React Native

iOS

Android

MOBILE APPLICATION REACT NATIVE ITB STIKOM BALI 2022

Dr. Putu Desiana Wulaning Ayu S.T.,M.T

DAFTAR ISI

REACT NATIVE – OVERVIEW

React Native adalah sebuah framework javascript yang dikembang kan oleh facebook. React Native memungkinkan kamu untuk membuat aplikasi mobile android atau ios menggunakan teknologi web. Banyak framework javascript yang bisa digunakan untuk membuat aplikasi android atau ios, akan tetapi React Native ini berbeda dengan framework-framework javascript yang lainya. React Native tidak membuat aplikasi hybrid dimana aplikasi berjalan di javascript runtime tetapi membuat real aplikasi dimana untuk android akan di compile di java dan untuk IOS akan di compile diObjective-C.

Fitur React Native

Dalam React Native terdapat beberapa kompenen yang membuat React Native dapat digunakan untuk membuat aplikasi mobile cross-platform. Berikut ini adalah beberapa fitur dari React Native:

ReactJS - Sebelum React Native dikeluarkan oleh facebook, framework yang pertama dikenal adalah ReactJS, yaitu framework javascript yang bisa digunakan untuk membuat aplikasi web dengan javascript.

Native - Android dan IOS memiliki arsitektur yang berbeda, fitur native inilah yang berfungsi untuk mengatur komponen dari masing - masing platform.

Platform - Untuk saat ini platform yang telah terintegrasi dengan react native adalah Android dan IOS.

Keunggulan React Native

Javascript – Menggunakan bahasa pemrograman javascript untuk membangun aplikasi mobile. Jika sudah berpengalaman dengan web development tentunya sudah akrab dengan javascript dan sudah memiliki dasar untuk mulai membuat aplikasi dengan react native.

Code Sharing – Code yang dibuat dapat dibagikan untuk platform yang berbeda yaitu android dan ios.

Community - React Native memiliki cukup banyak peminat dari kalangan developer di dunia.

Kelemahan React Native

Components - Jika ingin membuat aplikasi, harus banyak memahami komponen dari masing-masing platform. Ketersediaan komponen untuk kedua platform tersebut masih terbatas sehingga perlu membuat beberapa komponen khuhus untuk platform tersebut.

REACT NATIVE – ENVIRONMENT SETUP

Ada dua cara untuk melakukan installasi dan membangun React Native :

Expo CLI - Jika baru mengenal pengembangan mobile, cara termudah untuk memulai adalah dengan Expo CLI. Expo adalah seperangkat alat yang dibangun di sekitar React Native dan, meskipun memiliki banyak fitur, fitur yang paling relevan saat ini adalah dapat membuat Anda menulis aplikasi React Native dalam beberapa menit. Hanya memerlukan Node.js versi terbaru dan mobile atau emulator.

React Native CLI - Jika ingin mencoba React Native langsung di browser web dapat mencoba Snack. Dengan syarat jika sudah terbiasa dengan pengembangan mobile. Memulai dengan React Native CLI diperlukan beberapa tambahan tools pendukung seperti Xcode atau Android Studio.

A. Expo CLI

Cara Pertama - Dengan Expo CLI ini perlu menginstal Node 12 LTS atau lebih, dapat menggunakan npm untuk menginstal utilitas baris perintah CLI Expo:

```
npm install -g expo-cli
```

Kemudian jalankan perintah berikut untuk membuat proyek React Native baru bernama "AwesomeProject":

```
expo init AwesomeProject

cd AwesomeProject

npm start
```

Jika proses running telah berhasil maka pada workspace command prompt akanmuncul QR Code.

Menjalankan Aplikasi :

Install Expo Client pada Handphone (Android / iOS) kemudian hubungkan dengan jaringan yang sama dengan komputer. Di Android, gunakan aplikasi Expo untuk memindai kode QR dari hasil running yang berhasil pada workspace command prompt untuk membuka Aplikasinya. Di iOS, gunakan pemindai kode QR bawaan aplikasi Kamera.

B. React Native CLI

Cara Kedua – Untuk melakukan instalasi React Native CLI, ada beberapa tools yang harus disipkan seperti

- NodeJS,
- Python2,
- JDK
- Android Studio (Android SDK)

Instalasi dapat dilakukan melalui Chocolatey yang merupakan

package manager untuk Windows. Untuk menjalankan aplikasi yang dibangun dengan React Native CLI, pengguna perlu melakukan instalasi Android SDK yang dapat diinstal melalui Android Studio. Setelah itu pastikan untuk melakukan konfigurasi environtment variable ANDROID_HOME dan menambahkan platform-tools pada path windows.

Instalasi melalui Chocolatey :

Buka Administrator Command Prompt (Klink kanan Command Promt dan pilih Run as Administrator) kemudian jalankan script atau bisa lihat panduannya pada link berikut :

https://docs.chocolatey.org/en-us/choco/setup#more-install-options

```
@"%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe" -NoProfile -
InputFormat None -ExecutionPolicy Bypass -Command
"[System.Net.ServicePointManager]::SecurityProtocol = 3072; iex ((New-Object
System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))"
&& SET "PATH=%PATH%;%ALLUSERSPROFILE%\chocolatey\bin"
```

Kemudian install Node js dan JDK

```
choco install -y nodejs.install openjdk8

choco install -y python2
```

Android Development Environment

- **Instalasi Android Studio**

Dibutuhkan beberapa fitur yang ada di android seperti :

- Android SDK

- Android SDK Platform
- Android Virtual Device

- **Instalasi Android SDK**

Untuk membangun aplikasi React Native dengan kode native membutuhkan Android 10 (Q) SDK secara khusus. Android SDK tambahan dapat diinstal melalui SDK Manager di Android Studio.

Pilih tab "Platform SDK" dari dalam SDK Manager, lalu "Show Package Details" di pojok kanan bawah. Cari entri Android 10 (Q), lalu pastikan item berikut dicentang:

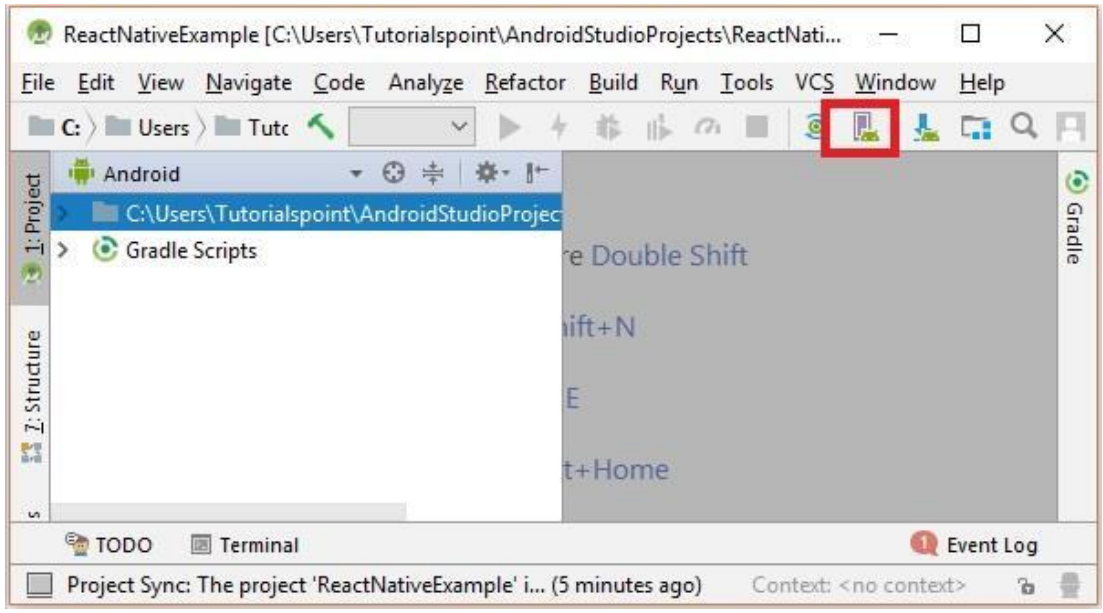
- Android SDK Platform 29
- Intel x86 Atom_64 System Image or Google APIs Intel x86 Atom System Image

Berikutnya, pilih “SDK Tools”, lalu "Show Package Details" di pojok kanan bawah, kemudian pilih SDK Tools versi 29.0.2

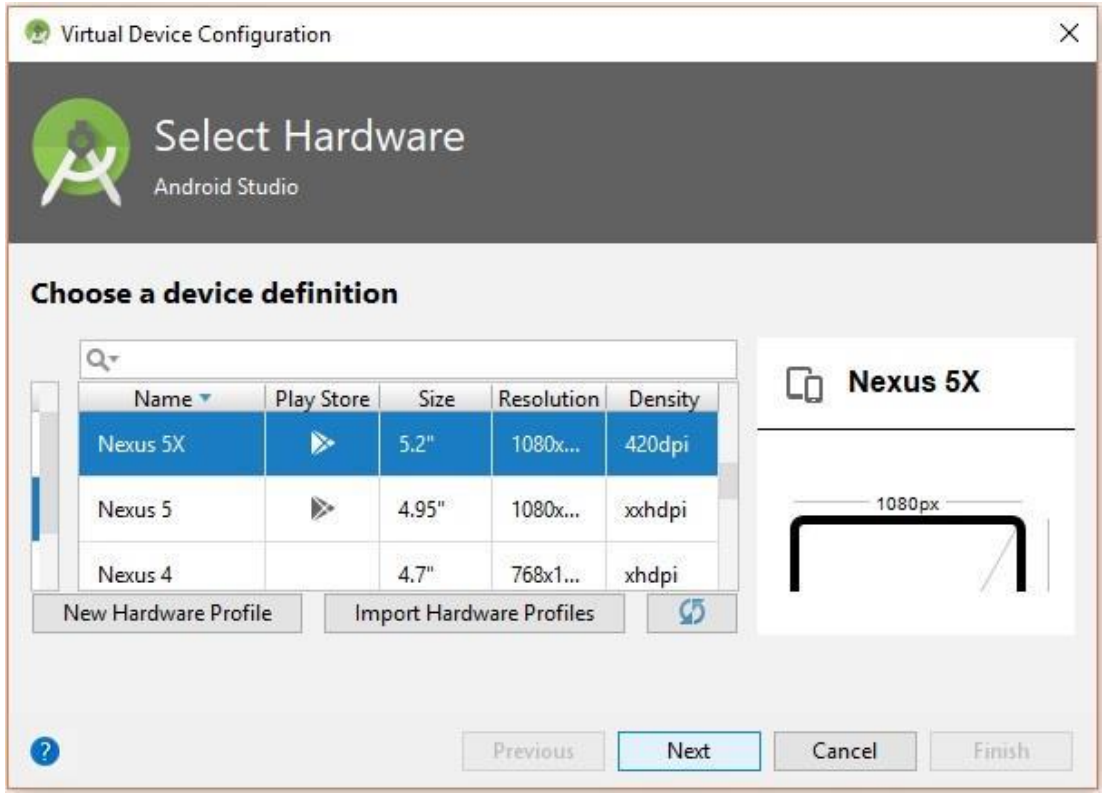
Terakhir, klik "Apply" untuk mengunduh dan Install Android SDK dan tools lainya.

- **Konfigurasi AVD Manager**

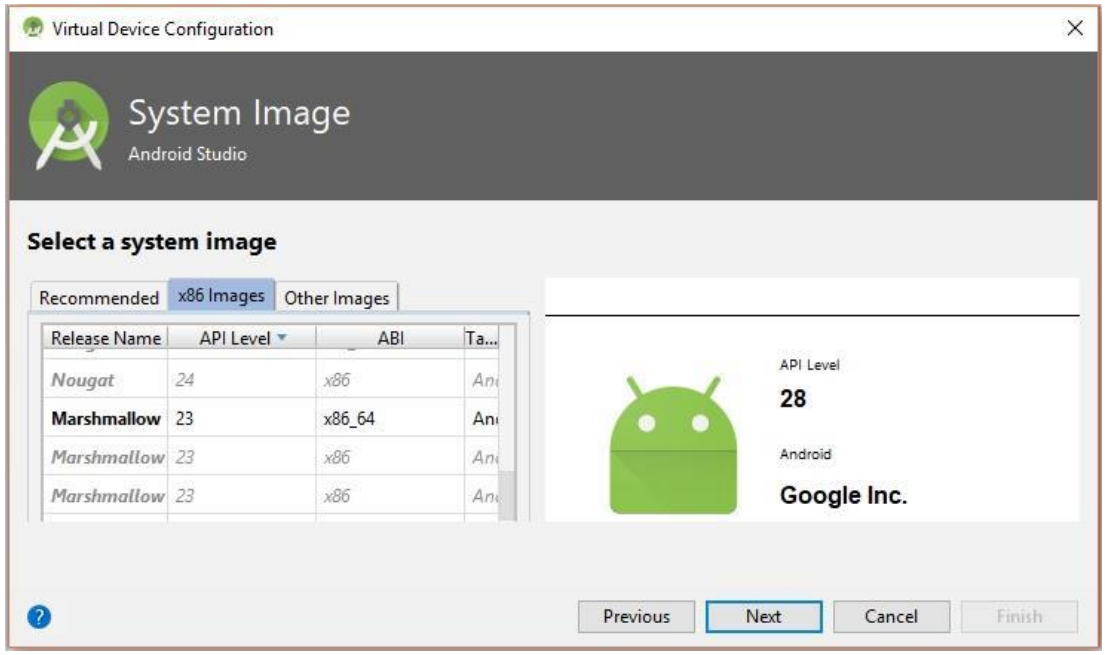
Untuk konfigurasi AVD Manager, klik icon seperti berikut :



Pilih perangkat, disarankan Nexus 5x



Pilih Next Button, maka akan tampil System Image Window dan pilih tab x86 Images. Kemudian pilih API Level 29 Image, Next.



Terakhir klik finish

Setelah selesai melakukan konfigurasi Android Virtual Manager, kemudian perhatikan kolom Actions dan pilih start android emulator seperti gambar berikut.

- **Konfigurasi ANDROID_HOME**

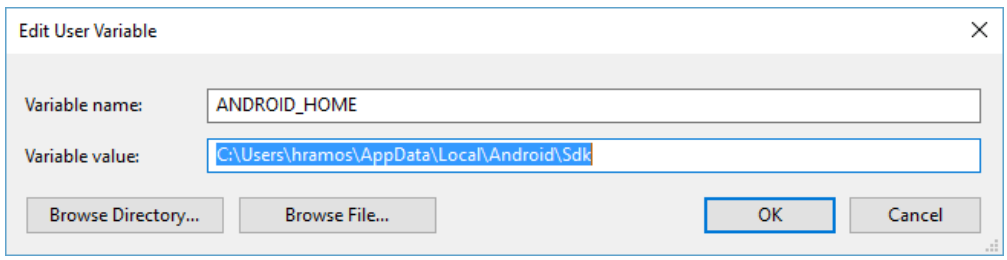
React Native memerlukan beberapa environment variables untuk disiapkan untuk membangun aplikasi.

a) Buka Windows Control Panel

b) Klik User Accounts

c) Klik Change my environment variables

d) Klik New, kemudian buat ANDROID_HOME sebagai points path Android SDK.



- **Menambahkan Platform-tool pada Path**

a) Buka Windows Control Panel

b) Klik User Accounts

c) Klik Change my environment variables

d) Pilih Path

e) Klik Edit

f) Klik New dan tambahkan path list untuk Platform-tools.

Biasanya lokasi foldernya :

```
%LOCALAPPDATA%\Android\Sdk\platform-tools
```

CREATE NEW APPLICATION

Buat project baru dengan nama ”AwesomeProject” :

```
npx react-native init AwesomeProject
```

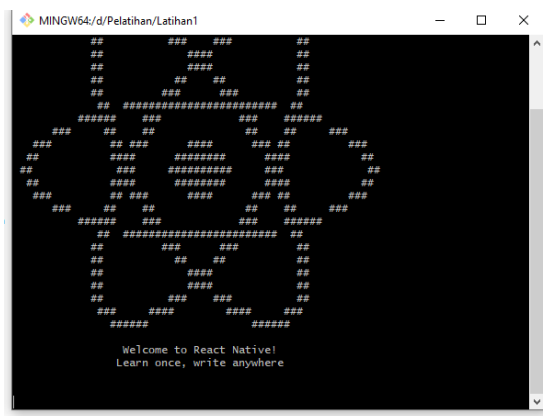
Pada perintah menggunakan npx, dimana React Native memiliki antarmuka baris perintah bawaan, yang dapat digunakan untuk menghasilkan proyek baru. Jadi dengan perintah ini package-package react native dapat diakses secara global menggunakan npx, yang disertakan dengan Node.js.

- **Menjalankan Aplikasi**

Jalankan server react native didalam project dengan perintah :

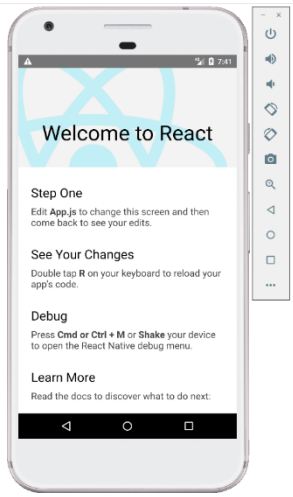
```
npx react-native start
```

Kalau sudah tampil seperti ini bearti berhasil menjalankan server



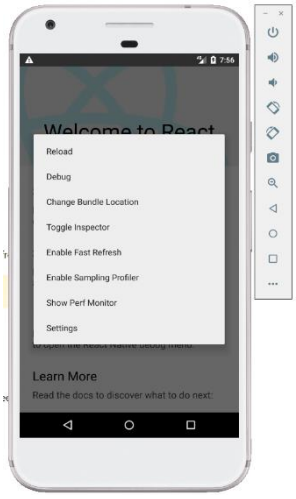
Kemudian Buka terminal baru di dalam folder project React Native untuk membuka aplikasi pada emulator. Jalankan perintah berikut:

```
npx react-native run-android
```



- **Hot Reloading**

Perubahan sintak pada app.js akan tampil secara otomatis pada emulator. Jika belum klik emulator android tekan *ctrl* + *m* lalu pilih Enable Hot Reloading / Enable Fast Refresh.

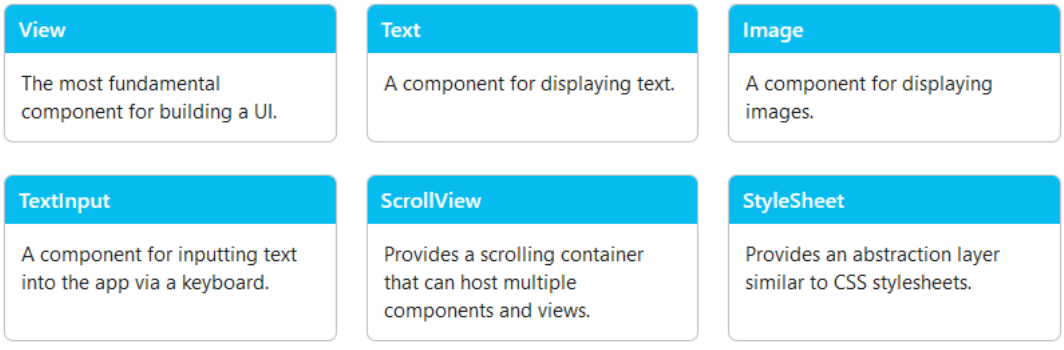


KOMPONEN DASAR REACT NATIVE

Komponen adalah bagian dari suatu aplikasi yang biasa terdiri dari prop dan state. Dalam react native komponen ini dapat dibangun dengan 2 cara yaitu **functional component (tidak memiliki state)** dan **class component (memiliki state dan prop)**. Didalam komponen dapat terdiri dari komponen-komponen kecil lainnya yang artinya menunjukan suatu bagian. Contoh : Membuat menu navigasi pada aplikasi, dimana dalam navigasi tersebut dapat terdiri dari berbagai komponen seperti tombol menu home, biodata dan seterusnya seperti berikut :

Komponen Navigasi	- Komponen home - Komponen biodata - Komponen blog - Komponen pencarian

Berikut adalah 6 jenis komponen yang sering dipakai dalam membuat aplikasi dengan react native :



a. Komponen View – Sebagai wadah untuk komponen lainnya.
b. Komponen Text – Untuk memunculkan sebuah Text.
c. Komponen Image – Untuk memunculkan sebuah Gambar.
d. Komponen TextInput – Untuk menerima inputan ke aplikasi.
e. Komponen ScrollView – untuk scroll halaman naik-turun.
f. Komponen SyleSheet - untuk style komponen-komponen.

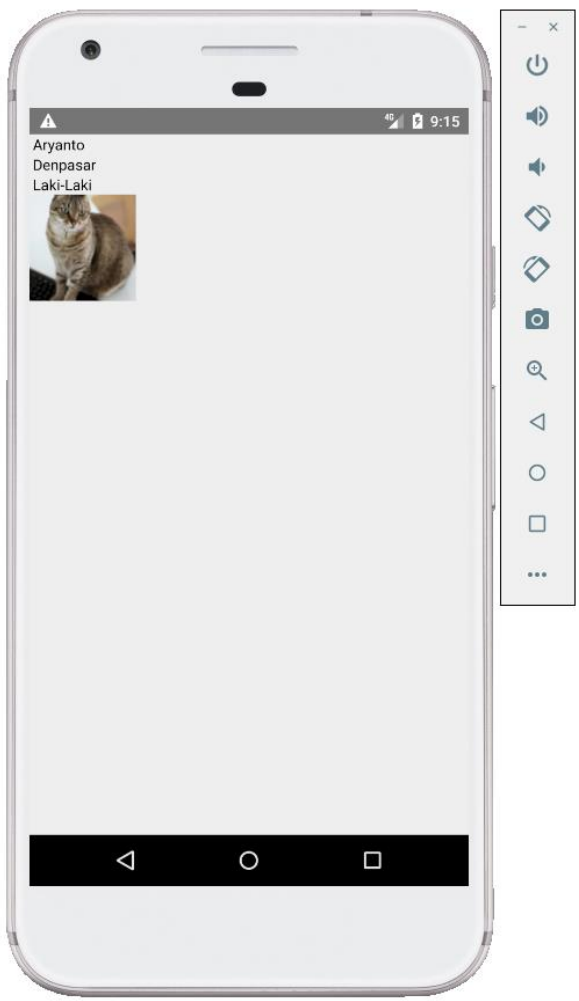
Berikut contoh membuat komponen secara custome dengan dua cara seperti berikut :

1. Functional Component (Hooks)

```
1. import React from 'react';
2. import {Text, View, Image} from 'react-native';
3.
4. const App  = () => {
5.   return(
6.     <View>
7.       <Biodata />
8.       <Foto />
9.     </View>
10.
11.  );
12.
13.};
14.
15.//Komponen dengan nama Biodata
16.const Biodata = () => {
17.  return(
18.    <View>
19.        <Text> Aryanto </Text>
20.        <Text> Denpasar </Text>
21.        <Text> Laki-Laki </Text>
22.    </View>
23.  );
24.};
25.
```

```
26. //Komponen dengan nama Foto
27. const Foto = () => {
28.   return(
29.     <Image source={{uri:'https://placeimg.com/100/100/any'}} sty
    le= {{width:100, height:100}} />
30.   );
31. };
32.
33. export default App;
```

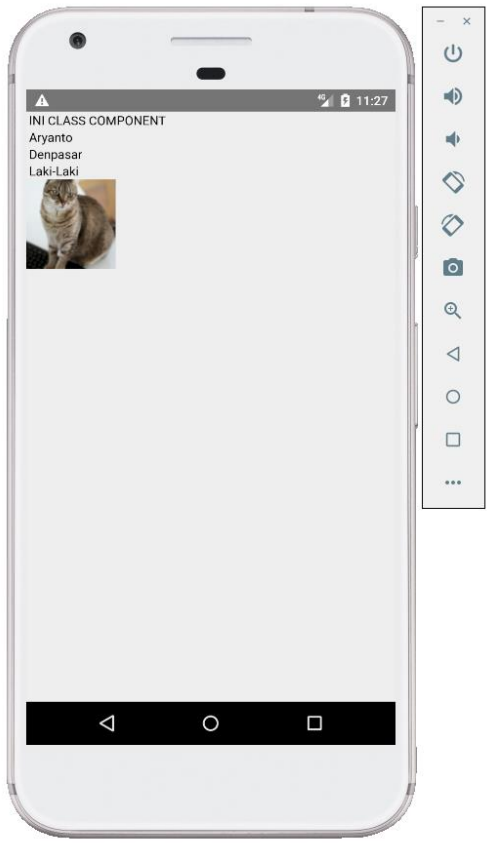
Dalam contoh sintak Functional Component terdapat dua komponen yang dibuat secara custome yaitu Biodata dan Foto. Dengan output aplikasi seperti berikut :



2. Class Component

```
1. import React, {Component} from 'react';
2. import {Text, View, Image} from 'react-native';
3.
4. const App  = () => {
5.   return(
6.     <View>
7.       <Biodata />
8.       <Foto />
9.     </View>
10.
11.  );
12.
13.};
14.
15.//Komponen Class Biodata
16.class Biodata extends Component {
17.  render (){
18.    return(
19.      <View>
20.        <Text> INI CLASS COMPONENT </Text>
21.        <Text> Aryanto </Text>
22.        <Text> Denpasar </Text>
23.        <Text> Laki-Laki </Text>
24.       </View>
25.    );
26.  }
27.}
28.
29.//Komponen Class Foto
30.class Foto extends Component{
31.  render (){
32.    return(
33.      <Image source={{uri:'https://placeimg.com/100/100/any'}} s
   tyle= {{width:100, height:100}} />
34.    );
35.  }
36.}
37.
38.export default App;
```

Dalam contoh sintak Class Component terdapat dua komponen yang dibuat secara custome yaitu Biodata dan Foto (sama seperti functional component sebelumnya). Dengan output aplikasi seperti berikut :



REACT NATIVE - STYLING

Terdapat dua cara penulisan style di React Native :

a. Cara yang pertama property style dituliskan langsung pada komponen (ini bukan yang terbaik karena sulit untuk membaca kode jika melakukan perubahan desain).
b. Cara yang kedua dituliskan dengan menggunakan Stylesheet.

Nama dan nilai value pada styling di React Native hampir sama seperti bagaimana CSS bekerja di web. Perbedaannya adalah cara penulisan namanya, pada React Native dituliskan dengan menggunakan camel case.

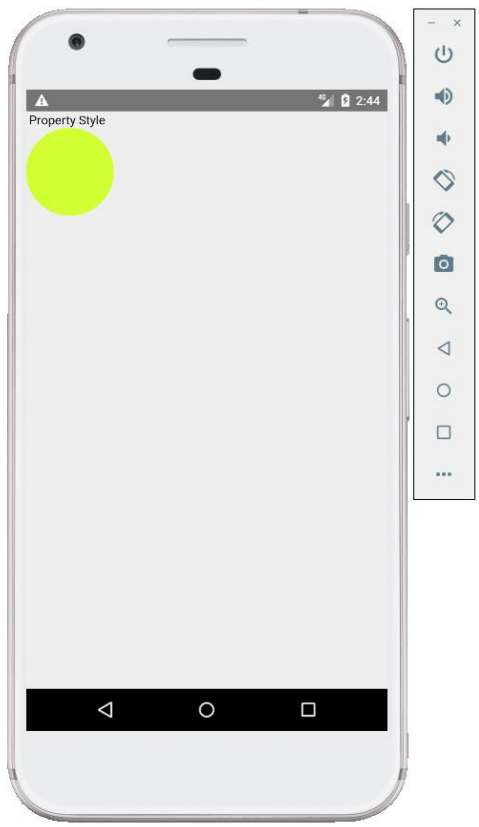
Berikut adalah contoh sintaks untuk styling komponen dengan cara Property Style atau Stylesheet:

1. Property Styling

```
1. import React, {Component} from 'react';
2. import {Text, View, Image} from 'react-native';
3.
4. const App  = () => {
5.   return(
6.     <View>
7.       <StylingProperty />
8.     </View>
9.
10.  );
11.
12.};
13.
14. const StylingProperty = () => {
15.   return(
16.     <View>
```

```
17.          <Text> Property Style</Text>
18.          <View style={{
19.            width: 100,
20.            height: 100,
21.            backgroundColor : '#d1ff33',
22.            borderRadius : 60,
23.          }}>
24.          </View>
25.       </View>
26.    );
27. }
28.
29.export default App;
```

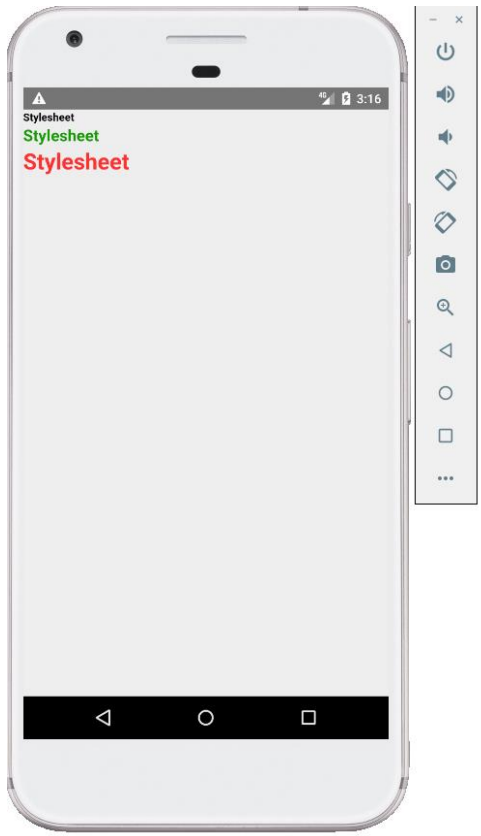
Pada sintaks styling property ini dicontohkan membuat sebuah lingkaran dengan komponen View, kemudian didalam komponen tersebut langsung dituliskan style-nya. Berikut hasilnya :



2. Stylesheet Styling

```
1. import React, {Component} from 'react';
2. import {Text, View, Image, StyleSheet} from 'react-native';
3.
4. const App  = () => {
5.   return(
6.     <View>
7.       <Text style={styles.textKecil}>Stylesheet</Text>
8.       <Text style={[styles.textMenengah, styles.textHijau]}>Styl
   esheet</Text>
9.       <Text style={[styles.textBesar, styles.textMerah]}>Stylesh
   eet</Text>
10.    </View>
11.  );
12.};
13.
14.const styles = StyleSheet.create({
15.  textKecil :{
16.    fontSize : 12,
17.    fontWeight : 'bold'
18.  },
19.  textMenengah :{
20.    fontSize : 18,
21.    fontWeight : 'bold'
22.  },
23.  textBesar :{
24.    fontSize : 25,
25.    fontWeight : 'bold'
26.  },
27.  textMerah :{
28.    color: '#ff3333'
29.  },
30.  textHijau : {
31.    color : '#169c0c'
32.  },
33.});
34.
35.export default App;
```

Pada sintaks stylesheet ini dicontohkan mengubah ukuran font Text, dimana stylingnya berada dalam sebuah function styles, kemudian styles ini digunakan pada komponen. Berikut hasilnya :



REACT NATIVE – EXPORT/IMPORT

Konsep export/import sangat penting untuk diketahui dikarenakan kebanyakan project yang besar nantinya akan membutuhkan ini untuk membagi file-file agar menjadi terstruktur dan rapi.

Sebagai contoh pada project sebelumnya telah dibuat fungsi Biodata dan Foto yang diletakan dalam satu file yang sama. Berikutnya, akan dipisahkan antaran fungsi ini menjadi filenya masing-masing (Biodata.js dan Foto.js) kemudian akan dilakukan proses export/import file kedalam App. js. Berikut adalah langkah-langkahnya :

Pertama – buat sebuah file baru dengan nama Biodata.js, file ini bisa diletakan sejajar dengan index.js atau dapat dibuatkan folder.

Biodata.js

```
import React from 'react';
import {Text, View, Image} from 'react-native';

//Komponen dengan nama Biodata
const Biodata = () => {
  return(
    <View>
        <Text> Aryanto </Text>
        <Text> Denpasar </Text>
        <Text> Laki-Laki </Text>
        <Text> STIKOM BALI </Text>
    </View>
  );
};
export default Biodata;
```

Kedua – kembali membuat sebuah file baru dengan nama Foto.js, file ini bisa diletakan sejajar dengan index.js atau dapat dibuatkan folder. Pastikan pada akhir sintak fungsinya di export dengan sintaks ”export default Foto”.

Foto.js

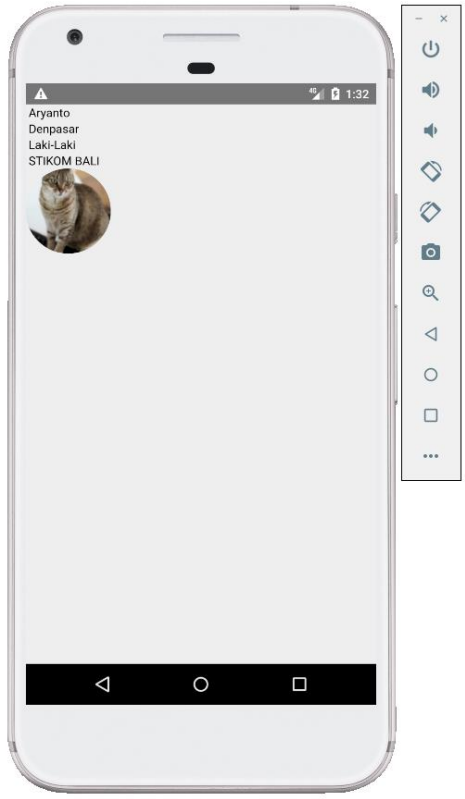
```
import React from 'react';
import {Text, View, Image} from 'react-native';

//Komponen dengan nama Foto
const Foto = () => {
  return(
    <Image source={{uri:'https://placeimg.com/100/100/any'}} style= {{w
idth:100, height:100, borderRadius:50}} />
  );
};

export default Foto;
```

Ketiga – setelah kedua file di-export berikutnya dilakukan import pada file App.js, dengan sintak seperti berikut :

App.js

```
import React from 'react';
import {View} from 'react-native';
import Biodata from './Biodata';
import Foto from './Foto';

const App  = () => {
  return(
    <View>
      <Biodata />
      <Foto />
    </View>
  );
};

export default App;
```

Output :



1) Flexbox

Dengan Flexbox dapat membuat ukuran tinggi lebar komponen secara responsif yang artinya akan menyesuaikan dengan ukuran perangkat layar yang digunakan untuk mengaksesnya. Untuk mendapatkan posisi tata letak yang responsif dapat mengkombinasikan property flexDirection, alignItems, dan justifyContent. Pada tabel berikut menunjukan deskripsi dari masing-masing property :

Property	Values	Description
flexDirection	column, row, rowreverse, column-reverse	Digunakan untuk menentukan apakahelemen akan diletakan sejajar secaravertical atau horizontal
justifyContent	center, flex-start,flex-end, stretch, baseline	Digunakan untuk menentukan bagaimanaelemen didistribusikan di dalamsecondary axis container (kebalikan dariflexDirection)
alignItems	Center, flex-start,flex-end, space-around,space-between, spaceevenly	Digunakan untuk menentukan bagaimanaelemen didistribusikan di dalamcontainer

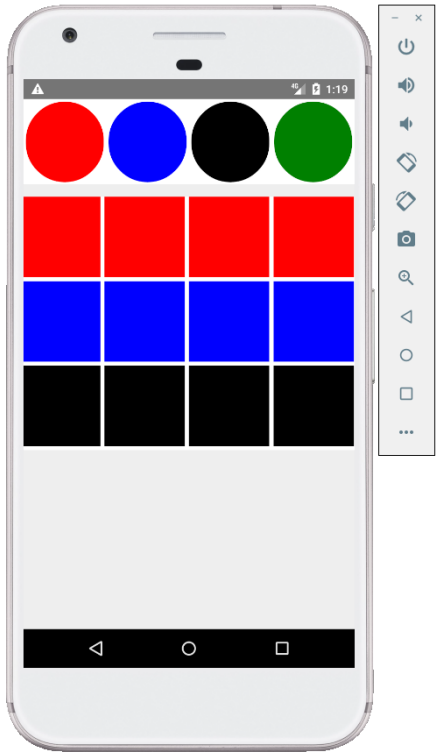
Berikut contoh sintaks Flex untuk membuat sebuah box yang dinamis :

App.js

```
import React, {Component} from 'react';
import {View, StyleSheet} from 'react-native';

const App  = () => {
  return(
    <View>
         <View style = {styles.containerRow}>
          <View style = {[styles.redbox, styles.lingkaran, {flex:1}]} /
>
          <View style = {[styles.bluebox, styles.lingkaran, {flex:1}]}
/>
          <View style = {[styles.blackbox, styles.lingkaran, {flex:1}]}
 />
          <View style = {[styles.greenbox, styles.lingkaran, {flex:1}]}
 />
        </View>

        <View style = {styles.containerRow}>
          <View style = {styles.containerColumn}>
            <View style = {[styles.redbox, {marginBottom:5}]} />
            <View style = {[styles.bluebox, {marginBottom:5}]} />
            <View style = {[styles.blackbox, {marginBottom:5}]} />
          </View>
          <View style = {styles.containerColumn}>
            <View style = {[styles.redbox, {marginBottom:5}]} />
            <View style = {[styles.bluebox, {marginBottom:5}]} />
            <View style = {[styles.blackbox, {marginBottom:5}]} />
          </View>
          <View style = {styles.containerColumn}>
            <View style = {[styles.redbox, {marginBottom:5}]} />
            <View style = {[styles.bluebox, {marginBottom:5}]} />
            <View style = {[styles.blackbox, {marginBottom:5}]} />
          </View>
          <View style = {styles.containerColumn}>
            <View style = {[styles.redbox, {marginBottom:5}]} />
            <View style = {[styles.bluebox, {marginBottom:5}]} />
            <View style = {[styles.blackbox, {marginBottom:5}]} />
          </View>
        </View>

    </View>
```

```
  );

};

const styles = StyleSheet.create ({
  containerRow: {
    flexDirection: 'row',
    justifyContent: 'center',
    alignItems: 'center',
    backgroundColor: 'white',
    marginBottom: 15,
  },
  containerColumn: {
    flexDirection: 'column',
    marginRight: 5,
    backgroundColor: 'white',
  },
  redbox: {
    width: 100,
    height: 100,
    backgroundColor: 'red'
  },
  bluebox: {
    width: 100,
    height: 100,
    backgroundColor: 'blue'
  },
  blackbox: {
    width: 100,
    height: 100,
    backgroundColor: 'black'
  },
  greenbox: {
    width: 100,
    height: 100,
    backgroundColor: 'green'
  },
  lingkaran: {
    borderRadius:50,
    margin:3,
  },
  })

export default App;
```

Output tampilan aplikasinya :



2) ListView

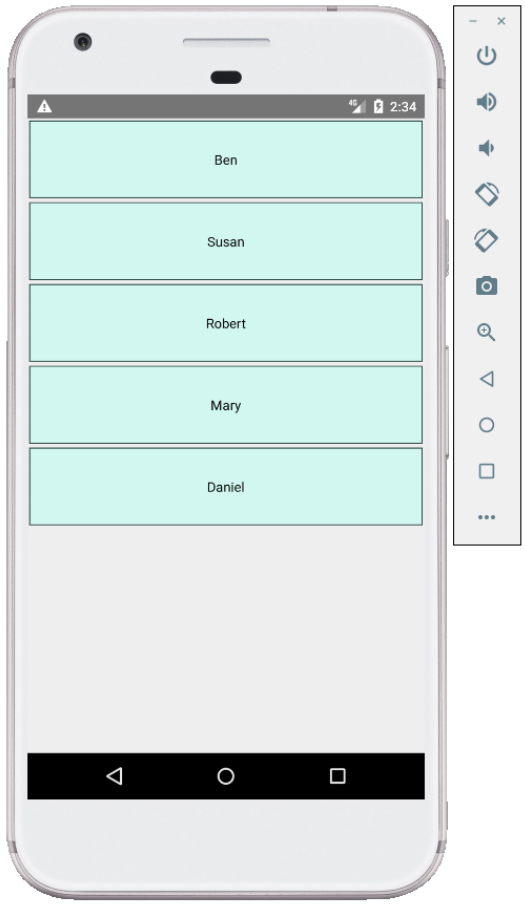
Berikutnya pembuatan listView. Dalam membuat sebuah list digunakan method map(). Method tersebut akan melakukan iterasi sebuah array dari item dan merendernya masing-masing. Berikut contoh penggunaan List View :

App.js

```
import React, {Component} from 'react'
import { View, Text, TouchableOpacity, StyleSheet} from 'react-native'

const App = () => {
   return(
    <ListView ></ListView>
   )
}
export default App;
```

```
class ListView extends Component {
   state = {
      names: [
         {'name': 'Ben', 'id': 1},
         {'name': 'Susan', 'id': 2},
         {'name': 'Robert', 'id': 3},
         {'name': 'Mary', 'id': 4},
         {'name': 'Daniel', 'id': 5},
      ]
   }
   alertItemName = (item) => {
      alert(item.name)
   }
render() {
   return (
      <View>
         {
            this.state.names.map((item, index) => (
               <TouchableOpacity
                  key = {item.id}
                  style = {styles.item}
                  onPress = {() => this.alertItemName(item)}>
                  <Text style = {styles.text}>
                     {item.name}
                  </Text>
               </TouchableOpacity>
            ))
         }
      </View>
      )
   }
  }

const styles = StyleSheet.create ({
   item: {
      justifyContent: 'space-between',
      alignItems: 'center',
      padding: 30,
      margin: 2,
      borderColor: '#2a4944',
      borderWidth: 1,
      backgroundColor: '#d2f7f1'
   },
   text: {
      color: 'black',
      alignItems: 'center'
   }
})
```

Output :



3) ScrollView

Pembangunan aplikasi terkadang membutuhkan banyak komponen, elemen, dan halaman yang panjang. Untuk mendukung hal tersebut, React Native memiliki elemen ScrollView. Elemen ini harus diletakkan di dalam komponen View dan memiliki tag penutup. Pembuatan ScrollView dapat dilakukan dengan dua cara yaitu :

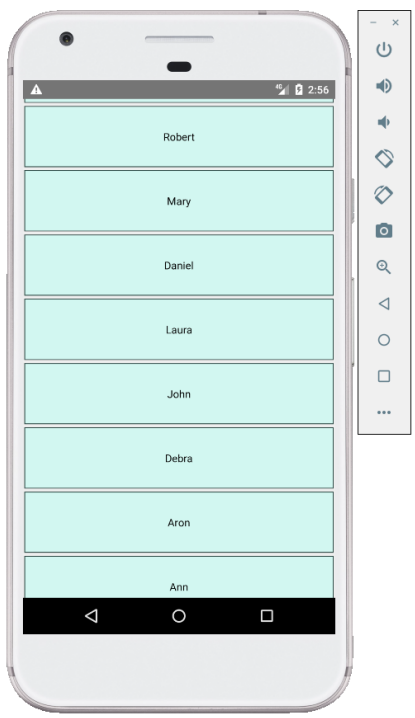
❖ ScrollView Vertikal

App.js

```
import React, {Component} from 'react'
import { View, Text, TouchableOpacity, StyleSheet, ScrollView } from 'r
eact-native'

const App = () => {
   return(
    <Scrollview  ></Scrollview  >
   )
}
export default App;

class Scrollview extends Component {
   state = {
      names: [
         {'name': 'Ben', 'id': 1},
         {'name': 'Susan', 'id': 2},
         {'name': 'Robert', 'id': 3},
         {'name': 'Mary', 'id': 4},
         {'name': 'Daniel', 'id': 5},
         {'name': 'Laura', 'id': 6},
         {'name': 'John', 'id': 7},
         {'name': 'Debra', 'id': 8},
         {'name': 'Aron', 'id': 9},
         {'name': 'Ann', 'id': 10},
         {'name': 'Steve', 'id': 11},
         {'name': 'Olivia', 'id': 12},
      ]
   }
   alertItemName = (item) => {
      alert(item.name)
   }
render() {
   return (
      <View>
         <ScrollView>
         {
            this.state.names.map((item, index) => (
               <TouchableOpacity
                  key = {item.id}
                  style = {styles.item}
                  onPress = {() => this.alertItemName(item)}>
                  <Text style = {styles.text}>
                     {item.name}
```

```
                    </Text>
                </TouchableOpacity>
            ))
          }
          </ScrollView>
      </View>
      )
   }
  }

const styles = StyleSheet.create ({
   item: {
      justifyContent: 'space-between',
      alignItems: 'center',
      padding: 30,
      margin: 2,
      borderColor: '#2a4944',
      borderWidth: 1,
      backgroundColor: '#d2f7f1'
   },
   text: {
      color: 'black',
      alignItems: 'center'
   }
})
```

Output :



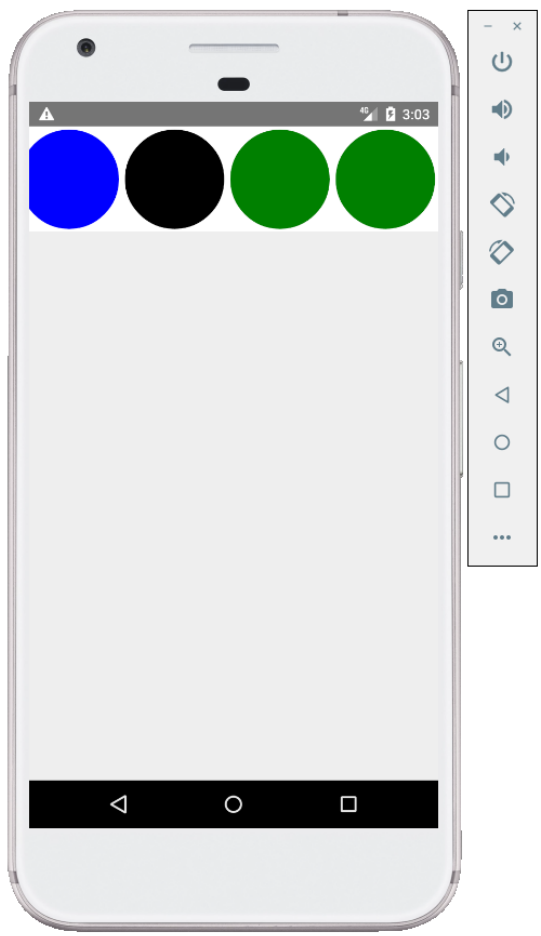
❖ **ScrollView Horizontal**

App.js

```
import React, {Component} from 'react';
import {View, StyleSheet, ScrollView} from 'react-native';

const App  = () => {
  return(
    <View>
          <View style = {styles.containerRow}>
            <ScrollView horizontal = {true}>
            <View style = {[styles.redbox, styles.lingkaran]} />
            <View style = {[styles.bluebox, styles.lingkaran]} />
            <View style = {[styles.blackbox, styles.lingkaran]} />
            <View style = {[styles.greenbox, styles.lingkaran]} />
            <View style = {[styles.greenbox, styles.lingkaran]} />
            <View style = {[styles.greenbox, styles.lingkaran]} />
            </ScrollView>
          </View>
    </View>


  );

};

const styles = StyleSheet.create ({
  containerRow: {
    flexDirection: 'row',
    justifyContent: 'center',
    alignItems: 'center',
    backgroundColor: 'white',
    marginBottom: 15,
  },
  redbox: {
    width: 100,
    height: 100,
    backgroundColor: 'red'
  },
  bluebox: {
    width: 100,
    height: 100,
    backgroundColor: 'blue'
  },
  blackbox: {
    width: 100,
```

```
    height: 100,
    backgroundColor: 'black'
  },
  greenbox: {
    width: 100,
    height: 100,
    backgroundColor: 'green'
  },
  lingkaran: {
    borderRadius:50,
    margin:3,
  },
  })

export default App;
```

Output :



REACT NATIVE – NAVIGATION

Navigasi merupakan struktur menu yang ada dalam sebuah aplikasi yang memungkinkan pengguna dapat beralih dari satu halaman ke halaman lainnya. Pada react native memiliki komponen navigasi yang dapat dibuat dengan createStackNavigator dan createBottonTabNavigator yang diletakkan di dalam sebuah container. Kemudian container ini dibuat dengan elemen NavigationContainer. Berikut penjelasan singkat dari komponen navigasi di react native.

- **createStackNavigator**, pengguna dapat beralih dari satu menu ke menu yang lain menggunakan tombol
- **createBottonTabNavigator** pengguna dapat beralih dari satu menu ke menu yang lain dengan menggunakan Menu Tab di bagian bawah layar.

Sebelum ke sintak program ada beberapa library yang perlu ditambahkan kedalam project aplikasi, berikut perintah untuk memasukan library tersebut.

```
npm install --global yarn
```

```
yarn add @react-navigation/native
```

```
yarn add react-native-reanimated react-native-gesture-handler react-native-screens react-native-safe-area-context @react-native-community/masked-view
```

```
yarn add @react-navigation/stack
```

```
yarn add @react-navigation/bottom-tabs
```

Berikut adalah contoh sintak dari komponen-komponen tersebut.

1. **createStackNavigator** Navigation Menggunakan Button **:**

App.js

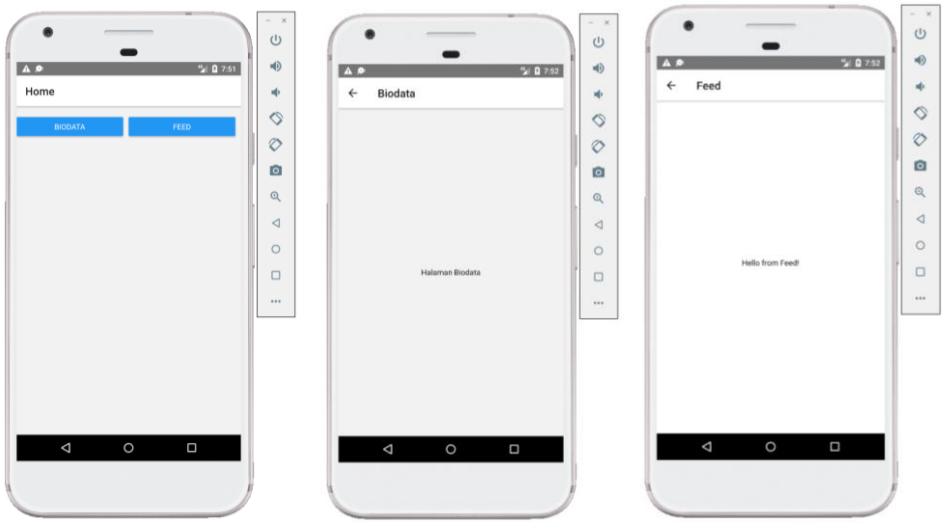
```
import * as React from 'react';
import { Button, View, Text } from 'react-native';
import { NavigationContainer } from '@react-navigation/native';
import { createStackNavigator } from '@react-navigation/stack';
import Feed from './Feed';


function HomeScreen({ navigation }) {
  return (
    <View style={{flexDirection:'row', marginTop:20}}>
      <View  style={{flex:1, paddingRight:10}}      >
          <Button
            title="Biodata"
            onPress={() => navigation.navigate('Biodata')}
          />
      </View>

      <View  style={{flex:1, paddingRight:10}}      >
          <Button
            title="Feed"
            onPress={() => navigation.navigate('Feed')}
          />
      </View>
    </View>
  );
}

function Biodata() {
  return (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'cent
er' }}>
      <Text>Halaman Biodata</Text>
    </View>
  );
}

const Stack = createStackNavigator();

function App() {
```

```
  return (
    <NavigationContainer>
      <Stack.Navigator initialRouteName="Home">
        <Stack.Screen name="Home" component={HomeScreen} />
        <Stack.Screen name="Biodata" component={Biodata} />
        <Stack.Screen name="Feed" component={Feed} />
      </Stack.Navigator>
    </NavigationContainer>
  );
}

export default App;
```

Feed.js

```
import React from 'react'
import {StyleSheet, Text, View} from 'react-native';

const Feed = () => {
    return (
        <View style={styles.container}>
            <Text> {"Hello from Feed!\n"} </Text>
        </View>
    )
}

export default Feed;

const styles = StyleSheet.create({
    container:{
        flex: 1,
        backgroundColor: '#fff',
        alignItems: 'center',
        justifyContent: 'center',
    },
});
```

Output :



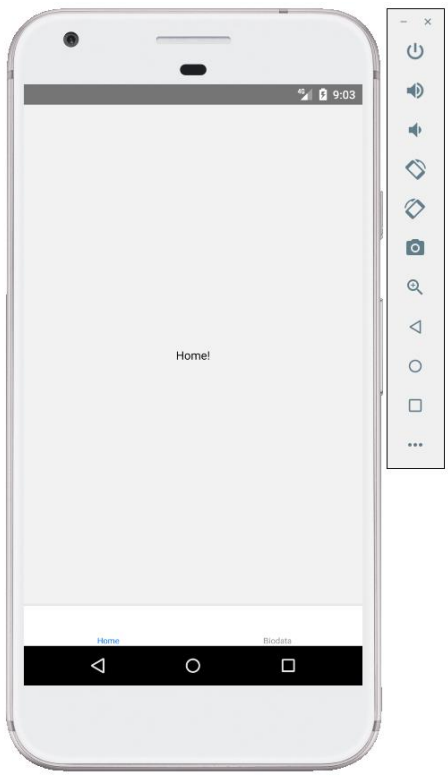
2. **createBottonTabNavigator** Navigation Menggunakan Tab :

App.js

```
import * as React from 'react';
import { Text, View } from 'react-native';
import { NavigationContainer } from '@react-navigation/native';
import { createBottomTabNavigator } from '@react-navigation/bottom-
tabs';

function HomeScreen() {
  return (
    <View style={{ flex: 1, justifyContent: 'center', alignItems: 'cent
er' }}>
      <Text>Home!</Text>
    </View>
  );
}

function Biodata() {
  return (
    <View style={{ flex: 1, justifyContent: 'center', alignItems: 'cent
er' }}>
      <Text>Settings!</Text>
    </View>
  );
```

```
}

const Tab = createBottomTabNavigator();

export default function App() {
  return (
    <NavigationContainer>
      <Tab.Navigator>
        <Tab.Screen name="Home" component={HomeScreen} />
        <Tab.Screen name="Biodata" component={Biodata} />
      </Tab.Navigator>
    </NavigationContainer>
  );
}
```

Output :



REACT NATIVE – PROPS

Data yang ada dalam komponen React di kelola dengan mengunakan **state** dan **props.** Pada pembahasan ini menjelaskan meteri komponen yang dinamis dengan menggunakan props (property). Dalam pemrograman web props itu seperti sebuah atribut dalam tag HTML. Dalam contoh berikut dijelaskan pembuatan props pada,

a) **Functional komponen** (props disebut dengan parameter).

b) **Class komponen** (props disebut property dari class diakses dengan keyword 'this')

Berikut contoh sintak props pada pembuat galeri produk:

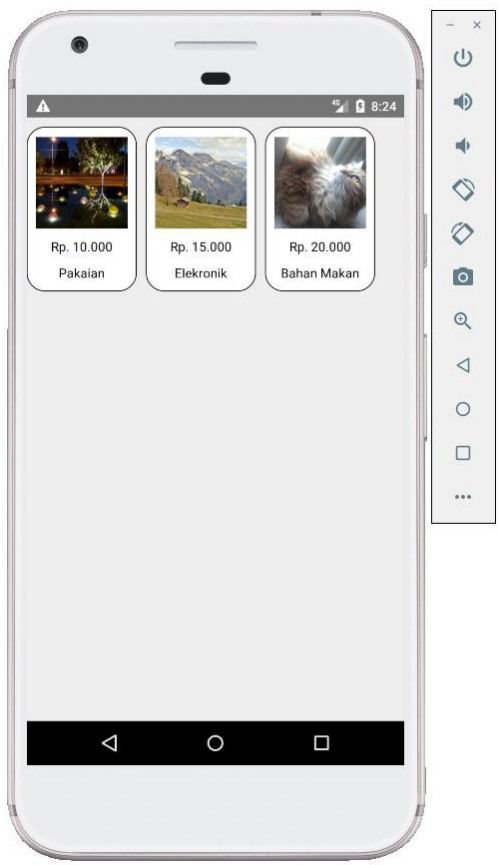
- **Contoh dengan Functional Component**

  App.js

```
import React from 'react'
import { View, Text, Image } from 'react-native'

const App = () => {
   return (
      <View style={{flexDirection:'row'}}>
         <Gallery
            gambar = 'https://placeimg.com/640/480/arch'
            harga = 'Rp. 10.000'
            deskripsi = 'Pakaian'
         />
         <Gallery
            gambar = 'https://placeimg.com/640/480/nature'
            harga = 'Rp. 15.000'
            deskripsi = 'Elekronik'
         />
         <Gallery
            gambar = 'https://placeimg.com/640/480/animals'
            harga = 'Rp. 20.000'
            deskripsi = 'Bahan Makan'
         />
      </View>
```

```
    );
}
const Gallery = props => {
    return(
        <View style={{
                borderColor:'black',
                borderWidth:1,
                borderRadius:20,
                padding:10,
                width:120,
                backgroundColor:'white',
                marginTop:10,
                marginRight:10,
                alignItems:'center'}}>
  <Image source={{uri:props.gambar}} style={{height:100, width:100}} />
  <Text style={{marginTop:10, color:'black'}}>{props.harga}</Text>
  <Text style={{marginTop:10, color:'black'}}>{props.deskripsi}</Text>
      </View>
    );
}
export default App;
```

Output :



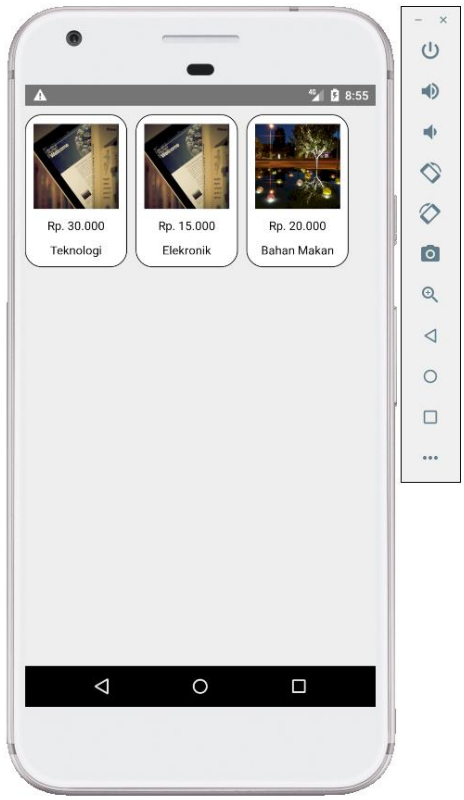
- **Contoh dengan Class Component**

  App.js

```
import React, {Component} from 'react'
import { View, Text, Image } from 'react-native'

const App = () => {
   return (
      <View style={{flexDirection:'row'}}>
         <Gallery
            gambar = 'https://placeimg.com/640/480/tech'
            harga = 'Rp. 30.000'
            deskripsi = 'Teknologi'
         />
         <Gallery
            gambar = 'https://placeimg.com/640/480/tech'
            harga = 'Rp. 15.000'
            deskripsi = 'Elekronik'
         />
         <Gallery
            gambar = 'https://placeimg.com/640/480/arch'
            harga = 'Rp. 20.000'
            deskripsi = 'Bahan Makan'
         />
      </View>
   );
}

class Gallery extends Component{
   render(){
      return(
         <View style={{
            borderColor:'black',
            borderWidth:1,
            borderRadius:20,
            padding:10,
            width:120,
            backgroundColor:'white',
            marginTop:10,
            marginRight:10,
            alignItems:'center'}}>
            <Image source={{uri:this.props.gambar}} style={{height:100,
 width:100}} />
            <Text style={{marginTop:10, color:'black'}}>{this.props.har
ga}</Text>
```

```
            <Text style={{marginTop:10, color:'black'}}>{this.props.des
kripsi}</Text>
          </View>
      );
   }
}
export default App;
```

Output :



Berikut adalah point penting yang perlu diperhatikan dalam pengunaan Props :

a) Fungsi atau Class dari props hanya dapat diubah valuenya oleh pemangilannya.

b) Prop umumnya digunakan untuk komunikasi data component dari parent komponent ke child component

REACT NATIVE – STATE

State merupakan data private dari sebuah komponen. Data ini hanya tersedia untuk komponen tersebut dan tidak bisa di akses dari komponen lain. Komponen merubah nilai state-nya sendiri. Berikut adalah contoh implementasi dari state :

Dalam contoh berikut dijelaskan pembuatan state pada,

a) **Functional komponen**.

b) **Class komponen**

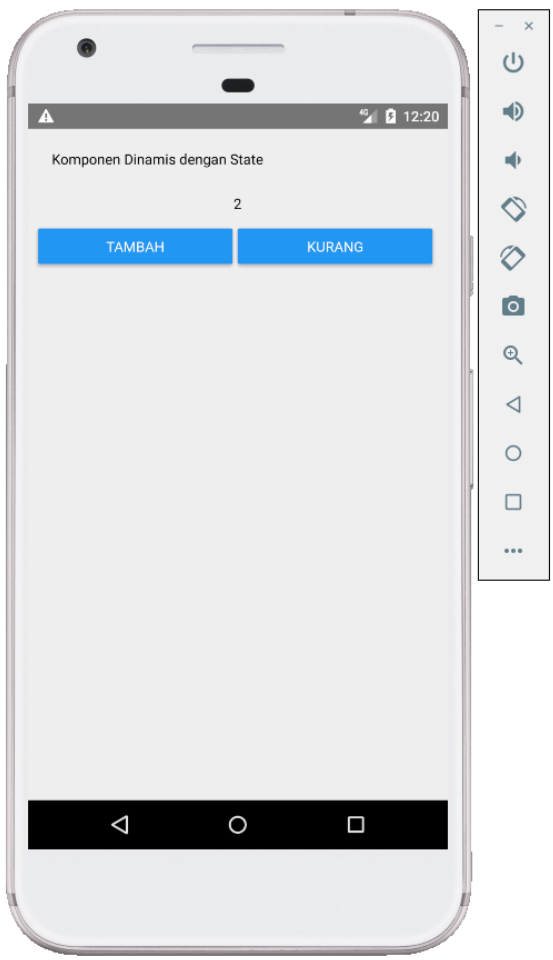
- **State dengan Functional Komponen**

App.js

```
import React, {useState, Component} from 'react'
import { View, Text, Button, StyleSheet } from 'react-native'

const App = () => {
   return(
      <View style={{padding:10}}>
            <Text style={{alignItems:'center', padding:10}}> Komponen D
inamis dengan State </Text>
            <Counter/>
      </View>
   );
}


const Counter = () => {
   const [number, setNumber] = useState(0);
   return(
      <View>
         <Text style={{textAlign:'center', padding:15}}>
            {number}
         </Text>
         <View style={{flexDirection:'row'}}>
            <View style={{marginRight:5, flex:1}}>
```

```
              <Button title='Tambah' onPress={() => setNumber(number +
1)} />
            </View>
            <View style={{marginRight:5, flex:1}}>
              <Button title='Kurang' onPress={() => setNumber(number -
1)} />
            </View>
          </View>
        </View>
    );
}
export default App;
```

Output :

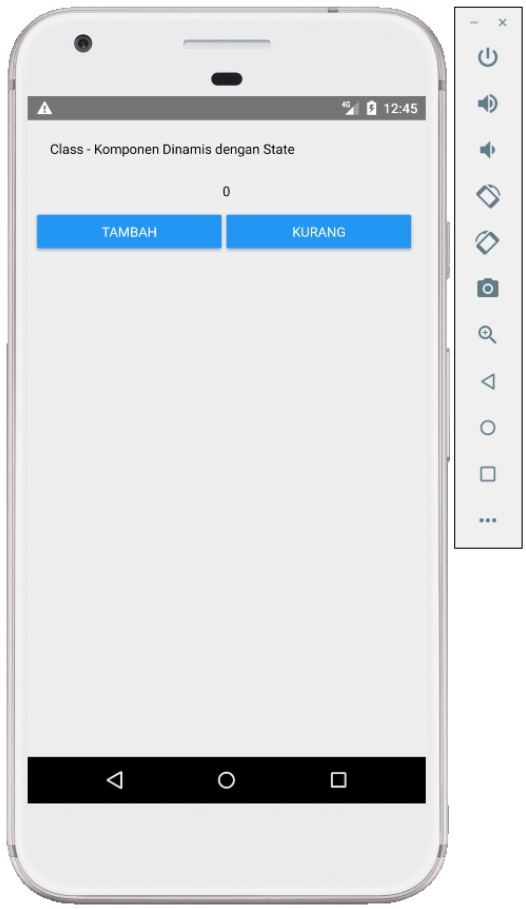


- **State dengan Class Komponen**

App.js

```
import React, {useState, Component} from 'react'
import { View, Text, Button, StyleSheet } from 'react-native'

const App = () => {
   return(
      <View style={{padding:10}}>
            <Text style={{alignItems:'center', padding:10}}> Class - Ko
mponen Dinamis dengan State </Text>
            <CounterClass />
      </View>
   );
}

class CounterClass extends Component{
   state = {
      number: 0,
   };
   render (){
      return(
         <View>
            <Text style={{textAlign:'center', padding:15}}>
               {this.state.number}
            </Text>
            <View style={{flexDirection:'row'}}>
               <View style={{marginRight:5, flex:1}}>
               <Button title='Tambah' onPress={() => this.setState({num
ber : this.state.number + 1})} />
               </View>
               <View style={{marginRight:5, flex:1}}>
               <Button title='Kurang' onPress={() => this.setState({num
ber : this.state.number - 1})} />
               </View>
            </View>
         </View>
      );
   }
}
export default App;
```

Output :



Berikut penjelasan umum dari State :

a) Inisialiasi State

Pada beberapa contoh inisialisasi di buat dengan menggunakan *constructor()*. Tapi saya lebih memilih style ini. karena lebih simple

b) Update State

Untuk merubah state gunakan perintah ***this.setState()***. Ketika state berubah secara otomatis component akan di render ulang. method disini menggunakan arrow function untuk menghindari problem *javascript bind*.

c) Read Component State

Untuk membaca state gunakan perintah ***this.state.keyName***

d) Call Method

Semua perubahan state dilakukan di dalam method **render()**

BASIC JAVASCRIPT

VARIABEL, TIPE DATA, CONDITION, LOOPING

Berikut adalah basic Javascript yang wajib dipahami dalam belajar React Native :

1. Variabel dan Tipe Data

Variabel adalah sebuah nama yang mewakili sebuah nilai. Variabel bisa diisi dengan berbagai macam nilai seperti string (teks), number (angka), objek, array, dan sebagainya.

Aturan penamaan variabel dalam JavaScript:

- Harus diawali dengan karakter (huruf atau baris bawah).
- Tidak boleh menggunakan spasi.
- Huruf Kapital dan kecil memiliki arti yang berbeda (case sensitive)
- Tidak boleh menggunakan kata-kata yang merupakan perintah dalam JavaScript.

Tipe data adalah jenis-jenis data yang bisa kita simpan di dalam variabel. Namun, Tidak seperti bahasa pemrograman lainnya, JavaScript tidak memiliki tipe data secara explisit. Tapi, JavaScript mempunyai tipe data implisit dalam pemrograman Javascript:

- String (teks)
- Integer atau Number (bilangan bulat)
- Boolean (True atau False)
- Object (Array)

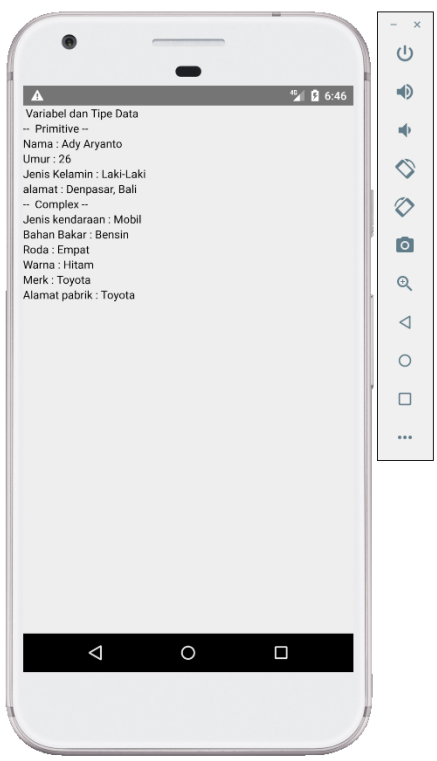
Dalam dekralasi variabel dalam javascript ada dua cara dengan menggunakan *const* dan *let*. Kemudian pendeklarasian tipe data pada javascript React Native dibedakan menjadi dua jenis yaitu *primitive* dan *complex* berikut adalah contohnya :

App.js

```
import React from 'react'
import { View, Text } from 'react-native'

const App = () =>{
   //Primitive
   const nama = 'Ady Aryanto';
   let usia = 26;
   const jenisKelamin = 'Laki-Laki';
   let alamat = 'Denpasar, Bali';
   //complex
   //array atau object
   const kendaraan = {
      jenis : 'Mobil',
      bahanBakar : 'Bensin',
      roda : 'Empat',
      warna : 'Hitam',
      merk : {
         pabrik : 'Toyota',
         alamat : 'Jakarta',
      }
   }
   return(
      <View>
         <Text> Variabel dan Tipe Data </Text>
         <Text> --  Primitive -- </Text>
         <Text>
            Nama : {nama}
         </Text>
         <Text>
            Umur : {usia}
         </Text>
         <Text>
            Jenis Kelamin : {jenisKelamin}
         </Text>
         <Text>
            alamat : {alamat}
```

```
        </Text>
        <Text>--  Complex --</Text>
        <Text>
           Jenis kendaraan : {kendaraan.jenis}
        </Text>
        <Text>
           Bahan Bakar : {kendaraan.bahanBakar}
        </Text>
        <Text>
           Roda : {kendaraan.roda}
        </Text>
        <Text>
           Warna : {kendaraan.warna}
        </Text>
        <Text>
           Merk : {kendaraan.merk.pabrik}
        </Text>
        <Text>
           Alamat pabrik : {kendaraan.merk.pabrik}
        </Text>
      </View>
   );
}
export default App;
```

Output :



2. Condition

Penggunaan Condition statement pada Javascript, sangat mirip dengan bahasa pemrograman C atau Java, dan algoritma pada umumnya. Berikut pembagian dari kedua statement tersebut,

- Condition
    - If-Else
    - Switch-case

Condition statement dibutuhkan ketika ingin menjalankan sebuah blok kode program pada kondisi tertentu. Condition statement berfungsi untuk mengontrol alur dari kode program yang telah kita buat berdasarkan kondisi-kondisi tertentu. Berikut adalah contoh program menghitung nilai mahasiswa dengan menggunakan sintaks *if-else*:

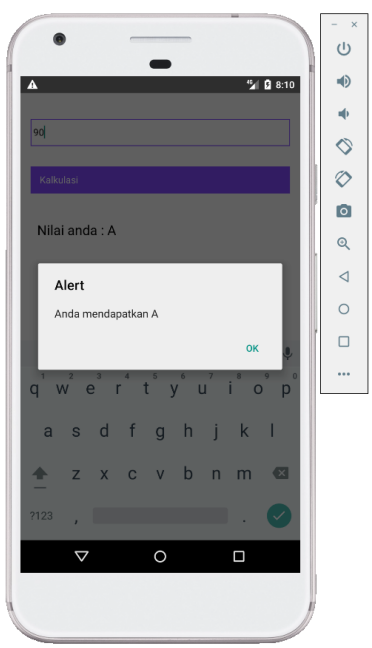
App.js

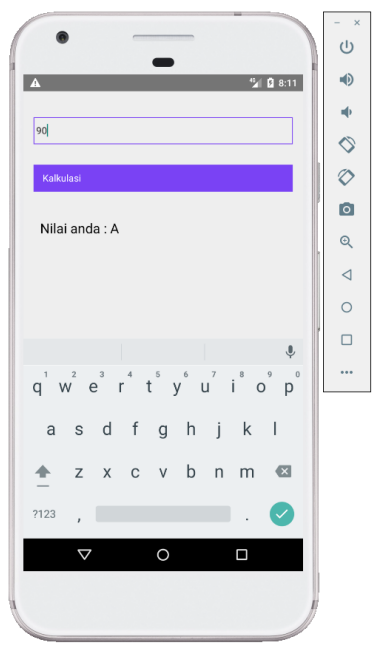
```
import React, { Component } from 'react'
import { View, Text, TouchableOpacity, TextInput, StyleSheet } from 'react-native'

class App extends Component {
   state = {
      nilai: '',
      huruf:'',
   }
   handleNilai= (text) => {
      this.setState({ nilai: text })
   }
   totalNilai = (nilai) => {
      if(nilai <= 50)
      {
         alert("Anda mendapatkan E");
         this.setState({ huruf: 'E' });
      }
      else if(nilai <= 60)
      {
```

```
      alert("Anda mendapatkan D");
      this.setState({ huruf: 'D' });
   }
   else if(nilai <= 70)
   {
      alert("Anda mendapatkan C");
      this.setState({ huruf: 'C' });
   }
   else if(nilai <= 80)
   {
      alert("Anda mendapatkan B");
      this.setState({ huruf: 'B' });
   }
   else if(nilai <= 85)
   {
      alert("Anda mendapatkan AB");
      this.setState({ huruf: 'AB' });
   }
   else if(nilai <= 100)
   {
      alert("Anda mendapatkan A");
      this.setState({ huruf: 'A' });
   }
   else
   {
      alert("Nilai yang anda masukan salah!");
      this.setState({ huruf: '-' });
   }
}

render() {
   return (
      <View style = {styles.container}>
         <TextInput style = {styles.input}
            underlineColorAndroid = "transparent"
            placeholder = "Nilai"
            placeholderTextColor = "#9a73ef"
            autoCapitalize = "none"
            onChangeText = {this.handleNilai}/>

         <TouchableOpacity
            style = {styles.submitButton}
            onPress = {
               () => this.totalNilai(this.state.nilai)
            }>
            <Text style = {styles.submitButtonText}> Kalkulasi </Text>
         </TouchableOpacity>
```

```
            <Text style={{padding:25, fontSize:20}}>Nilai anda : {this.state.huruf}</Text>
          </View>
        )
    }
}
export default App;

const styles = StyleSheet.create({
    container: {
        paddingTop: 23
    },
    input: {
        margin: 15,
        height: 40,
        borderColor: '#7a42f4',
        borderWidth: 1
    },
    submitButton: {
        backgroundColor: '#7a42f4',
        padding: 10,
        margin: 15,
        height: 40,
    },
    submitButtonText:{
        color: 'white'
    }
})
```

Output :





3. Looping

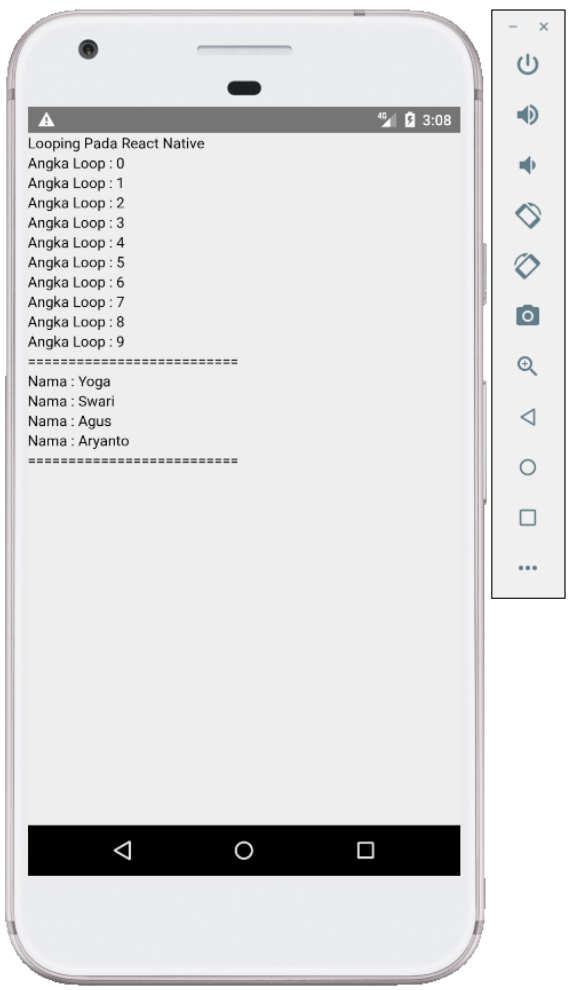
Loop atau perulangan adalah suatu cara untuk mengulang suatu statement sampai batas yang diinginkan. Pada umumnya didalam pemrograman looping selalu memilki nilai awal, nilai akhir dan proses increment/decrement. Berikut adalah contoh sintaks looping pada react native.

App.js

```
import React from 'react'
import {View, Text} from 'react-native';

export default class App extends React.Component {


   render() {
      const myloop = [];
      const identitas = [];
      const nama = ['Yoga', 'Swari', 'Agus','Aryanto'];


      for (let i = 0; i < 10; i++) {
         myloop.push(
            <View key={i}>
               <Text>Angka Loop : {i}</Text>
            </View>
         );
      }


      for (let i = 0; i < 4; i++) {
         identitas.push(
            <View key={i}>
               <Text>Nama : {nama[i]}</Text>
            </View>
         );
      }


      return (
         <View >
```

```
            <Text >Looping Pada React Native</Text>
            {myloop}
            <Text>==========================</Text>
            {identitas}
            <Text>==========================</Text>
          </View>
      );
    }
}
```

Output :



REACT NATIVE – HTTP, API, JSON

HTTP (Hypertext Transfer Protocol) adalah protokol jaringan lapisan aplikasi (*application layer*) yang dikembangkan untuk membantu proses transfer data antara client dan server. Di dalam transfer ini, client melakukan permintaan dengan mengakses alamat IP atau domain (URL). Kemudian server mengelola permintaan tersebut sesuai dengan kode yang dimasukkan. Fungsi HTTP yaitu mengatur metode dan bagaimana data ditransmisikan. Pada React Native metode ini dibagi menjadi lima bagian yaitu :

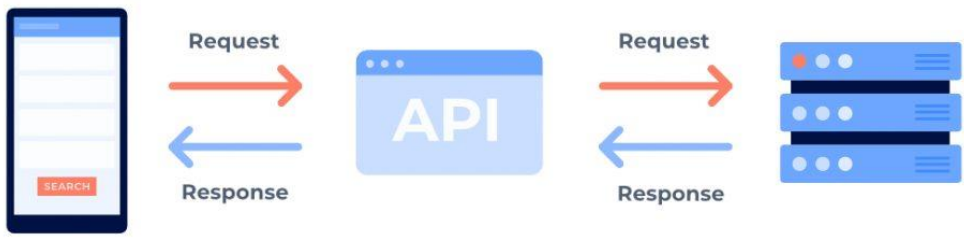
a. **POST**mengirimkan data ke server, biasanya digunakan untuk menambah data.
b. **GET** Mengambil data dari server
c. **PUT** Menambah atau menganti data dengan yang baru
d. **PATCH** Merubah data dengan yang baru
e. **DELETE** Menghapus data

API atau Application Programming Interface adalah sebuah interface yang dapat menghubungkan aplikasi satu dengan aplikasi lainnya. Pada penggunaannya, terdapat empat jenis API sesuai dengan hak aksesnya yaitu, Public API, Private API, Partner API, Composite API.

Manfaat yang didapatkan dalam pengembangan Aplikasi adalah :

a. Memudahkan Membangun Aplikasi yang Fungsional
b. Pengembangan Aplikasi Menjadi Lebih Efisien
c. Meringankan Beban Server

Cara kerja API seperti berikut :



a. Aplikasi mengakses API
b. API melakukan request ke server
c. Server memberi respon ke API
d. API menyampaikan respon ke aplikasi

JavaScript object notation **atau JSON** adalah format yang digunakan untuk menyimpan dan mentransfer data. JSON memiliki struktur data yang sederhana dan mudah dipahami. Itulah mengapa JSON sering digunakan pada API.

JSON sendiri terdiri dari dua struktur, yaitu:

- Kumpulan *value* yang saling berpasangan.
- Daftar *value* yang berurutan, seperti array.

Sintax JSON :

Key	Value
{"city":"New York", "country":"United States "}	

JSON selalu dibuka dan ditutup dengan tanda {} atau kurung kurawal. Syntax-nya terdiri dari dua elemen, yaitu **key** dan **value**. Keduanya dipisahkan oleh titik dua agar jelas. Apabila ada lebih dari satu pasang key dan value, perlu memisahkannya dengan tanda koma yang diikuti spasi. Ini dapat dilihat pada contoh tabel di atas. Ada enam jenis data yang dapat digunakan sebagai *value* JSON, yaitu

a. *String*
b. *Object*
c. *Array*
d. *Boolean*
e. *Number*
f. *Null*

Berikut adalah contoh implementasi penggunan HTTP, API dan JSON. Pada program ini menggunakan dummy API yang sudah ada di internet. Simulasi program ini menampilkan data pada halaman *console debug* beserta halaman aplikasi mobilenya. Berikut penyedia API dummy yang dapat digunakan :

- https://jsonplaceholder.typicode.com
- https://reqres.in/
- https://fakejson.com
- https://mocki.io/fake-json-api

Pada contoh ini akan digunakan dummy API dari https://reqres.in kemudian pada React Native digunakan *fetch,* berikut adalah sintaknya :

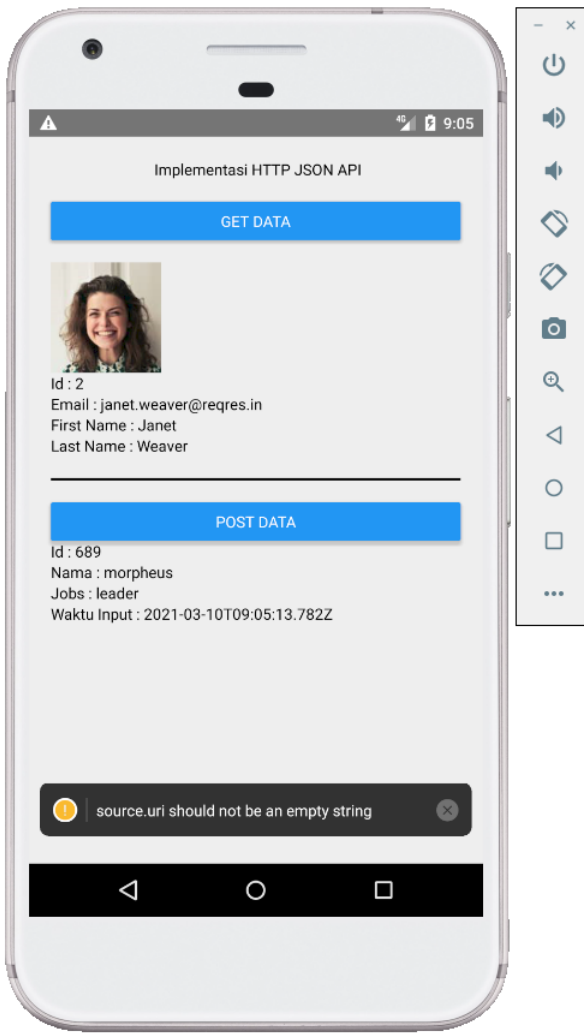
App.js

```
import React, {useState} from 'react'
import { View, Text, StyleSheet, Button, Image } from 'react-native'

const App = () => {
   const [dataUser, setUser] = useState({
      id:'',
      email:'',
      first_name:'',
      last_name:'',
      avatar:'',
   });
   const [dataJob, setJob] = useState({
      id:'',
      name:'',
      job:'',
      createdAt:'',
   });

   const getData = () => {
      //Call API Method GET
      fetch('https://reqres.in/api/users/2', {
```

```
        method:'GET'
      })
      .then(response => response.json())
      .then(json => {
        console.log(json); //Data ditampilkan pada console log
        setUser(json.data); //State
      })
   }

   const postData = () => {
      //Call API Method POST
      const formData = {
        name: "morpheus",
        job: "leader"
     }
      fetch('https://reqres.in/api/users', {
        method:'POST',
        headers:{
          'Content-Type' : 'application/json'
        },
        body: JSON.stringify(formData)
      })
      .then(response => response.json())
      .then(json =>{
        console.log(json) //Data ditampilkan pada console log
        setJob(json); //State
      })
   }

   return (
      <View style={styles.container}>
        <Text style={styles.textCenter}> Implementasi HTTP JSON API</T
ext>
        <Button title='GET DATA' onPress={getData}></Button>
        <Image source={{uri:dataUser.avatar}} style={{width:100, heigh
t:100, marginTop:20}} />
        <Text>Id : {dataUser.id}</Text>
        <Text>Email : {dataUser.email}</Text>
        <Text>First Name : {dataUser.first_name}</Text>
        <Text>Last Name : {dataUser.last_name}</Text>

        <View style={{height:2, backgroundColor:'black', marginVertica
l:20}}></View>

        <Button title='POST DATA' onPress={postData}></Button>
        <Text>Id : {dataJob.id}</Text>
        <Text>Nama : {dataJob.name}</Text>
        <Text>Jobs : {dataJob.job}</Text>
```

```
            <Text>Waktu Input : {dataJob.createdAt}</Text>
        </View>
    )
}

const styles = StyleSheet.create({
    container:{
        padding:20,
    },
    textCenter:{
        textAlign:'center',
        marginBottom: 20
    }
});

export default App;
```

Output :



REACT NATIVE – CRUD
(CREATE READ UPDATE DELETE)

Pada penjelasan CRUD ini mengambil contoh mengelola data mahasiswa. Contoh implementasinya menggunakan server lokal dengan menginstall XAMPP. Database yang digunakan MySQL, kemudian dalam database dibuat sebuah tabel dengan tb_mhs dengan dengan struktur seperti berikut :

Tabel Stuktur tb_mhs

No	Key	Field	Tipe Data	Panjang	Keterangan
1	PK	nim	Varchar	20	*Primary key*
2		nama	Varchar	100	
3		nama	Varchar	100	
4		telepon	Varchar	100	

Setelah selesai membuat database selanjutnya membuat file koneksi yang akan terus digunakan pada webservice CRUD ini. Berikut sintaks dari koneksi :

Silahkan sesuaikan nama server, username, password dan database.

```php
 <?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
  die("Connection failed: " . mysqli_connect_error());
}

?>
```

Berikut akan dimulai mencontohkan proses CRUD, disini akan lebih fokus untuk proses CRUD dengan membuat UI (*User Interface*) yang sederhana. Berikut tahapan prosesnya :

A. CREATE (Tambah)

Membuat sintak API webservice dengan bahasa pemrograman PHP

Tambah.php

```php
<?php
    require_once("koneksi.php");

    $_POST = json_decode(file_get_contents('php://input'), true);
    if(!empty($_POST["nim"]) and isset($_POST["nim"]))
    {
        $nim = $_POST["nim"];
        $nama = $_POST["nama"];
        $jurusan = $_POST["jurusan"];
        $telepon = $_POST["telepon"];

        $sql = "INSERT INTO tb_mhs (nim, nama, jurusan,telepon) VALUES(
'".$nim."','".$nama."','".$jurusan."','".$telepon."')";
        $result = mysqli_query($conn, $sql);
```

```
        if($result)
        {
            $data = array(
                'status' => 'Sukses'
            );
            echo json_encode($data);
        }
        else
        {
            $data = array(
                'status' => 'Gagal'
            );
            echo json_encode($data);
        }
    }
    else{
        $data = array(
            'status' => 'error'
        );
        echo json_encode($data);
    }
?>
```

B. READ (Tampil)

Tampil.php

```
<?php
    require_once("koneksi.php");
    $sql = "SELECT * FROM tb_mhs";
    $myArray = array();
    $result = mysqli_query($conn, $sql);
    while($row = mysqli_fetch_array($result))
    {
        $myArray[] = $row;
    }
    echo json_encode($myArray);

?>
```

C. UPDATE (Edit)

Update.php

```php
<?php
    require_once("koneksi.php");

    $_POST = json_decode(file_get_contents('php://input'), true);
    if(!empty($_POST["nim"]) and isset($_POST["nim"]))
    {
        $nim = $_POST["nim"];
        $nama = $_POST["nama"];
        $jurusan = $_POST["jurusan"];
        $telepon = $_POST["telepon"];

        $sql = " UPDATE tb_mhs SET nama = '".$nama."',jurusan = '".$jur
usan."',telepon = '".$telepon."'WHERE nim = '".$nim."'";
        $result = mysqli_query($conn, $sql);

        if($result)
        {
            $data = array(
                'status' => 'Sukses'
            );
            echo json_encode($data);
        }
        else
        {
            $data = array(
                'status' => 'Gagal'
            );
            echo json_encode($sql);
        }


    }
    else{
        $data = array(
            'status' => 'error'
        );
        echo json_encode($data);
    }

?>
```

D. DELETE (Hapus)

Hapus.php

```php
<?php
    require_once("koneksi.php");

    $_POST = json_decode(file_get_contents('php://input'), true);
    if(!empty($_POST["nim"]) and isset($_POST["nim"]))
    {
        $nim = $_POST["nim"];


        $sql = "DELETE FROM tb_mhs WHERE nim = '".$nim."'";
        $result = mysqli_query($conn, $sql);

        if($result)
        {
            $data = array(
                'status' => 'Sukses'
            );
            echo json_encode($data);
        }
        else
        {
            $data = array(
                'status' => 'Gagal'
            );
            echo json_encode($sql);
        }


    }
    else{
        $data = array(
            'status' => 'error'
        );
        echo json_encode($data);
    }

?>
```

Setelah selesai membuat Webservice, selanjutnya membuat sintak di Mobile, membuat form inputan field nim, nama, jurusan dan telepon serta sebuah button. Berikut seluruh sintak program untuk proses CRUD di Mobile.

app.js:

Sesuaikan alamat IP dari API nya dan contoh disini menggunakan fungsi *fetch.*

```
import React, {useState, useEffect} from 'react'
import { View, Text,StyleSheet, TextInput, Button, Alert, ScrollView, TouchableOpacity } from
 'react-native'

const App = () => {
   return (
      <View>
         <FormBiodata />
      </View>
   )
}

const FormBiodata = () => {
   const [nim, setNim] = useState("");
   const [nama, setNama] = useState("");
   const [jurusan, setJurusan] = useState("");
   const [telepon, setTelepon] = useState("");
   const [mahasiswa, setMahasiswa] = useState([]);
   const [button, setButton] = useState("Simpan");

   const Submit = () => {
      const formData = {
         nim : nim,
         nama : nama,
         jurusan : jurusan,
         telepon : telepon,
      }
      console.log(formData);

      /// PROSES CREATE (TAMBAH)
      if(button === 'Simpan')
      {
```

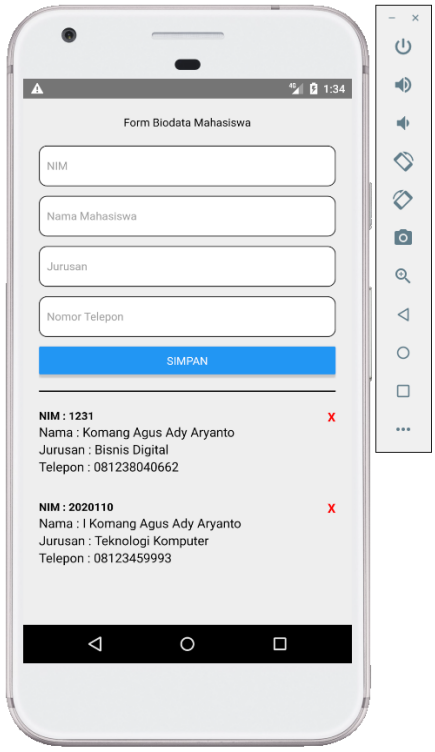
```
        fetch('http://192.168.1.8/reactnative_latihan/tambah.php', {
            method:'POST',
            headers:{
                'Content-Type' : 'application/json'
            },
            body: JSON.stringify(formData)
        })
        .then(response => response.json())
        .then(json =>{
            if(json.status === 'Sukses')
            {
                Alert.alert("Data berhasil disimpan");
                Tampil();
                BersihForm();
            }
            else
            {
                Alert.alert("Data gagal disimpan");
            }
        }) //Data ditampilkan pada console log
    }
    /// PROSES UPDATE (EDIT)
    else if(button === 'Update')
    {
        fetch('http://192.168.1.8/reactnative_latihan/update.php', {
            method:'POST',
            headers:{
                'Content-Type' : 'application/json'
            },
            body: JSON.stringify(formData)
        })
        .then(response => response.json())
        .then(json =>{
            console.log(json);
            if(json.status === 'Sukses')
            {
                Alert.alert("Data berhasil diupdate");
                Tampil();
                BersihForm();
            }
            else
            {
                Alert.alert("Data gagal diupdate");
            }
        }) //Data ditampilkan pada console log
    }
}
```

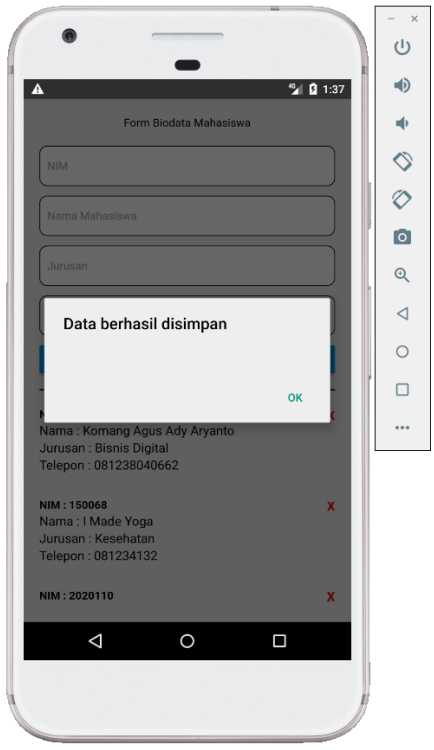
```
useEffect(() => {
   Tampil();
}, [])

/// PROSES READ (TAMPIL)
const Tampil = () =>{
   fetch('http://192.168.1.8/reactnative_latihan/tampil.php', {
      method:'GET'
   })
   .then(response => response.json())
   .then(json => {
      console.log(json)
      setMahasiswa(json)
   }) //Data ditampilkan pada console log

}

const SelectMhs = (mhs) => {
   console.log(mhs);
   setNim(mhs.nim);
   setNama(mhs.nama);
   setJurusan(mhs.jurusan);
   setTelepon(mhs.telepon);
   setButton("Update");
}

/// PROSES DELETE (HAPUS)
const HapusMhs = (mhs) => {
   console.log(mhs);
   const formData = {
      nim : mhs.nim,
   }

   fetch('http://192.168.1.8/reactnative_latihan/hapus.php', {
      method:'POST',
      headers:{
         'Content-Type' : 'application/json'
      },
      body: JSON.stringify(formData)
   })
   .then(response => response.json())
   .then(json =>{
      console.log(json);
      if(json.status === 'Sukses')
      {
         Alert.alert("Data berhasil dihapus");
         Tampil();
      }
```

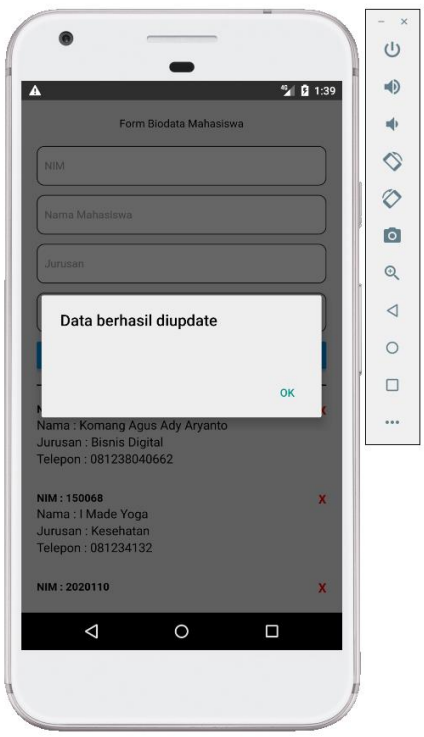
```
        else
        {
           Alert.alert("Data gagal dihapus");
        }
     }) //Data ditampilkan pada console log
  }

  const BersihForm = () =>{
     setNim("");
     setNama("");
     setJurusan("");
     setTelepon("");
     setButton("Simpan");
  }


  return(
     <View style = {styles.container}>
        <ScrollView>
           <Text style = {styles.textTitle}>Form Biodata Mahasiswa</Text>
           <TextInput placeholder="NIM" style={styles.formInput} value={nim} onChangeText={(
value) => setNim(value)} />
           <TextInput placeholder="Nama Mahasiswa" style={styles.formInput} value={nama} onC
hangeText={(value) => setNama(value)}/>
           <TextInput placeholder="Jurusan" style={styles.formInput} value={jurusan} onChang
eText={(value) => setJurusan(value)} />
           <TextInput placeholder="Nomor Telepon"style={styles.formInput} value={telepon} on
ChangeText={(value) => setTelepon(value)} />
           <Button title={button} onPress={Submit} />
           <View style={styles.line} />
           {mahasiswa.map(mhs => {
              return (
                 <ListBiodata
                    nim={mhs.nim}
                    nama={mhs.nama}
                    jurusan={mhs.jurusan}
                    telepon={mhs.telepon}
                    selectmhs={() => SelectMhs(mhs)}
                    hapus={() => Alert.alert(
                    'Peringatan',
                    'Apakah yakin dihapus?',
                    [
                       {
                          text: 'Tidak',
                          onPress: () => Tampil()
                       },
                       {
                          text: 'Ya',
```

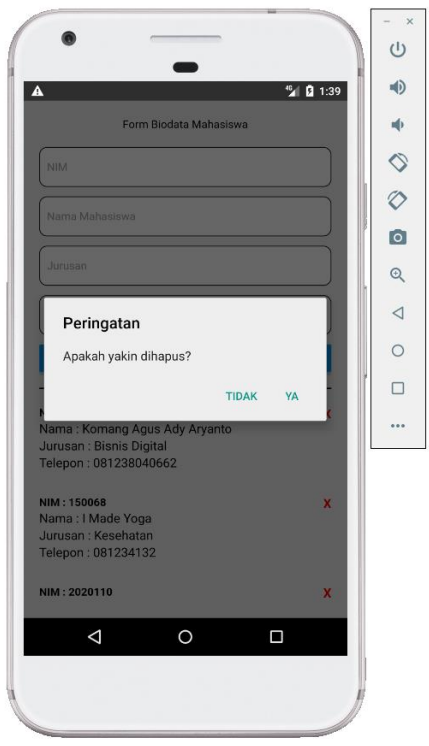
```
                        onPress: () => HapusMhs(mhs)
                    }
                ]

                )}>

            </ListBiodata>
        )
    })}
    </ScrollView >
  </View>
 );
}
const ListBiodata = ({nim, nama, jurusan, telepon, selectmhs, hapus}) =>
{
 return(
  <View style={{flexDirection:'row', marginBottom: 30}}>
      <View style={{flex:1}}>
        <TouchableOpacity onPress={selectmhs}>
          <Text style={styles.textMedium, {fontWeight:'bold'}} >NIM : {nim}</Text>
        </TouchableOpacity>
        <Text style={styles.textMedium}>Nama : {nama}</Text>
        <Text style={styles.textMedium}>Jurusan : {jurusan}</Text>
        <Text style={styles.textMedium}>Telepon : {telepon}</Text>
      </View>
      <TouchableOpacity onPress={hapus}>
        <Text style={{fontSize:16, fontWeight:'bold', color:'red'}}>X</Text>
      </TouchableOpacity>
  </View>
 )
}
const styles =  StyleSheet.create({
 container : {padding : 20},
 textTitle : {textAlign : 'center', marginBottom : 20},
 line : {height : 2, backgroundColor :'black', marginVertical: 20},
 formInput :{borderWidth:1, marginBottom:12, borderRadius:10, paddingHorizontal:10, backgro
undColor:'white'},
 textMedium : {fontSize:16}
})

export default App
```

Output :


1:34
Form Biodata Mahasiswa
NIM
Nama Mahasiswa
Jurusan
Nomor Telepon
SIMPAN
NIM : 1231
Nama : Komang Agus Ady Aryanto
Jurusan : Bisnis Digital
Telepon : 081238040662
NIM : 2020110
Nama : I Komang Agus Ady Aryanto
Jurusan : Teknologi Komputer
Telepon : 08123459993


1:37
Form Biodata Mahasiswa
Data berhasil disimpan
OK
Nama : Komang Agus Ady Aryanto
Jurusan : Bisnis Digital
Telepon : 081238040662
NIM : 150068
Nama : I Made Yoga
Jurusan : Kesehatan
Telepon : 081234132
NIM : 2020110


1:39
Form Biodata Mahasiswa
Data berhasil diupdate
OK
Nama : Komang Agus Ady Aryanto
Jurusan : Bisnis Digital
Telepon : 081238040662
NIM : 150068
Nama : I Made Yoga
Jurusan : Kesehatan
Telepon : 081234132
NIM : 2020110


1:39
Form Biodata Mahasiswa
Peringatan
Apakah yakin dihapus?
TIDAK
YA
Nama : Komang Agus Ady Aryanto
Jurusan : Bisnis Digital
Telepon : 081238040662
NIM : 150068
Nama : I Made Yoga
Jurusan : Kesehatan
Telepon : 081234132
NIM : 2020110

Dalam mendistribusikan aplikasi Android melalui Google Play Store, aplikasi perlu *Signed Key* rilis yang digunakan untuk pembaruan (update) kedepannya. Berikut panduan proses secara singkat langkah-langkah yang diperlukan untuk mengemas bundel JavaScript agar menjadi file yang dapat di upload di playstore.

Generating an upload key :

```
keytool -genkeypair -v -storetype PKCS12 -keystore my-upload-key.keystore -alias my-key-alias -keyalg RSA -keysize 2048 -validity 10000
```

Kemudian identitas untuk key store nya seperti nama depan, nama belakang, key dan sebagainya. Pastikan semua terisi dengan benar.

Setting up Gradle variables :

1. Setelah berhasil membuat key storenya silahkan buka folder projectnya dan pindahkan **my-upload-key.keystore** ke dalam folder android/app.
2. Edit file ~/.gradle/gradle.properties atau android/gradle.properties, dan tambahkan (replace ***** dengan keystore password, alias dan key password yang benar).

```
MYAPP_UPLOAD_STORE_FILE=my-upload-key.keystore
MYAPP_UPLOAD_KEY_ALIAS=my-key-alias
MYAPP_UPLOAD_STORE_PASSWORD=*****
MYAPP_UPLOAD_KEY_PASSWORD=*****
```

Sesusaikan password dan namanya dengan key store yang sudah dibuat sebelumnya.

Menambahkan konfigurasi ke Gradle aplikasi :

Edit file android/app/build.gradle di dalam folder project dan tambahkan *signing config* dan *build types*.

```
...
android {
    ...
    defaultConfig { ... }
    signingConfigs {
        release {
            if (project.hasProperty('MYAPP_UPLOAD_STORE_FILE')) {
                storeFile file(MYAPP_UPLOAD_STORE_FILE)
                storePassword MYAPP_UPLOAD_STORE_PASSWORD
                keyAlias MYAPP_UPLOAD_KEY_ALIAS
                keyPassword MYAPP_UPLOAD_KEY_PASSWORD
            }
        }
    }
    buildTypes {
        release {
            ...
            signingConfig signingConfigs.release
        }
    }
}
...
```

Generating the release AAB :

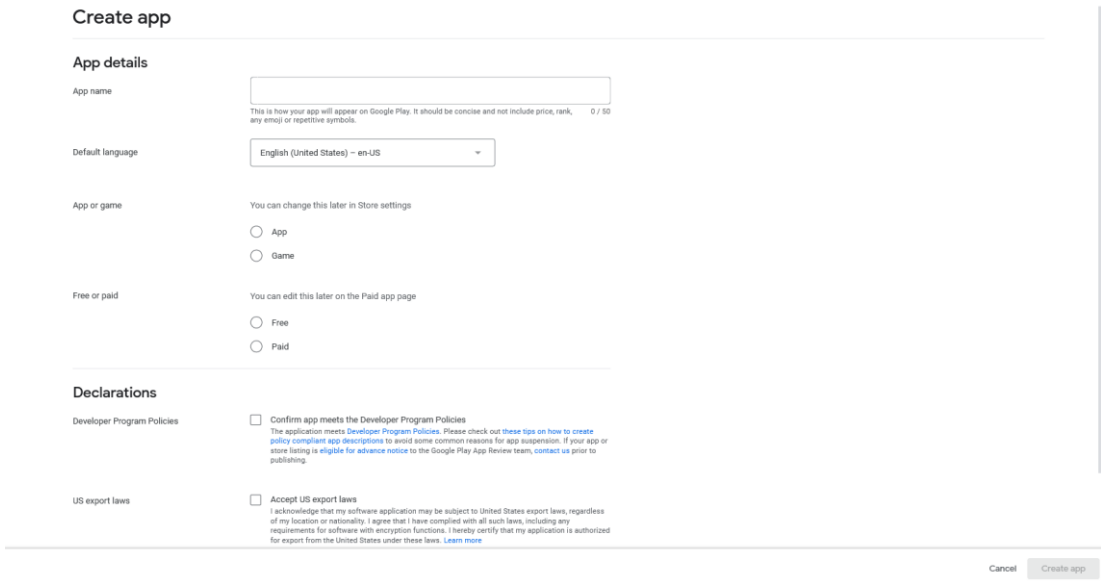
Ketikan perintah pada cmd didalam project :

```
cd android
./gradlew bundleRelease
```

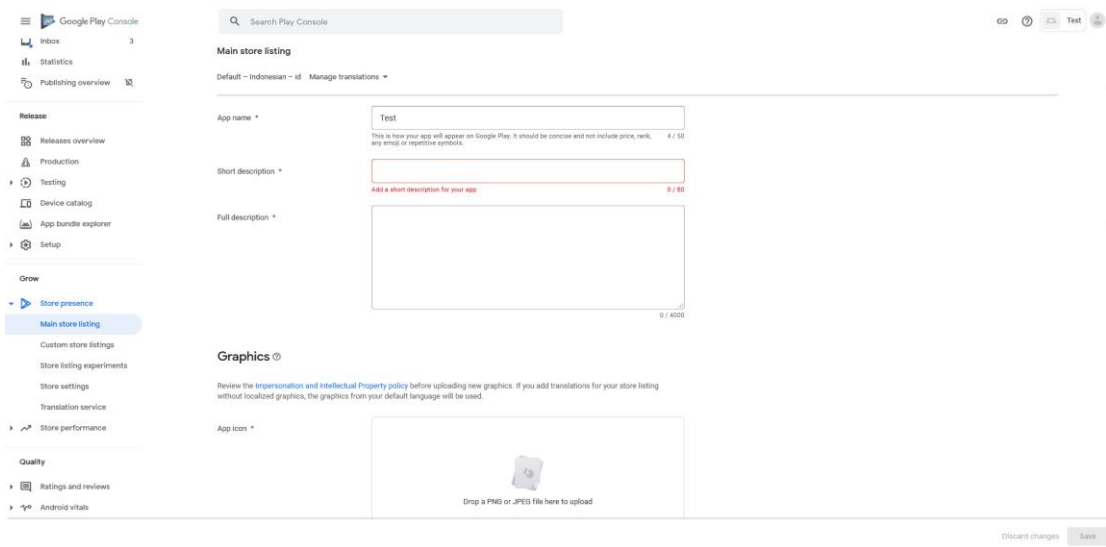
Hasil generate akan mendapat file *AAB* yang dapat dilihat pada *./android\app\build\outputs\bundle\release.* Kemudian file ini dapat diupload ke dalam playstore. Untuk membuat akun playstore pertama kali wajib melakukan pembayaran. Jika sudah berhasil membuat akun playstore nya, Kemudian upload aplikasi AAB.

Buka link https://play.google.com/console

Pilih **Create App**, dan masukan detail aplikasi pada form yang diminta :

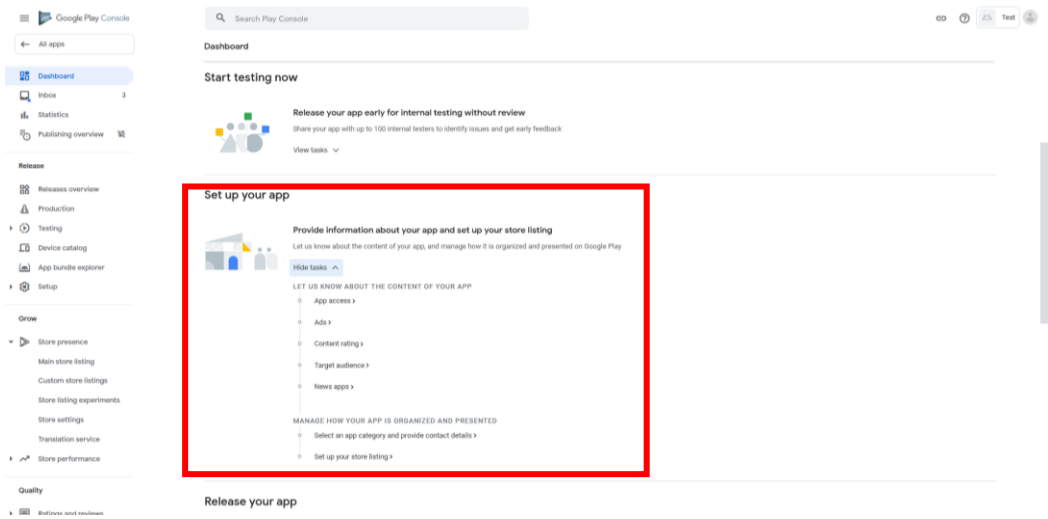


Pilih Menu **Main store listing**, lalu masukan Data yang diminta di bagian store listing. Seperti icon, foto-foto aplikasi dan informasi detail mengenai aplikasi android nya.

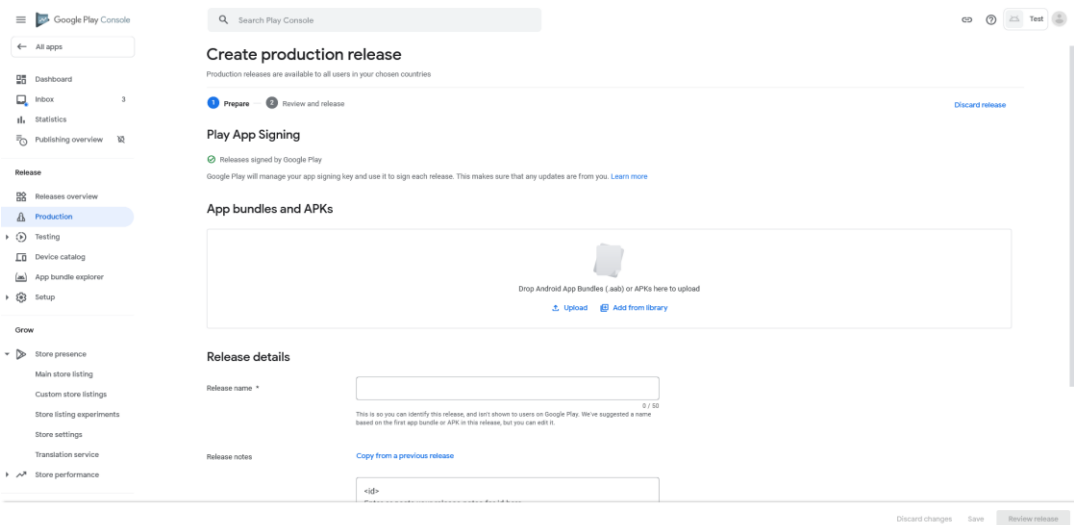


Berikutnya, isi data konten rating.

Pilh Menu **Dashboard -> Set Up Your App**



Setelah itu ke menu **Production, Create new release** dan upload **file AAB** dan isi data sesuai dengan form yang diminta.



Jika sudah upload, pilih menu review untuk mengecek kembali error dari proses production, apabila tidak ada error bisa langsung lakukan **peluncuran production**. Berikutnya, menunggu proses verifikasi pihak Google. Jika sudah selesai verifikasi maka proses publikasi ke playstore selesai.

REACT NATIVE – SOAL LATIHAN

Membuat aplikasi pencatatan data mahasiswa, berikut koding programnya : https://s.id/zbGP8