LAPORAN HASIL PRAKTIKUM
PEMROGRAMAN WEB & MOBILE

NAMA : ELNATAN KENINGATKO
NIM : 193020503038
KELAS : A
MODUL : VI (Search by Flat List)

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PALANGKA RAYA
2021

BAB I
LANDASAN TEORI

1.1. Landasan Teori

FlatList adalah sebuah component yang digunakan untuk me-render lists yang telah di-release pada React Native versi 0.43 dimana tidak memerlukan dataSource seperti pada ListView. Flatlist memiliki banyak fitur yang praktis sehingga menjadikannya pilihan utama untuk mengolah list sederhana maupun yang bersifat kompleks. Beberapa jenis fiturnya yaitu:

1. Sepenuhnya lintas platform.
2. Mode horizontal opsional.
3. Callback visibilitas yang dapat dikonfigurasi.
4. Dukungan header.
5. Dukungan footer.
6. Dukungan pemisah.
7. Tarik untuk Menyegarkan.
8. Gulir memuat.
9. Dukungan ScrollToIndex.
10. Dukungan banyak kolom.

BAB II

PEMBAHASAN

2.1. App.js

```
import React, { Component } from 'react';
import {
	Container,
	Header,
	Content,
	Left,
	Body,
	Icon,
	Text,
	ListItem,
	Thumbnail,
	Input,
	Item,
} from 'native-base';

let helperArray = require('./userList.json')
export default class ContentExample extends Component {
	constructor(props){
		super(props);
		this.state = {
			allUsers: helperArray,
			usersFiltered: helperArray,
		};
	}

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

searchUser(textToSearch){
	this.setState({
		usersFiltered: this.state.allUsers.filter(i =>
```

```
            i.name.includes(textToSearch),
            ),
      });
}

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

Blok code di atas merupakan file bernama *App.js* yang mana adalah main program *React Native* menggunakan *Android Studio*. Dengan diawali penggunaan *import* yang berfungsi untuk memasukan suatu method atau perintah, sehingga perintah tersebut dapat aktif dan digunakan.

Lalu, terdapat perintah *let* yang digunakan untuk mendeskripsikan variable bernama *helpArray* yang menampung data dari file *userList.json*. Kemudian, terdapat *export default* yang digunakan untuk mengekspor class bernama *ContentExample* dengan memanggil fungsi *Component*. Di dalam class *ContentExample* ini memiliki class *constructor* yang mana merupakan method untuk memberikan nilai awal dengan parameter berupa *props* kepada class *super* yang juga memiliki parameter berupa *props*. Terdapat juga penggunaan *state* yang berfungsi untuk mendeklarasikan variable *allUsers* dan *usersFiltered* yang menampung data dari variable *helperArray* ke dalam class *super*.

Setelah itu, terdapat dua buah metode searching bernama *searchUser* yang memiliki parameter berupa *textToSearch*. Di dalam *searchUser* ini terdapat perintah *setState* yang fungsinya ketika data di dalam *component* berubah, maka secara otomatis *component* akan dilakukan *render* ulang. Kedua metode searching ini hanya memiliki perbedaan yang mana *searchUser* pertama digunakan untuk mencari data menggunakan huruf kecil semua, sedangkan *searchUser* kedua digunakan untuk mencari data dengan penulisan yang sama sesuai dengan data yang dicari.

Terdapat fungsi *render* yang digunakan sebagai alternative dari template. Fungsi *render* ini memiliki properti-properti yang digunakan untuk membuat tampilan pada layar. Salah satunya yaitu seperti pada *Header searchBar rounded* yang berfungsi untuk menampilkan kolom pencarian.

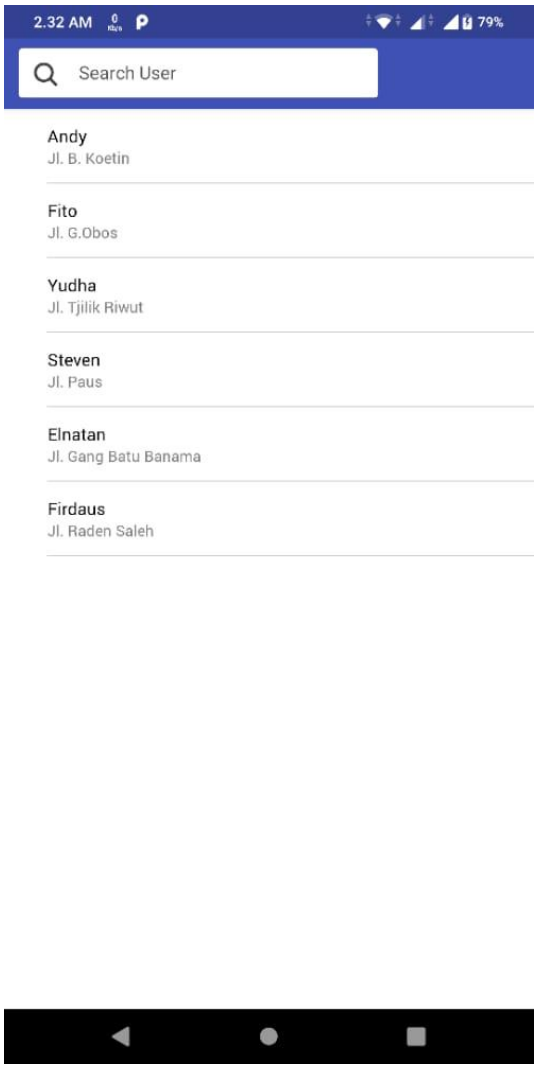
2.2. userList.json

```
[
    {
        "name": "Andy",
        "address": "Jl. B. Koetin",
    },

    {
        "name": "Fito",
```

```
        "address": "Jl. G.Obos",
    },

    {
        "name": "Yudha",
        "address": "Jl. Tjilik Riwut",
    },

    {
        "name": "Steven",
        "address": "Jl. Paus",
    },

    {
        "name": "Elnatan",
        "address": "Jl. Gang Batu Banama",
    },

    {
        "name": "Firdaus",
        "address": "Jl. Raden Saleh",
    }
]
```

Blok code di atas merupakan file bernama *userList.json* yang dipanggil ke dalam file *App.js*. File *userList.json* ini memiliki data-data berupa *name* dan *address* yang akan ditampilkan di layar.

Sehingga, apabila program dijalankan, akan menampilkan output seperti berikut.



Gambar 2.1. Tampilan Awal Program pada Layar.

Gambar 2.2. Melakukan Pencarian dengan Kata “El”.

BAB III
KESIMPULAN

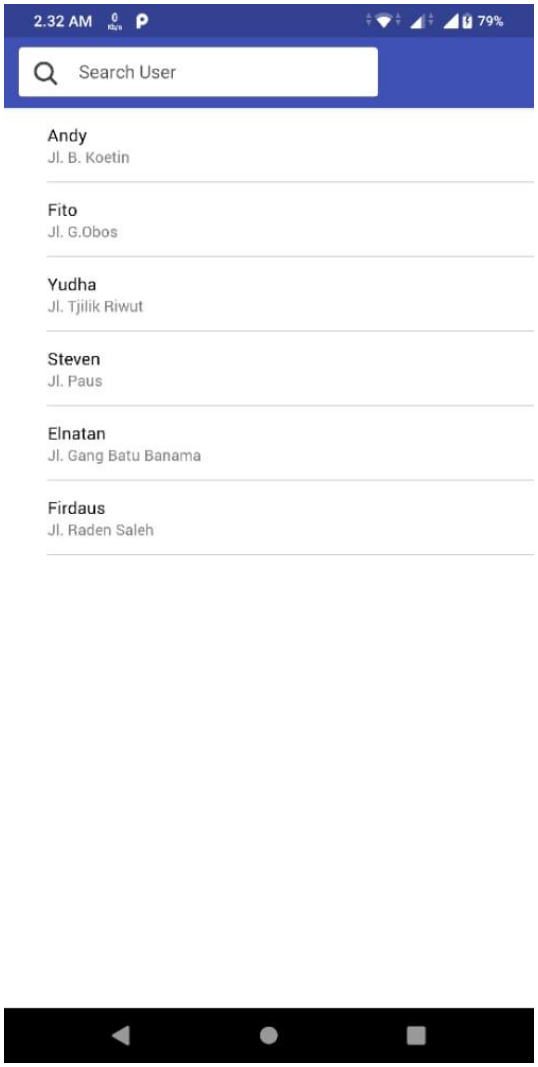
Penggunaan Flat List pada umumnya sangat dianjurkan untuk mengolah list sederhana maupun kompleks dikarenakan memiliki banyak fitur yang praktis, mengingat bahwa Flat List ini juga merupakan hal baru yang direlease oleh React Native.

DAFTAR PUSTAKA

Sandi, Anugrah. 2018. *React Native: Mengolah List Data*. Diakses dari https://daengweb.id/react-native-mengolah-list-data. Pada 23 Mei 2021.

Unknown. 2021. *FlatList*. Diakses dari reactnative.dev/docs/flatlist. Pada 23 Mei 2021.

LAMPIRAN



Gambar 2.1. Tampilan Awal Program pada Layar.

Gambar 2.2. Melakukan Pencarian dengan Kata “El”.