

**LAPORAN PRAKTIKUM  
PENGEMBANGAN WEB DAN MOBILE**



**NAMA : DZIKRI AHMADILLAH  
NIM : 193020503039  
KELAS : A  
MODUL : VI**

**JURUSAN TEKNIK INFORMATIKA  
FAKULTAS TEKNIK  
UNIVERSITAS PALANGKARAYA  
2021**

# **BAB I**

## **TUJUAN DAN LANDASAN TEORI**

### **1.1 Tujuan**

1. Mahasiswa mampu membuat search filter pada flat list pada react native

### **1.2 Landasan Teori**

#### **1.2.1 Flat List**

Flat list React native adalah tampilan yang digunakan untuk menyimpan item dalam daftar dan menyediakan fitur penting seperti menggulir secara horizontal dan vertikal. Flat list di react native dirancang untuk menangani kumpulan data besar yang mungkin tidak muat di layar perangkat. Flat list melakukan rendering hanya kepada item yang sedang ditampilkan di layar dan tidak semua item. (Pedamkar 2020)

Flat list merupakan salah satu komponen yang paling banyak digunakan dalam react native. Daftar data berfungsi pada alat peraga utama berikut:

- a) Data, yaitu array data yang berisi item individual yang akan digunakan untuk membuat flat list di react native.
- b) renderItem: yaitu fungsi yang tujuannya adalah untuk mengambil item individual dari array dan mengubahnya menjadi bagian dari flat list react-native.

Flatlist juga menyediakan beberapa fitur sebagai berikut :

- 1) Pull to refresh support.
- 2) Header support.
- 3) Footer support.
- 4) Separator support.
- 5) Scroll loading support.
- 6) ScrollToIndex support.
- 7) Cross-Platform support.
- 8) Configurable callbacks.
- 9) Optional support for horizontal mode

### 1.2.2 NativeBase

NativeBase adalah Library open-source yang membantu untuk memulai dan menjalankan komponen antarmuka pengguna seperti tombol, kartu, bidang masukan, dan banyak lagi.

NativeBase bisa dianggap sebagai kerangka kerja CSS untuk membangun aplikasi seluler dengan React Native. Anda bisa mendapatkan gaya yang cocok langsung dari kotaknya. Yang harus Anda lakukan adalah mengambil dan memanggil komponen dari perpustakaan NativeBase.(NativeBase 2021)

Komponen dasar dari native base sendiri terdiri atas :

- 1) *Header*, menyatakan header dari aplikasi.
- 2) *Left*, komponen dari header, menyatakan item yang akan dirender disisi kiri header.
- 3) *Right*, komponen dari header, menyatakan item yang akan dirender disisi kanan header.
- 4) *Body*, komponen dari header, menyatakan item yang akan dirender dibagian tengah header.
- 5) *Button*, komponen yang digunakan untuk membuat tombol aksi yang interaktif.
- 6) *List*, komponen yang digunakan untuk menampilkan daftar data ke layar. *List* harus memiliki *ListItem* didalamnya agar bisa menampilkan data.

## BAB II

### PEMBAHASAN

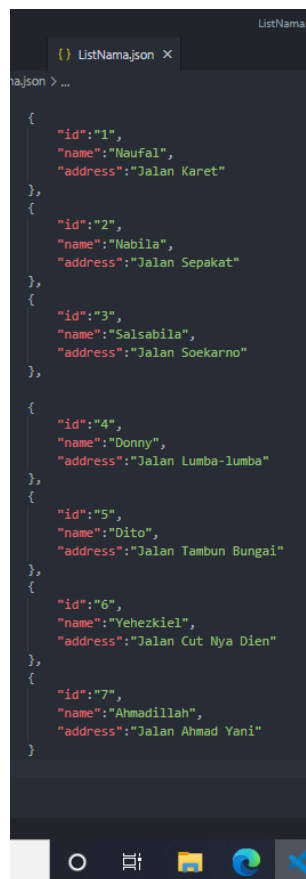
#### 2.1 SOAL

1. Implementasikanlah search filter untuk flat list pada sebuah aplikasi react native!

#### 2.2 JAWABAN

##### 2.2.1 Persiapan Data

Penyelesaian dari soal adalah mengimplementasikan fitur search filter terhadap sebuah flat list. untuk melakukan ini tentunya diperlukan data yang akan ditampilkan. Data yang akan ditampilkan berupa kumpulan nama dan alamat dari pemilik nama tersebut. Data disimpan dalam file berformat json dengan nama file *listNama*.

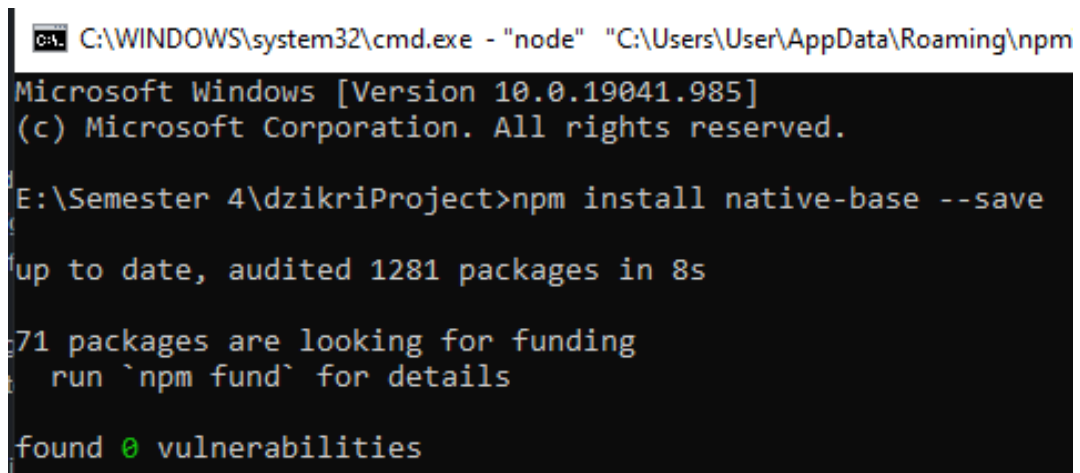


```
{
  "id": "1",
  "name": "Naufal",
  "address": "Jalan Karet"
},
{
  "id": "2",
  "name": "Nabila",
  "address": "Jalan Sepakat"
},
{
  "id": "3",
  "name": "Salsabila",
  "address": "Jalan Soekarno"
},
{
  "id": "4",
  "name": "Donny",
  "address": "Jalan Lumba-lumba"
},
{
  "id": "5",
  "name": "Dito",
  "address": "Jalan Tambun Bungai"
},
{
  "id": "6",
  "name": "Yehezkiel",
  "address": "Jalan Cut Nya Dien"
},
{
  "id": "7",
  "name": "Ahmadillah",
  "address": "Jalan Ahmad Yani"
}
}
```

**Gambar 2.1** Daftar data yang akan ditampilkan ke aplikasi

### 2.2.2 Instalasi NativeBase

Untuk mempercepat proses pembuatan aplikasi, saya menggunakan library open-source NativeBase. Dengan NativeBase, pembuatan tampilan dan list data akan menjadi lebih mudah karena hanya dengan menggunakan tag `<ListItem>`. Untuk melakukan instalasi library ke aplikasi cukup menggunakan command “npm install native-base”. Dilanjutkan dengan menghubungkan library ke project aplikasi menggunakan command “react-native link”.



```
C:\WINDOWS\system32\cmd.exe - "node" "C:\Users\User\AppData\Roaming\npm\node_modules\npm\bin\npm-cli.js"
Microsoft Windows [Version 10.0.19041.985]
(c) Microsoft Corporation. All rights reserved.

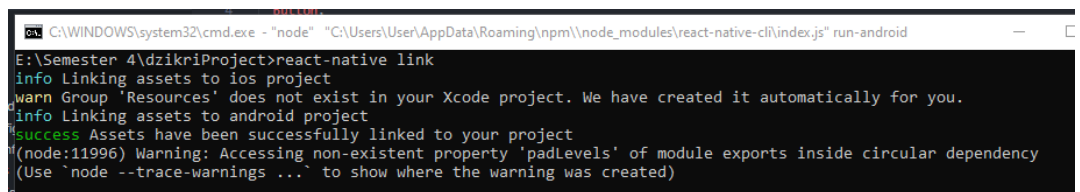
E:\Semester 4\dzikriProject>npm install native-base --save

up to date, audited 1281 packages in 8s

71 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

**Gambar 2.2** Instalasi library NativeBase



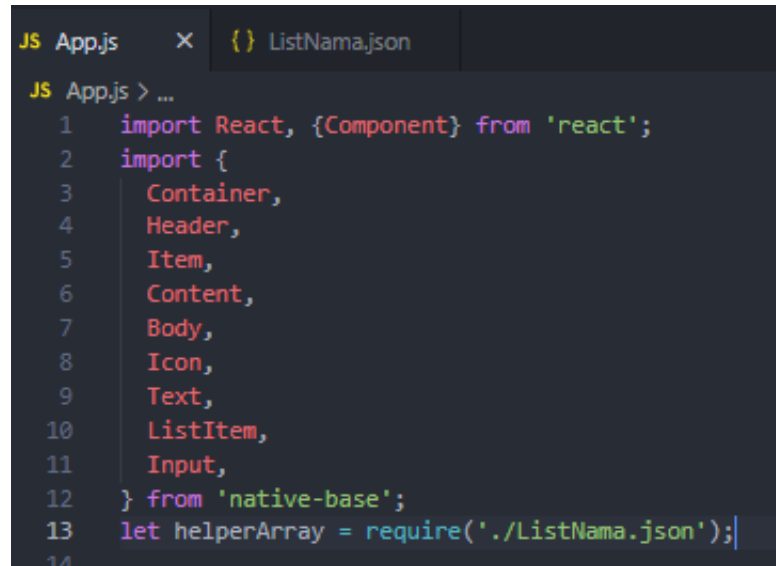
```
C:\WINDOWS\system32\cmd.exe - "node" "C:\Users\User\AppData\Roaming\npm\node_modules\react-native-cli\index.js" run-android
E:\Semester 4\dzikriProject>react-native link
info Linking assets to ios project
warn Group 'Resources' does not exist in your Xcode project. We have created it automatically for you.
info Linking assets to android project
success Assets have been successfully linked to your project
(node:11996) Warning: Accessing non-existent property 'padLevels' of module exports inside circular dependency
(Use `node --trace-warnings ...` to show where the warning was created)
```

**Gambar 2.3** Menghubungkan library ke project

### 2.2.3 Code Aplikasi

Peng-kodean aplikasi akan dilakukan dari file *app.js*. diawal kode diawali dengan melakukan import dari library *react* dan *nativeBase*. Hal ini dilakukan agar nantinya bisa menggunakan sintaks dari library *react* dan *NativeBase*.

Selanjutnya adalah membuat agar *app.js* dapat mengakses data dari *listNama.json*. untuk melakukannya, data dari *listNama.json* akan disimpan ke sebuah array bernama *helperArray*.



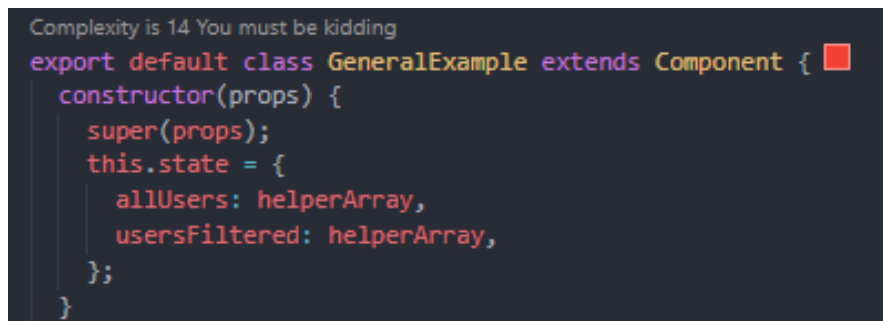
```

JS App.js  X  {} ListNama.json
JS App.js > ...
1  import React, {Component} from 'react';
2  import {
3    Container,
4    Header,
5    Item,
6    Content,
7    Body,
8    Icon,
9    Text,
10   ListItem,
11   Input,
12 } from 'native-base';
13 let helperArray = require('./ListNama.json');
14

```

**Gambar 2.4** Import library dan variabel *helperArray* yang menyimpan data dari *listnama.json*

Selanjutnya adalah membuat sebuah constructor untuk data pada array *helperArray*. Hal ini dilakukan agar isi data tersusun dengan rapi ketika ingin digunakan. Hasil constructor diberi nama *allUser* yang digunakan untuk menampilkan semua data dan *usersFiltered* yang digunakan untuk menampilkan user yang telah diseleksi.



```

Complexity is 14 You must be kidding
export default class GeneralExample extends Component {
  constructor(props) {
    super(props);
    this.state = {
      allUsers: helperArray,
      usersFiltered: helperArray,
    };
  }
}

```

**Gambar 2.5** Membuat constructor untuk menyusun data

Setelah selesai, dilanjutkan dengan membuat sebuah function bernama *searchUser*. Fungsi ini digunakan untuk memilih data mana yang akan ditampilkan sesuai dengan inputan dari user. Digunakan fungsi *toLowerCase()* terhadap acuan

kata dan data nama yang akan dicari. Hal ini dilakukan untuk menyamakan huruf kecil dan besar dari inputan user ke data yang ada.

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

**Gambar 2.6** Function *searchUser* untuk menyeleksi data berdasarkan inputan

Selanjutnya adalah membuat function *render()*. Function inilah nantinya yang akan mengatur bagaimana tampilan dari aplikasi. Function akan me-return, isi dari tag `<container>`. Secara garis besar, isi tag container terbagi menjadi 2, yaitu bagian `<header>` dan bagian `<content>`.

Bagian `<header>` akan digunakan sebagai bar inputan untuk melakukan seleksi user. Dengan menggunakan `<icon>` bernama “search”, program akan mengambil gambar logo pencarian(sebuah kaca pembesar) dari library NativeBase. Pada bagian `<input>`, terdapat syntax *onChangeText*. Syntax tersebut digunakan agar ketika terjadi perubahan pada text atau berarti user melakukan input, maka jalankan function *searchUser*.

```
Complexity is 14 You must be kidding
render() {
  return (
    <Container>
      <Header searchBar rounded>
        <Item>
          <Icon name="search" />
          <Input
            placeholder="Search User"
            onChangeText={text => {
              this.searchUser(text);
            }}
          />
        </Item>
      </Header>
    </Container>
  );
}
```

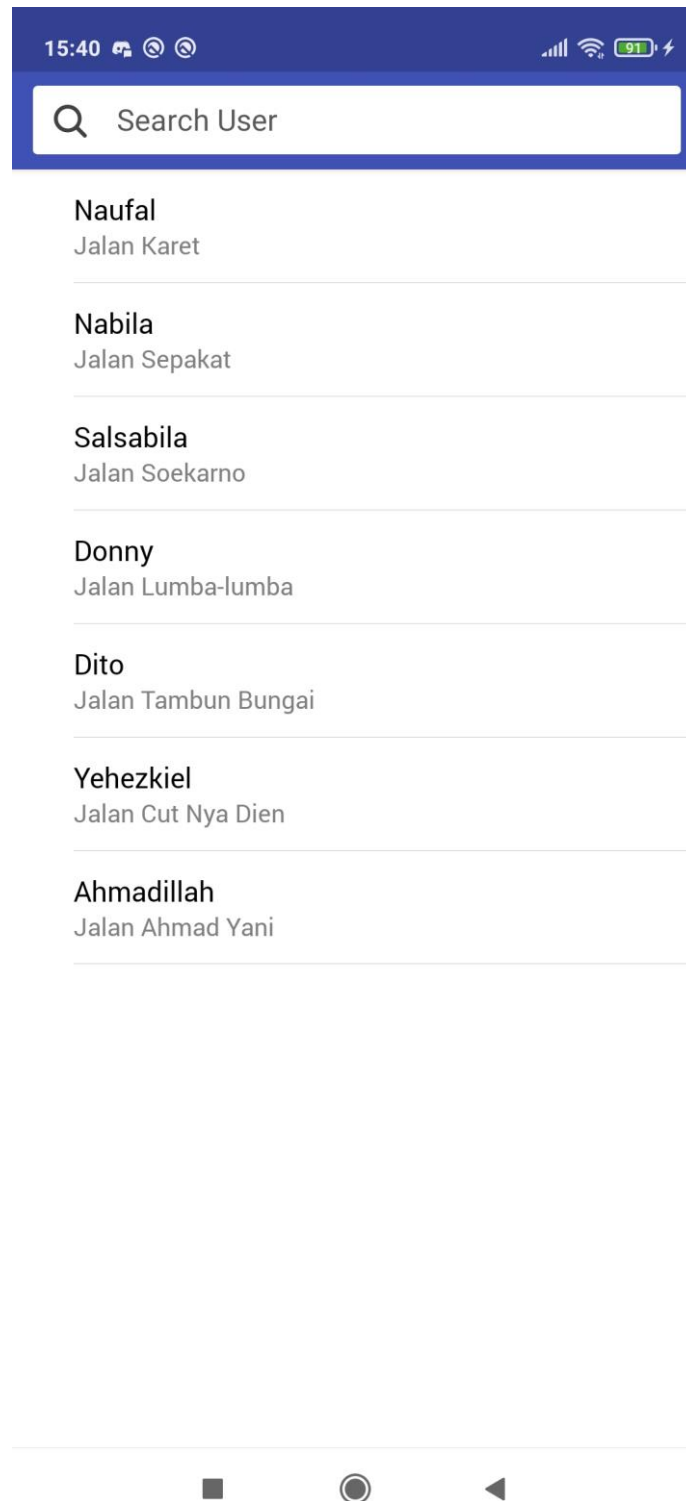
**Gambar 2.7** Komponen Header dari *render()*

Komponen `<content>` akan menampilkan daftar data. data ditampilkan dengan menggunakan tag `<ListItem>`. Data yang akan ditampilkan diambil dari *usersFiltered*.

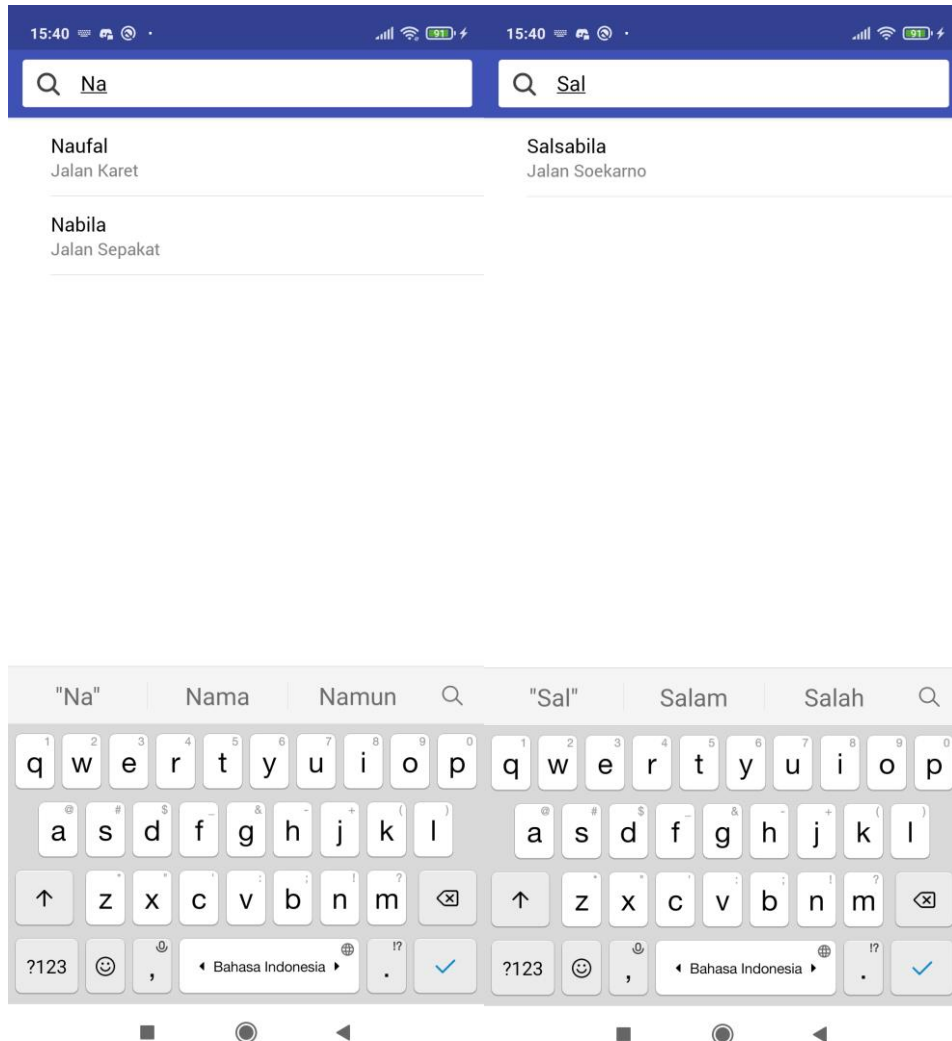
```
<Content>
  Complexity is 5 Everything is cool!
  {this.state.usersFiltered.map(item => (
    <ListItem avatar key={item.id}>
      <Body>
        <Text>{item.name}</Text>
        <Text note>{item.address}</Text>
      </Body>
    </ListItem>
  ))}
</Content>
</Container>
```

**Gambar 2.8** Komponen content dari *render()*





**Gambar 2.9** Tampilan awal Aplikasi



**Gambar 2.10** Hasil ketika melakukan pencarian

### **BAB III**

### **KESIMPULAN**

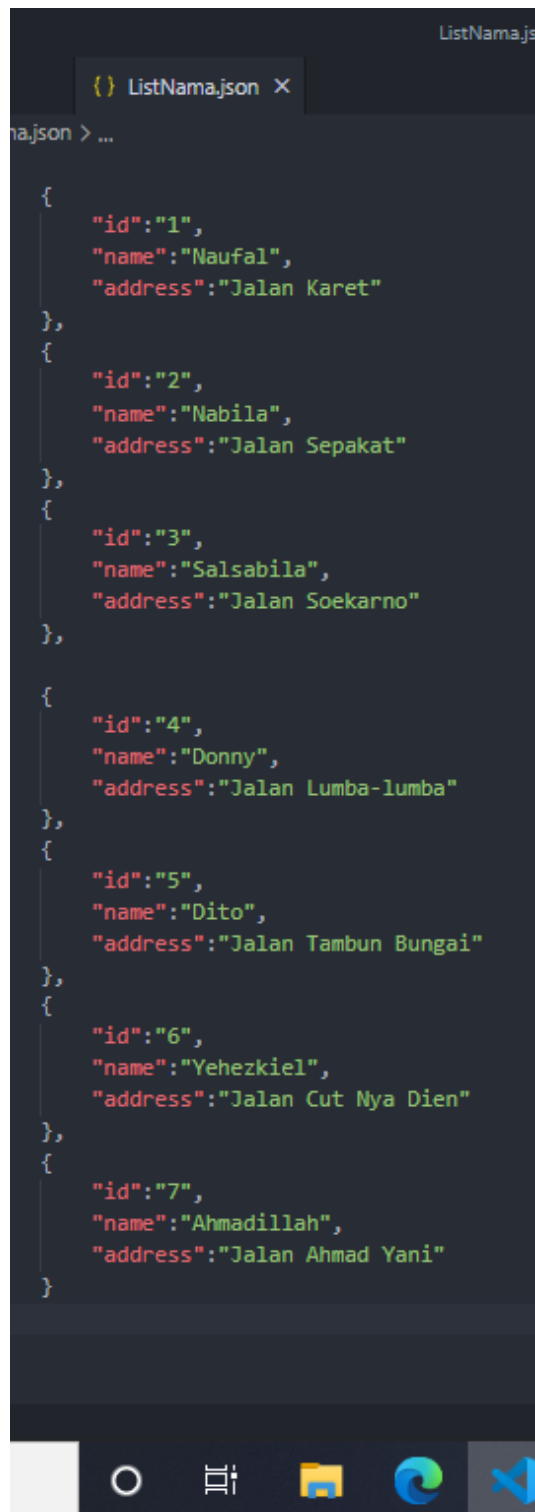
Flat list merupakan fitur yang sangat penting dalam react-native karena dengan menggunakan, sistem flat list, kita dapat menampilkan banyak data dalam aplikasi. Namun jika data terlalu banyak, tentunya akan menjadi susah untuk menemukan data yang diperlukan. Dengan menggunakan filtered search, kita mampu memfilter data yang diperlukan dan data yang tidak. Hal ini tentunya akan meningkatkan tingkat kepuasan dari pengalaman pengguna.

## **DAFTAR PUSTAKA**

NativeBase. 2021. "Introduction · NativeBase." <https://docs.nativebase.io/> (May 17, 2021).

Pedamkar, Priya. 2020. "React Native FlatList | Features & Uses of FlatList Component." <https://www.educba.com/react-native-flatlist/> (May 17, 2021).

## LAMPIRAN



```

ListNama.js
{} ListNama.json X
nama.json > ...
{
  "id": "1",
  "name": "Naufal",
  "address": "Jalan Karet"
},
{
  "id": "2",
  "name": "Nabila",
  "address": "Jalan Sepakat"
},
{
  "id": "3",
  "name": "Salsabila",
  "address": "Jalan Soekarno"
},
{
  "id": "4",
  "name": "Donny",
  "address": "Jalan Lumba-lumba"
},
{
  "id": "5",
  "name": "Dito",
  "address": "Jalan Tambun Bungai"
},
{
  "id": "6",
  "name": "Yehezkiel",
  "address": "Jalan Cut Nya Dien"
},
{
  "id": "7",
  "name": "Ahmadillah",
  "address": "Jalan Ahmad Yani"
}

```

**Gambar 2.1** Daftar data yang akan ditampilkan ke aplikasi

```

C:\WINDOWS\system32\cmd.exe - "node" "C:\Users\User\AppData\Roaming\npm
Microsoft Windows [Version 10.0.19041.985]
(c) Microsoft Corporation. All rights reserved.

E:\Semester 4\dzikriProject>npm install native-base --save
up to date, audited 1281 packages in 8s
71 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities

```

**Gambar 2.2** Instalasi library NativeBase

```

C:\WINDOWS\system32\cmd.exe - "node" "C:\Users\User\AppData\Roaming\npm\node_modules\react-native-cli\index.js" run-android
E:\Semester 4\dzikriProject>react-native link
info Linking assets to ios project
warn Group 'Resources' does not exist in your Xcode project. We have created it automatically for you.
info Linking assets to android project
success Assets have been successfully linked to your project
(node:11996) Warning: Accessing non-existent property 'padLevels' of module exports inside circular dependency
(Use `node --trace-warnings ...` to show where the warning was created)

```

**Gambar 2.3** Menghubungkan library ke project

```

JS App.js  X  {} ListNama.json
JS App.js > ...
1  import React, {Component} from 'react';
2  import {
3    Container,
4    Header,
5    Item,
6    Content,
7    Body,
8    Icon,
9    Text,
10   ListItem,
11   Input,
12 } from 'native-base';
13 let helperArray = require('./ListNama.json');
14

```

**Gambar 2.4** Import library dan variabel *helperArray* yang menyimpan data dari *listnama.json*

```
Complexity is 14 You must be kidding
export default class GeneralExample extends Component {
  constructor(props) {
    super(props);
    this.state = {
      allUsers: helperArray,
      usersFiltered: helperArray,
    };
  }
}
```

**Gambar 2.5** Membuat constructor untuk menyusun data

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

**Gambar 2.6** Function *searchUser* untuk menyeleksi data berdasarkan inputan

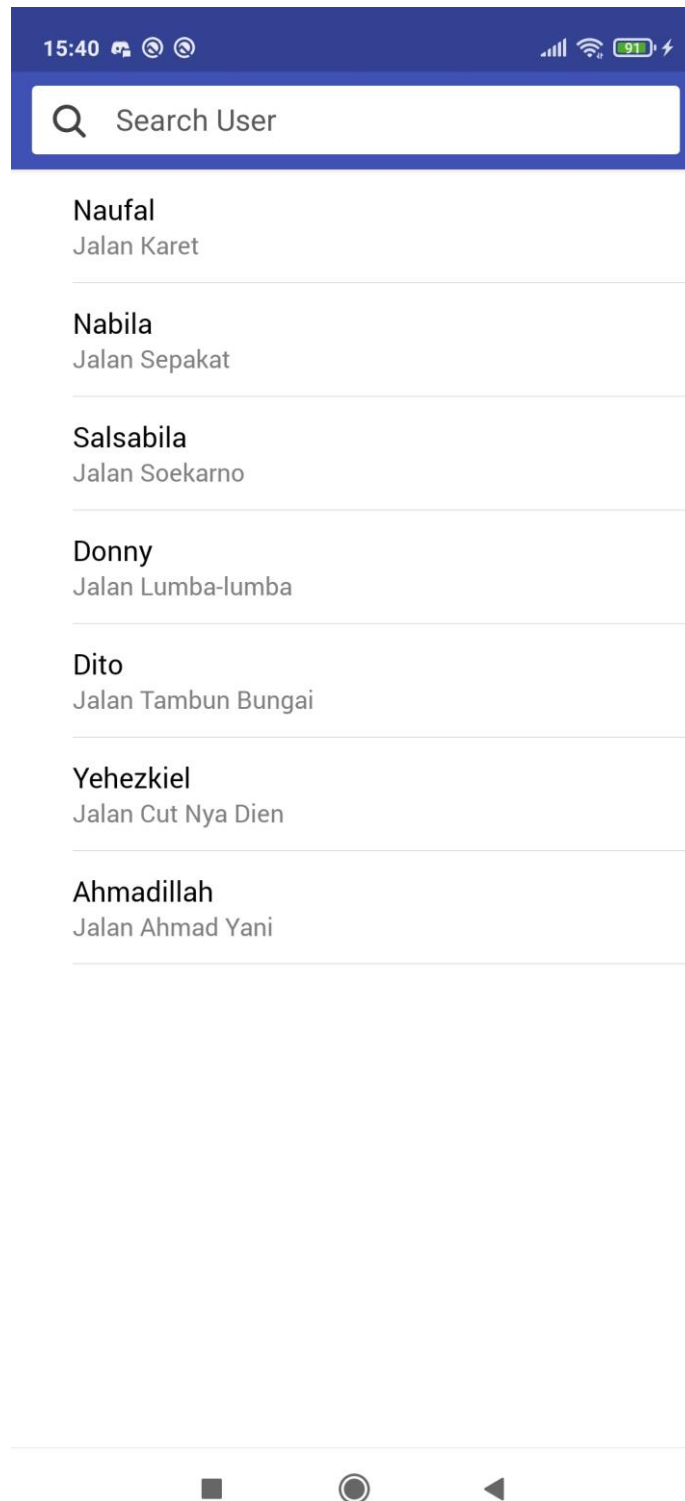
```
Complexity is 14 You must be kidding
render() {
  return (
    <Container>
      <Header searchBar rounded>
        <Item>
          <Icon name="search" />
          <Input
            placeholder="Search User"
            onChangeText={text => {
              this.searchUser(text);
            }}
          />
        </Item>
      </Header>
    </Container>
  );
}
```

**Gambar 2.7** Komponen Header dari *render()*

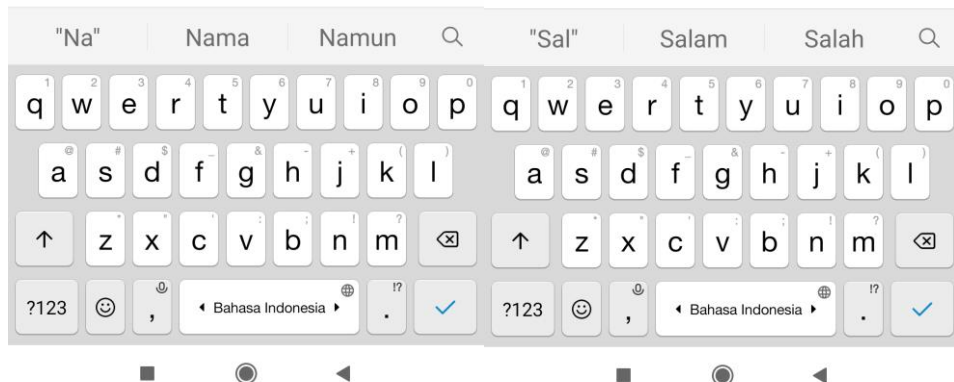
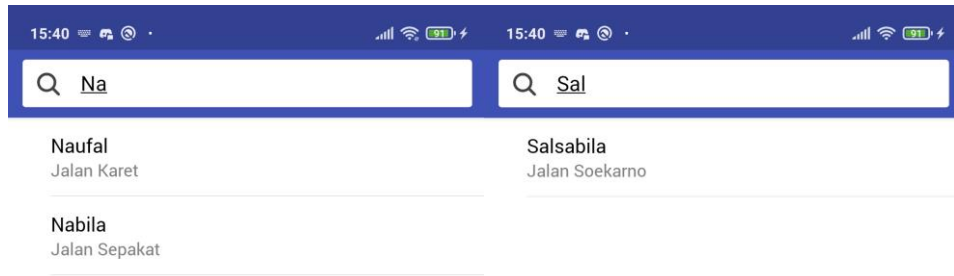
```
<Content>
  Complexity is 5 Everything is cool!
  {this.state.usersFiltered.map(item => (
    <ListItem avatar key={item.id}>
      <Body>
        <Text>{item.name}</Text>
        <Text note>{item.address}</Text>
      </Body>
    </ListItem>
  ))}
</Content>
</Container>
```

**Gambar 2.8** Komponen content dari *render()*





**Gambar 2.9** Tampilan awal Aplikasi



**Gambar 2.10** Hasil ketika melakukan pencarian