Javascript yang digunakan untuk membangun aplikasi mobile. Karena kamu tidak perlu repot-repot membuat 2 desain aplikasi untuk 2 platform yang berbeda. Perlu diketahui bahwa React Native bersifat Multi-Platform yakni 1 code yang bisa digunakan diberbagai platform

DAFTAR PUSTAKA

- Andy. (2020). Apa Itu React Native dan Kenapa Cocok Digunakan untuk Perusahaan Anda? <https://qwords.com/blog/apa-itu-react-native/>
- ISMI, TRIAS. 2021. "React Native, Framework Yang Jadi Topik Hangat Di Kalangan Developer." [glints.com.
https://glints.com/id/lowongan/reactnative-adalah/#.YJZafrUza01](https://glints.com/id/lowongan/reactnative-adalah/#.YJZafrUza01)

LAMPIRAN

```
C:\Users\ASUS>react-native init ModulVI_
```

Gambar 2.1 Create Project

```
C:\Users\ASUS>npm install native-base_
```

Gambar 2.2 Installasi

```
1  import React, {Component} from 'react';
2  import{
3    Container,
4    Header,
5    Content,
6    Left,
7    Right,
8    Body,
9    Icon,
10   Text,
11   ListItem,
12   Item,
13   Input,
14 } from 'native-base';
15
```

Gambar 2.3 import (bagian 1)

```
16  let helperArray = require('./nama.json');
17  export default class App extends Component {
18    constructor(props){
19      super(props);
20      this.state = {
21        allUsers: helperArray,
22        usersFiltered:helperArray,
23      };
24    }
25
```

Gambar 2.4 HelperArray (bagian 2)

```
1  [  
2    {  
3      "name": "Radif",  
4      "address": "Jalan Tembalu 1, pangkalan Bun"  
5    },  
6  
7    {  
8      "name": "Naya",  
9      "address": "Jalan Bamban, Pangkalan Bun"  
10   },  
11  
12   {  
13     "name": "Nara",  
14     "address": "Jalan Tembalu 3, pangkalan Bun"  
15   },  
16  
17   {  
18     "name": "Nabil",  
19     "address": "Jalan Tembalu 5, pangkalan Bun"  
20   },  
21  
22   {  
23     "name": "razzi",  
24     "address": "Jalan Tembalu 2, pangkalan Bun"  
25   },  
26  
27   {  
28     "name": "irfan",  
29     "address": "Jalan Tembalu 6, pangkalan Bun"  
30   },  
31  ]
```

Gambar 2.5 Nama.json Part 1 (bagian 3)

```

32  {
33      "name": "Henry",
34      "address": "Jalan Hasanuddin, pangkalan Bun"
35  },
36
37  {
38      "name": "Dandy",
39      "address": "Jalan Seroja, pangkalan Bun"
40  },
41
42  {
43      "name": "Reza",
44      "address": "Jalan Kampung, pangkalan Bun"
45  },
46
47  {
48      "name": "Ihsan",
49      "address": "Jalan Baru, pangkalan Bun"
50  },
51
52  {
53      "name": "Agus",
54      "address": "Jalan Tenggara, pangkalan Bun"
55  },
56
57  {
58      "name": "Deddy",
59      "address": "Jalan Smantri, pangkalan Bun"
60  },
61

```

Gambar 2.6 Nama.json Part 2 (bagian 3)

```

61
62  {
63      "name": "Messi",
64      "address": "Jalan Mendawai, pangkalan Bun"
65  }
66
67  ]

```

Gambar 2.7 Nama.json Part 3 (bagian 3)

```

✓ searchUser(textToSearch) {
✓   this.setState({
✓     usersFiltered: this.state.allUsers.filter(i =>
      i.name.toLowerCase().includes(textToSearch.toLowerCase()),
      ),
    });
  }
}

```

Gambar 2.8 searchUser (bagian 4)

```

render() {
  return (
    <Container>
      <Header searchBar rounded>
        <Item>
          <Icon name="search"/>
          <Input
            placeholder="Search User"
            onChangeText={text => {
              this.searchUser(text);
            }}
          />
        </Item>
      </Header>
      <Content>
        {this.state.usersFiltered.map((item,index)=>(
          <ListItem>
            <Left></Left>
            <Body>
              <Text>{item.name}</Text>
              <Text>{item.address}</Text>
            </Body>
            <Right />
          </ListItem>
        ))}
      </Content>
    </Container>
  );
}
}

```

Gambar 2.9 Render (bagian 5)