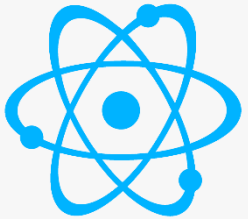
inixindo Surabaya

Workshop React Native

"Membangun Aplikasi Mobile Android / iOS
dengan Framework React Native"

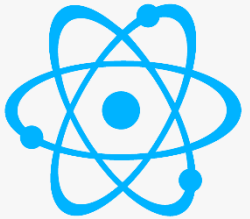
Kebutuhan Sistem



Laptop / Komputer

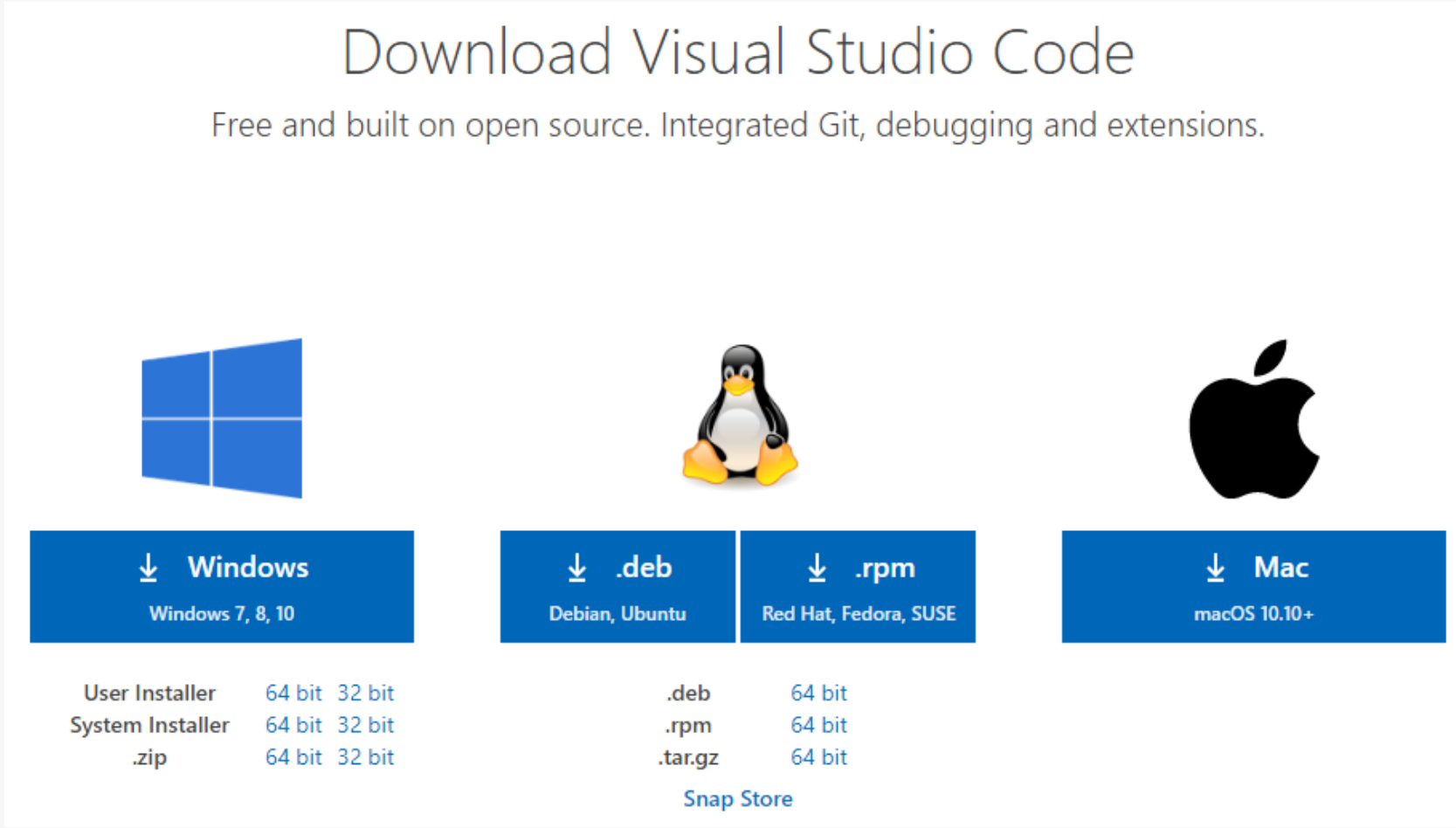
- Processor Core-i3/i5/i7
- Memori RAM 8GB
- Windows: Android Studio, SDK, AVD (Emulator)
- Mac OSX: XCode, iOS Emulator
- Node.js terbaru (NPM atau Yarn)
- Visual Studio Code sebagai text editor
- Koneksi Internet yang stabil

Agenda

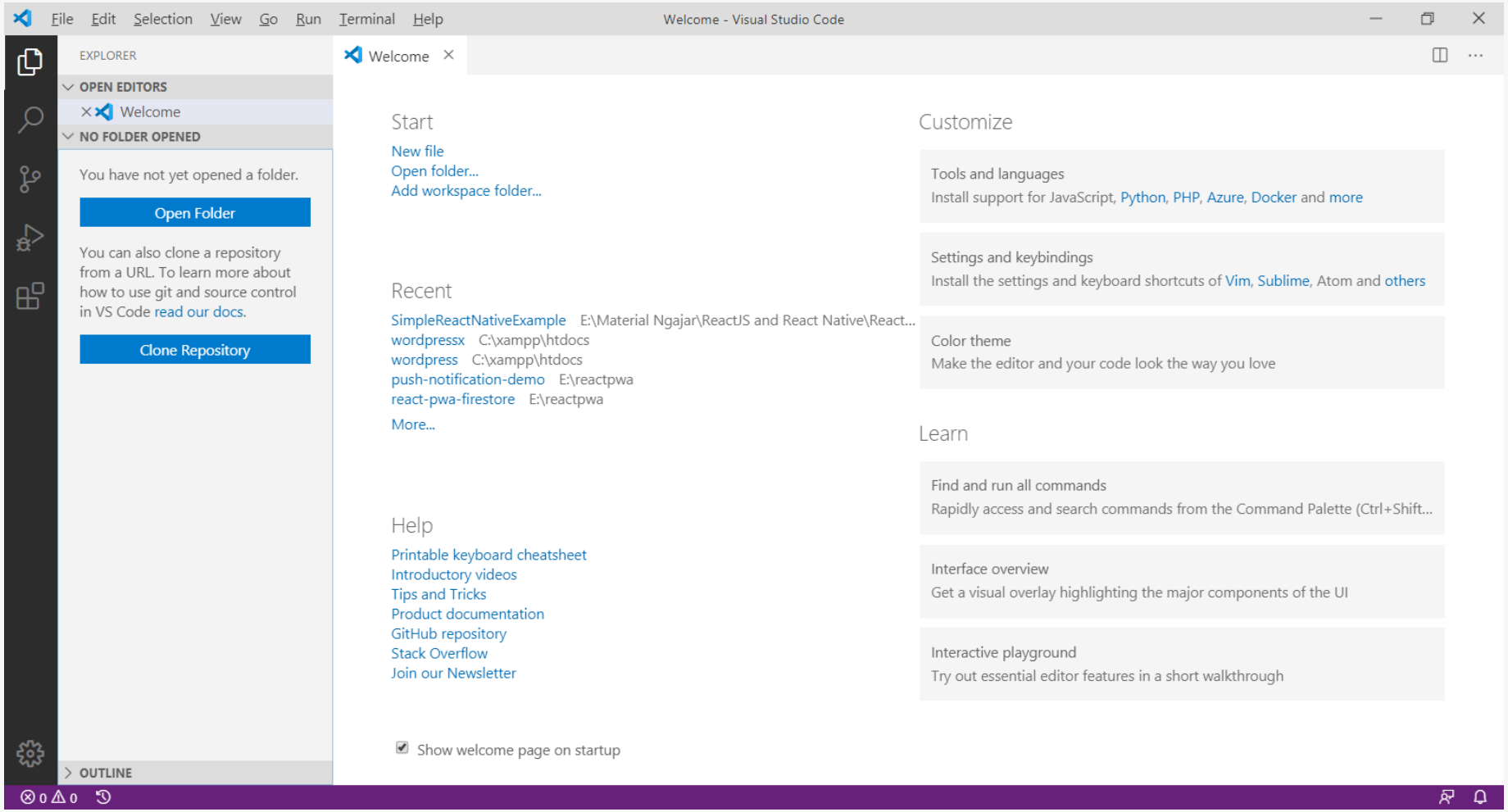


- Instalasi *React App Creator*
- Mengunduh dan Mengatur Ekstensi *Visual Studio Code*
- Membuat Aplikasi dan Menjalankannya di *Emulator* atau device
- Mengidentifikasi dan Mendeteksi Kesalahan di *React Native*
- Membedakan *State* dan *Props*
- Menggunakan *React Navigation*
- Studi Kasus (Menampilkan data dari Web API)
  https://github.com/openfootball/world-cup.json

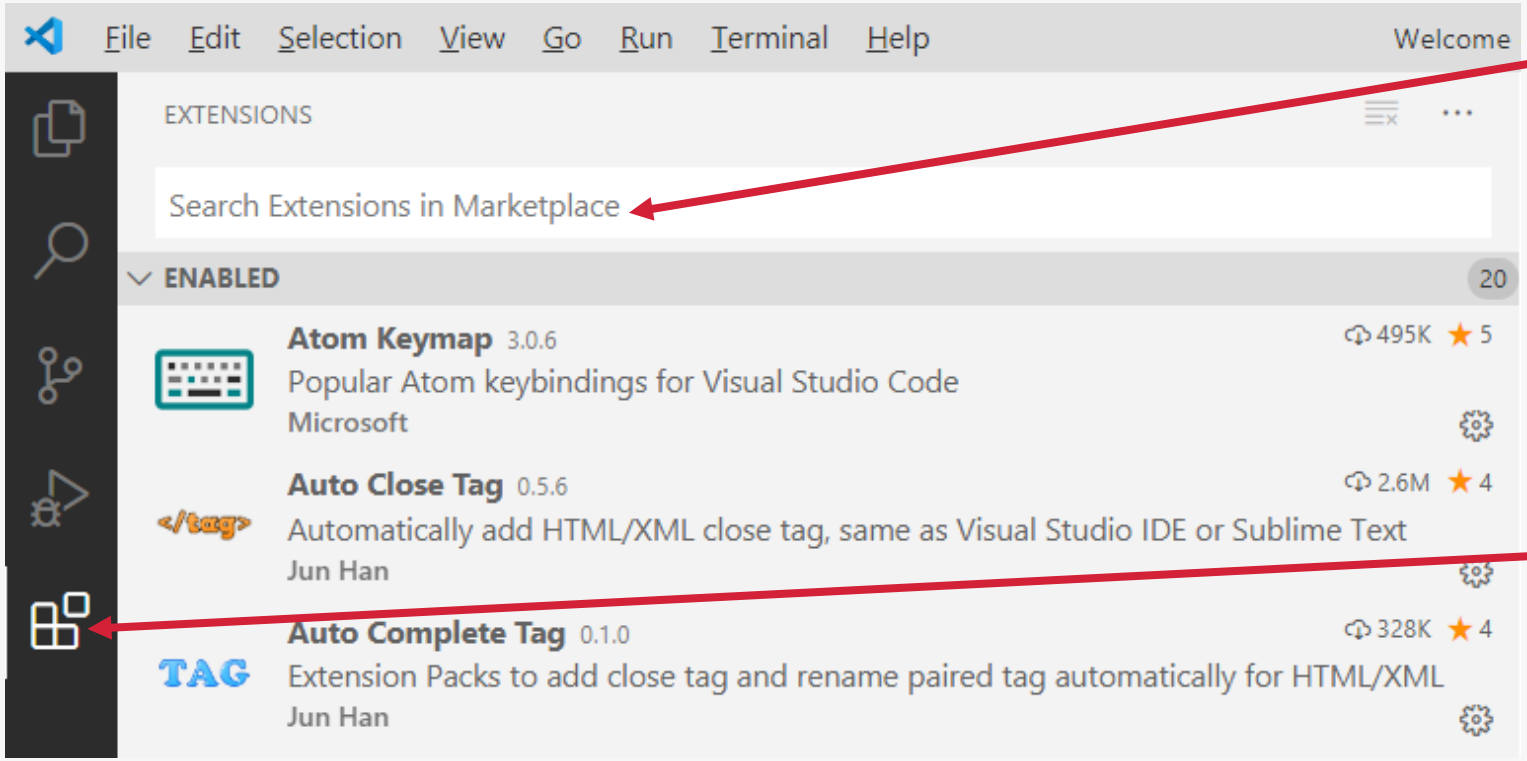
Visual Studio Code
https://code.visualstudio.com/Download



Visual Studio Code (2)



Mengatur Ekstensi Visual Studio Code



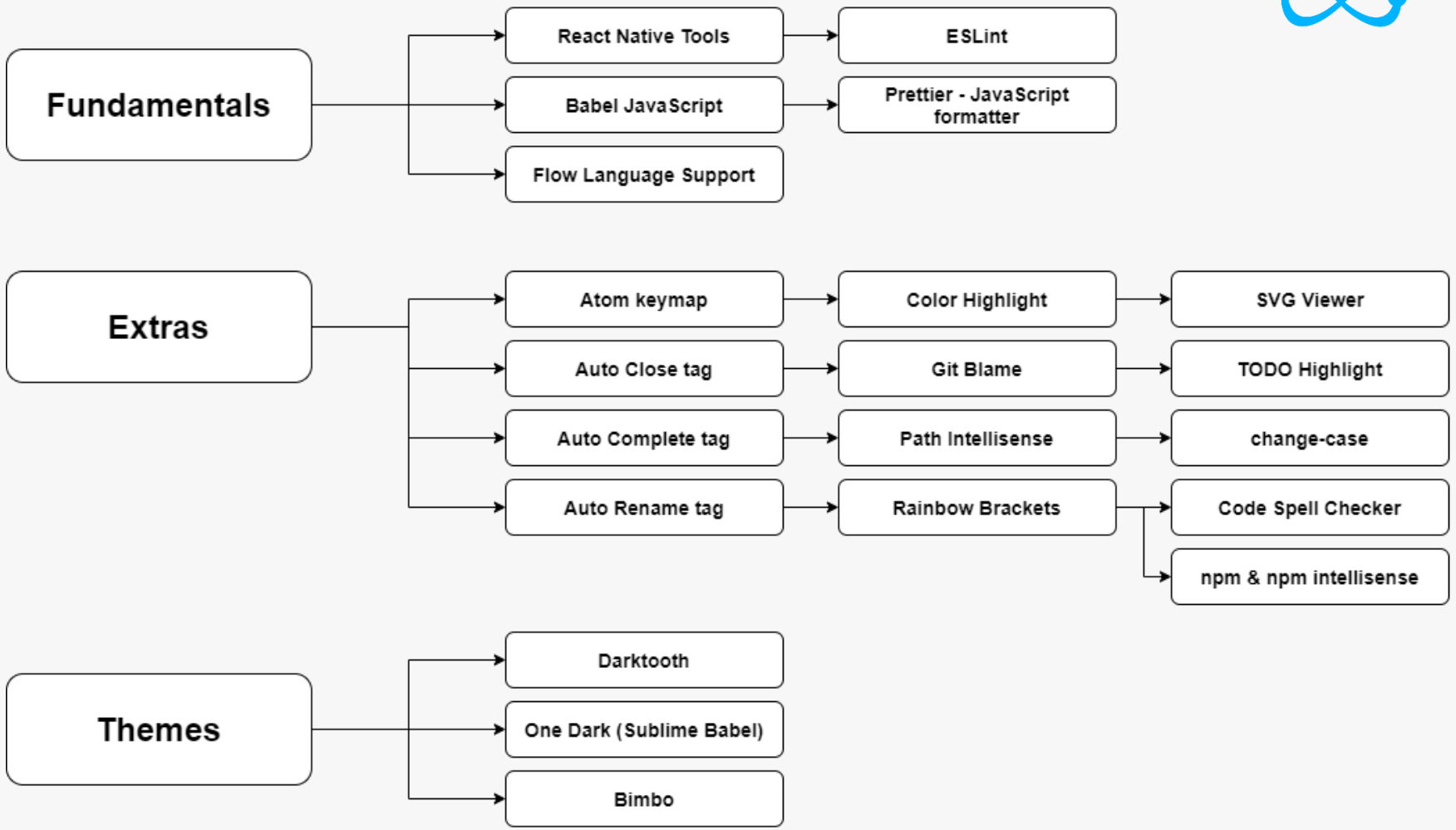
2 Cari ekstensi dengan mengetikkan nama ekstensi pada bagian ini

1 Klik pada menu Extension (Ctrl + Shift + X) untuk membuka dan mengaktifkan ekstensi

Ekstensi yang diaktifkan

Cari dan unduh semua ekstensi seperti yang tertera pada bagan disamping ini:

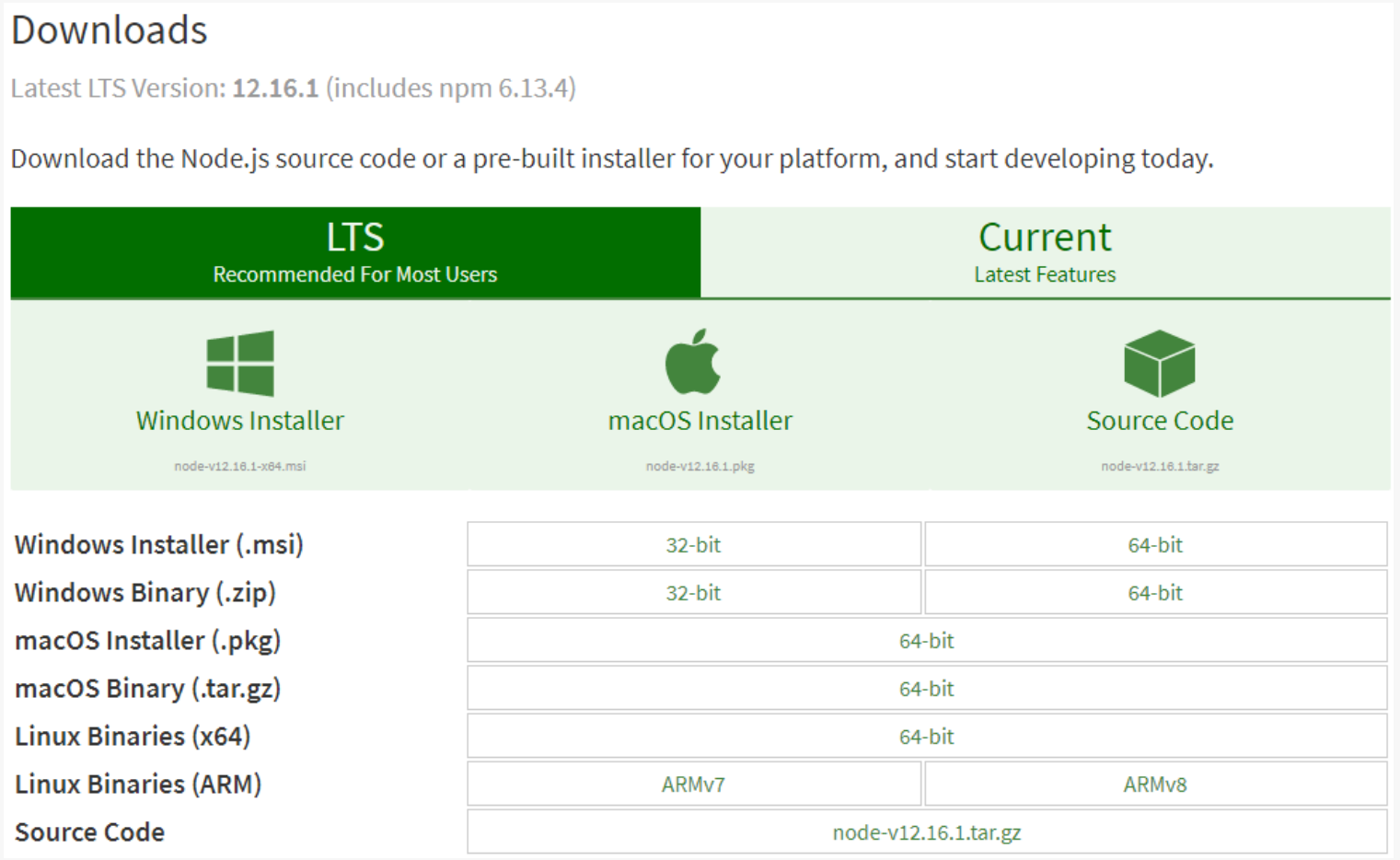
Fundamentals harus diunduh, Extras dan Themes bersifat tidak harus diunduh hanya sebagai tambahan.



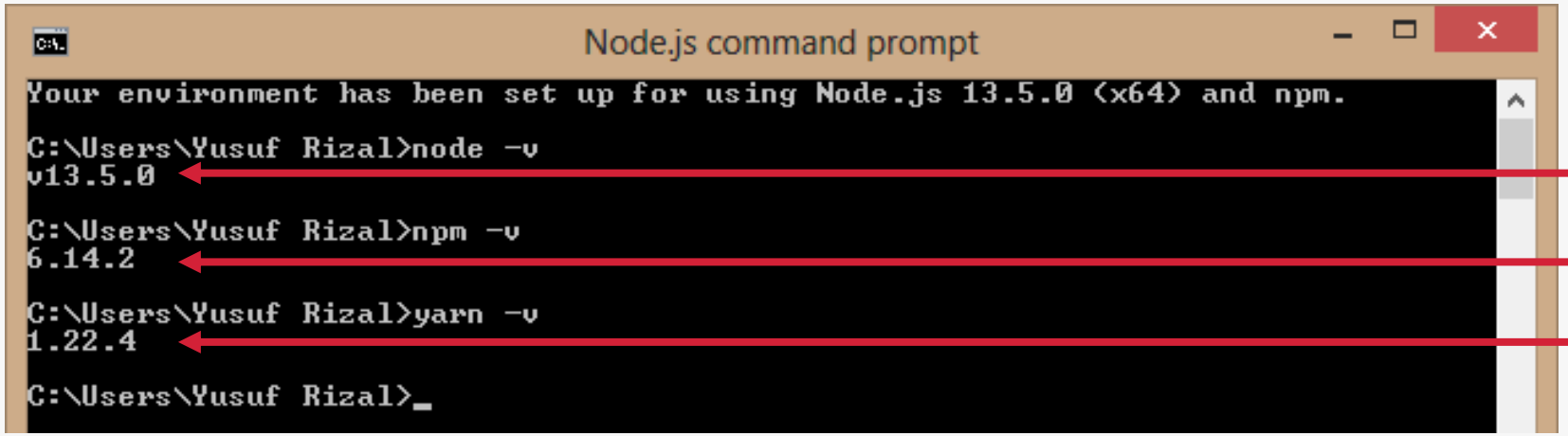
Membuat Aplikasi Pertama

Sebelum membuat aplikasi pertama, pastikan sudah mengunduh dan menginstal **Node.js** dan bisa menjalankan **npm / yarn** melalui 'command line' atau 'terminal'

https://nodejs.org/en/download

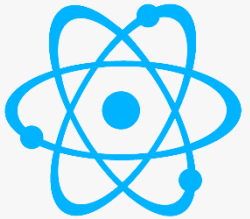


Cek versi Node, NPM, Yarn



Diatas merupakan versi NPM atau Yarn, pada workshop ini kita akan menggunakan Yarn secara umum.

Instalasi Kebutuhan Sistem

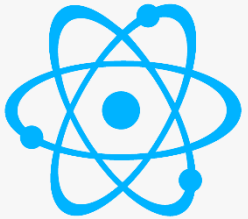


Kita memerlukan Node, React Native CLI, Python2, JDK dan Android Studio (SDK, AVD)

Instalasi Kebutuhan Sistem (2)

Kami merekomendasikan untuk menginstal kebutuhan sistem melalui Chocolatey, sebuah package manager yang popular digunakan di Windows.

Instalasi Chocolatey

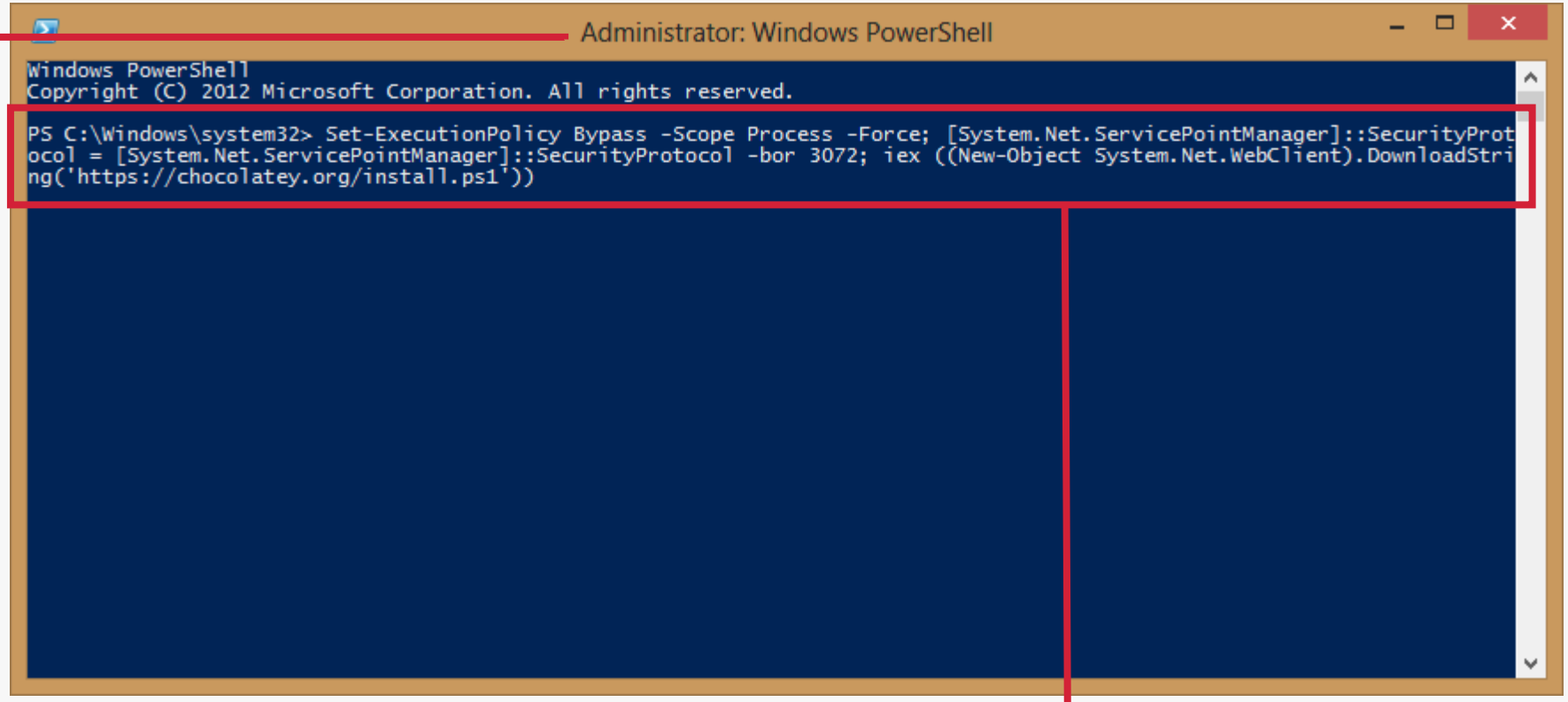


1) Pertama pastikan Windows Power Shell dalam posisi ‘Run as Administrator’
2) Copy dan Paste perintah dibawah kedalam Power Shell

Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))

Buka https://chocolatey.org/install

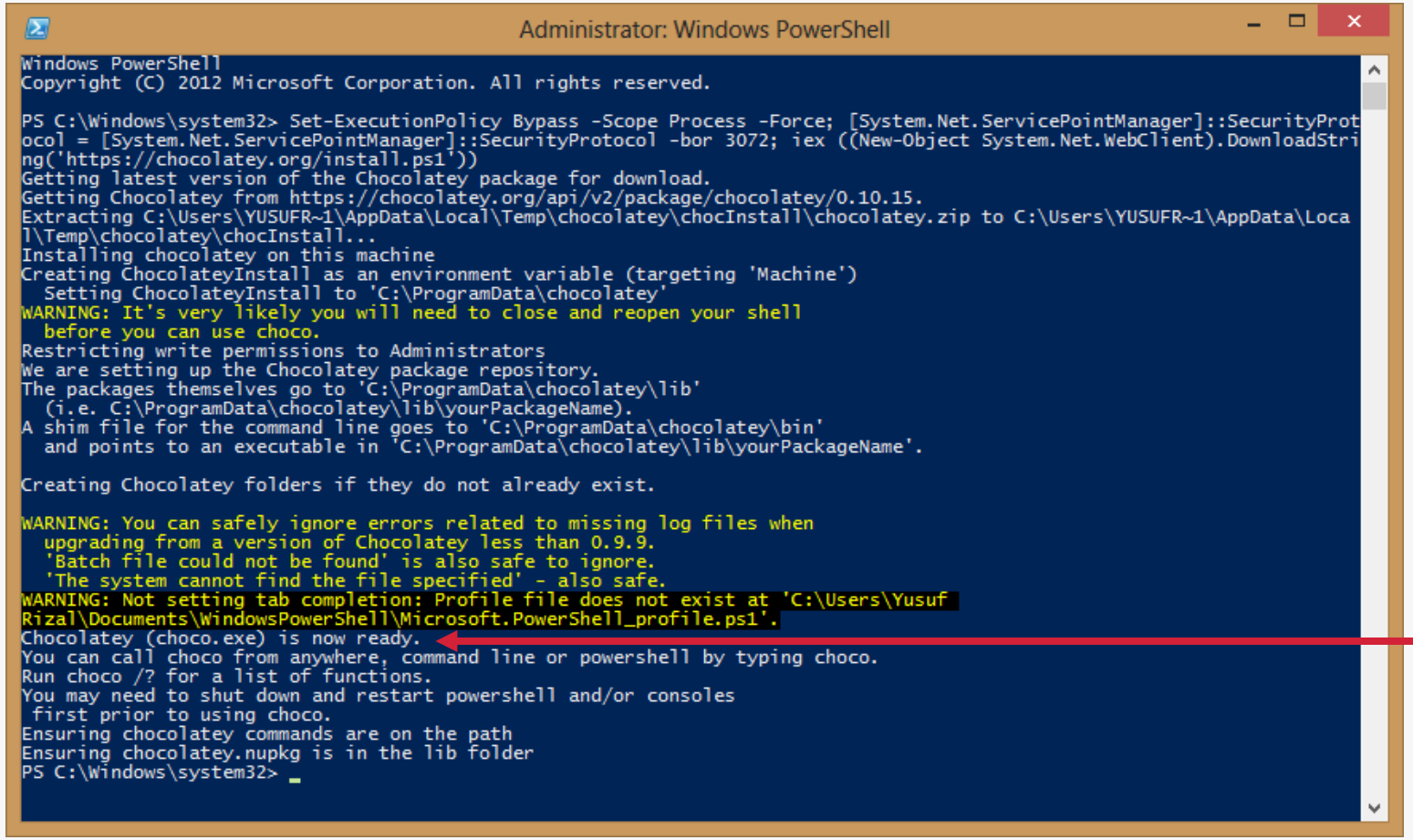
Instalasi Chocolatey (2)



Powershell posisi Administrator

Copy dan Paste perintah di Powershell

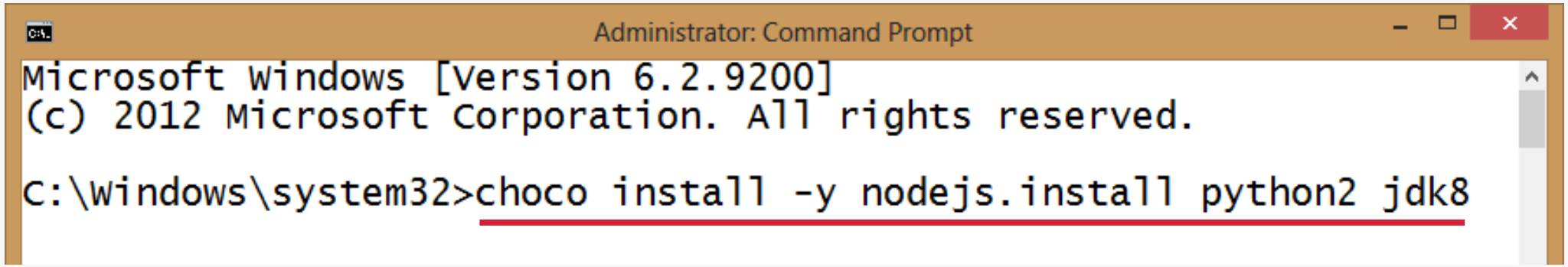
Instalasi Chocolatey (3)



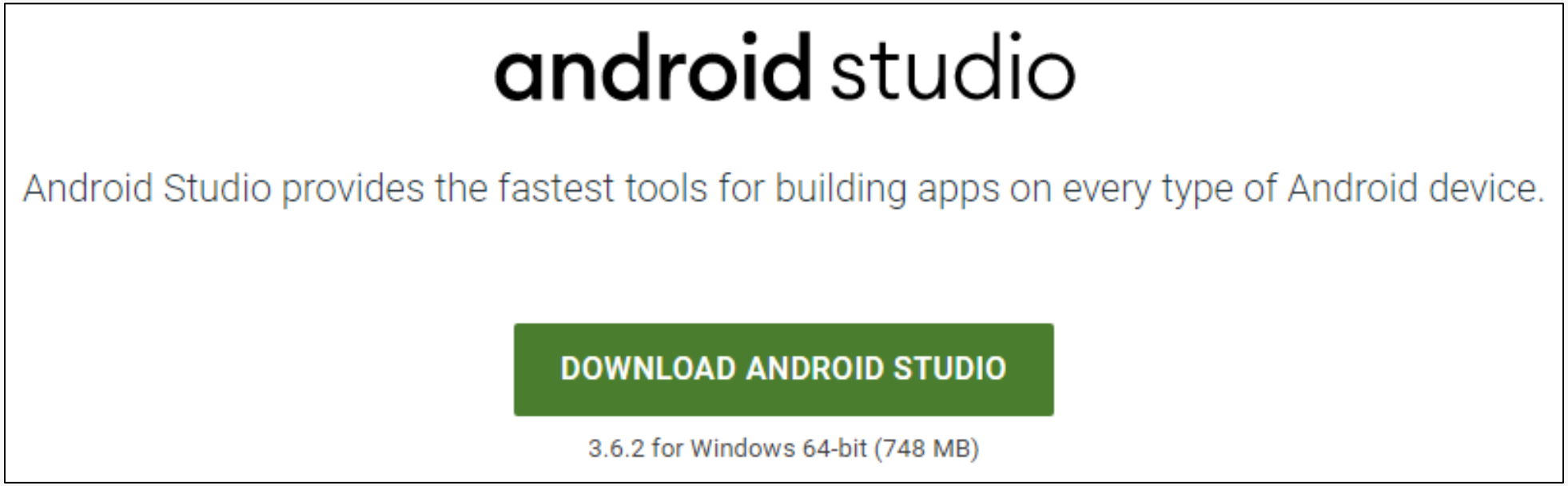
Proses instalasi hingga siap digunakan

Instalasi Kebutuhan Sistem (3)

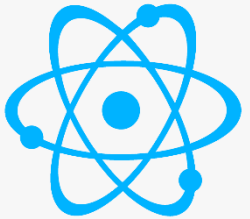
Buka CMD ‘Run as Administrator’ dan jalankan perintah dibawah:



Instalasi Android Studio, SDK, AVD (Emulator)



Instalasi React App Creator

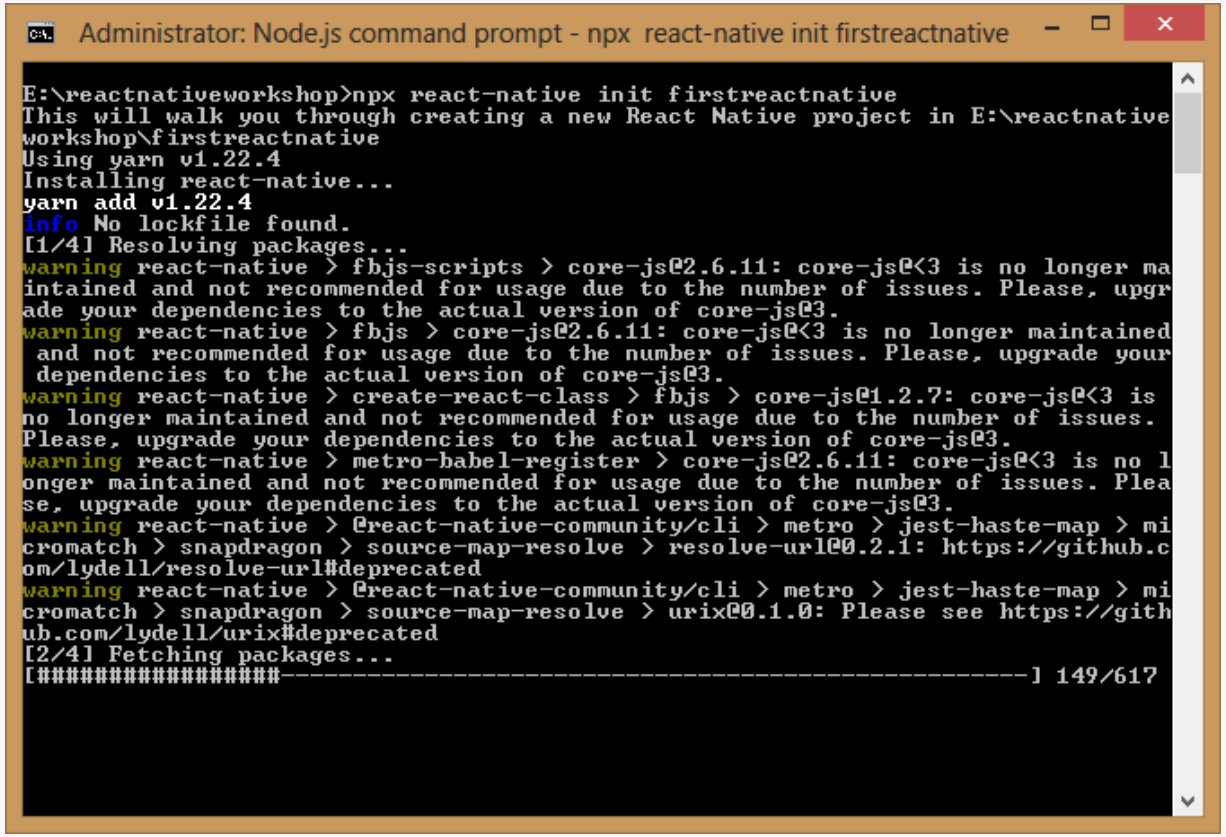


React Native memiliki antarmuka perintah tertulis yang built-in, dapat digunakan untuk menghasilkan proyek baru. Kita dapat mengaksesnya tanpa menginstal apapun menggunakan **npx**.

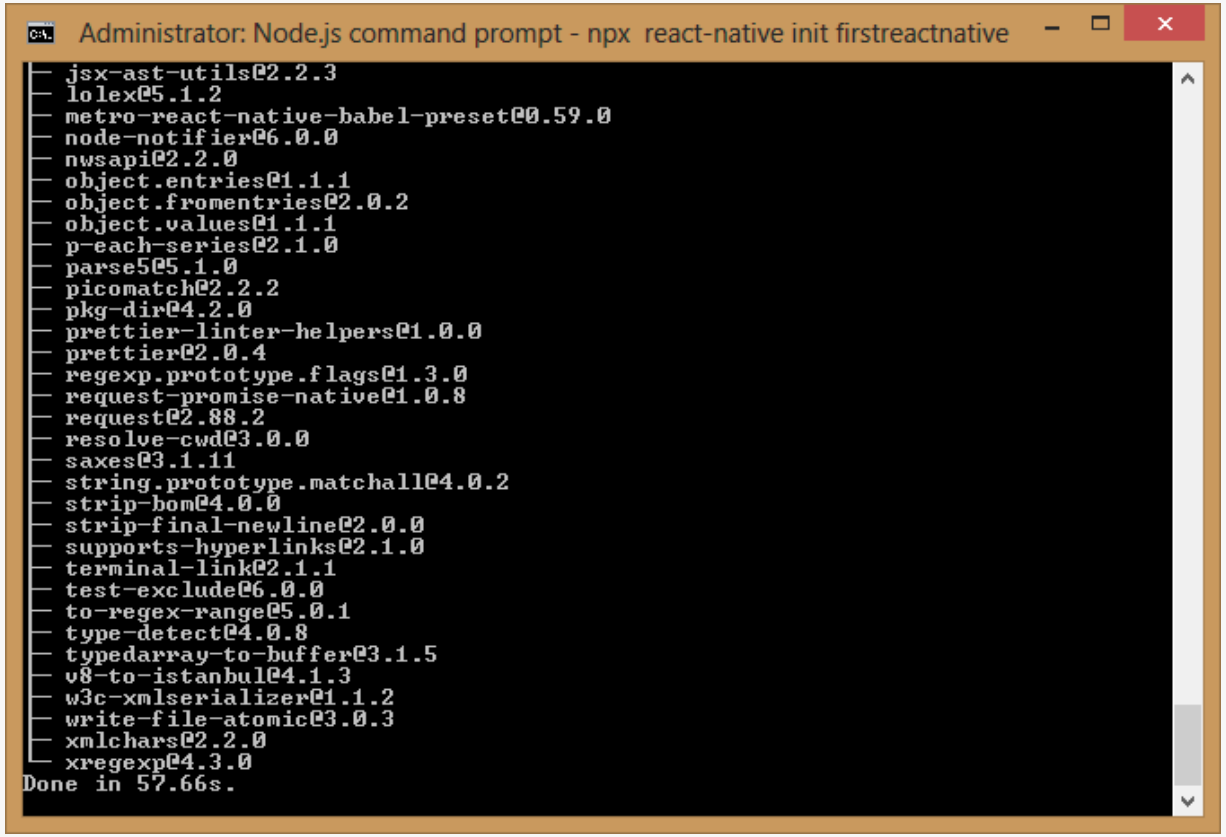
 # npx react-native init firstreactnative

Instalasi React App Creator (2)

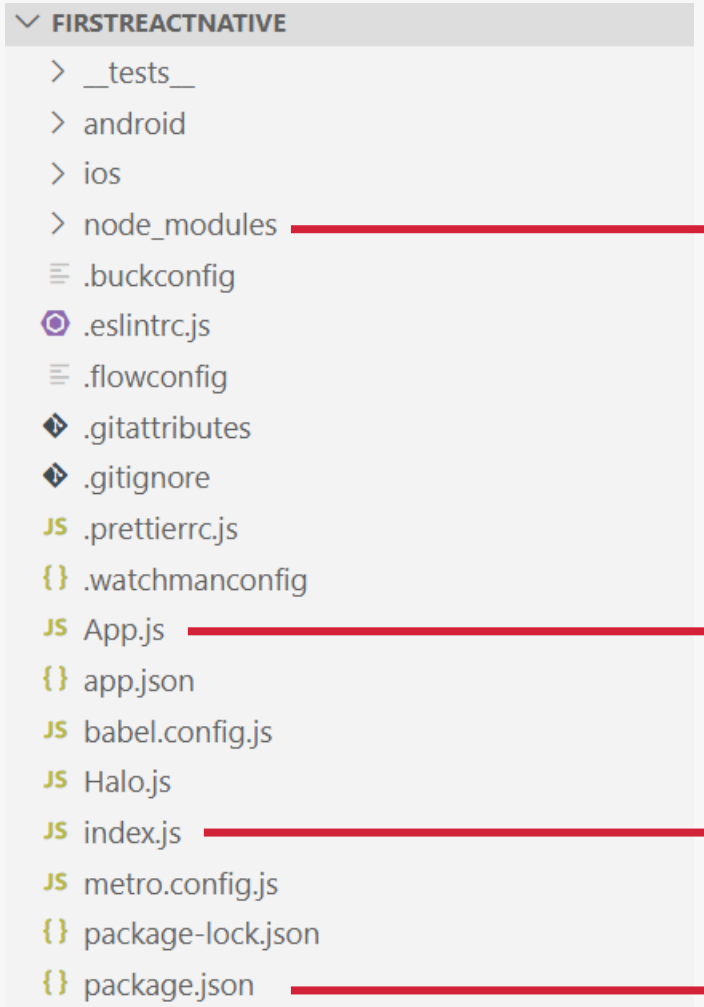
1



2



Struktur Folder Proyek



node_modules: Berisi seluruh modul yang dibutuhkan

App.js: berisi pemrograman javascript yang mengatur React native

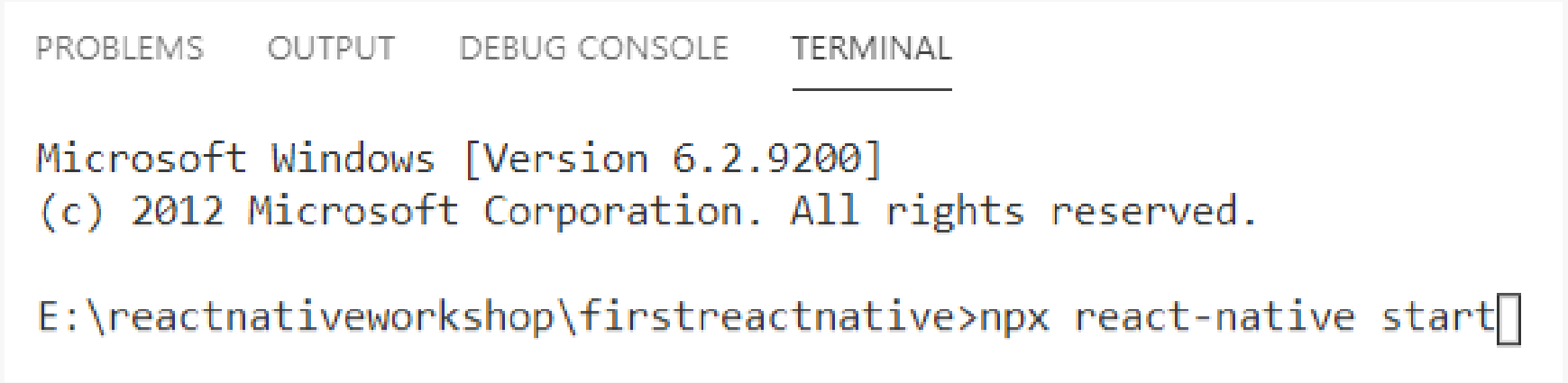
index.js: sebagai entry point aplikasi, class mana yang dibuka pertama kali

package.json: berisi informasi dependensi apa saja yang ada dalam aplikasi

Menjalankan aplikasi di Emulator Android

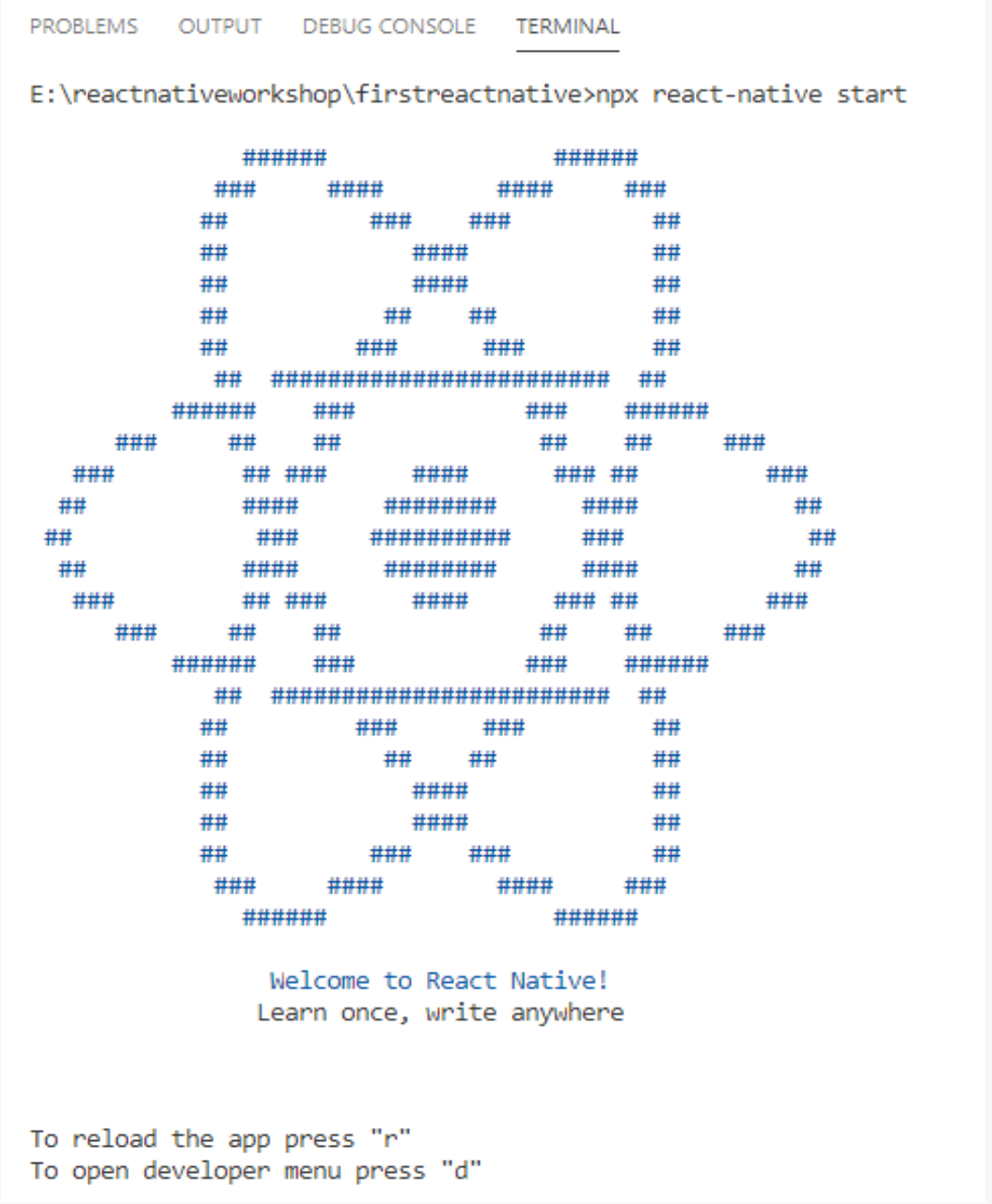
Pada Terminal, tuliskan perintah dibawah untuk memulai server:

 # npx react-native start

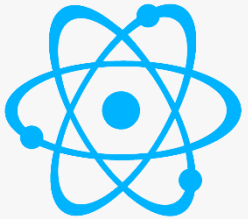


Menjalankan aplikasi di Emulator Android (2)

Tunggu hingga terminal memberikan keluaran seperti pada gambar disamping



Menjalankan aplikasi di Emulator Android (3)



Buka Terminal baru dan tuliskan perintah dibawah untuk mulai menjalankan aplikasi di Emulator:

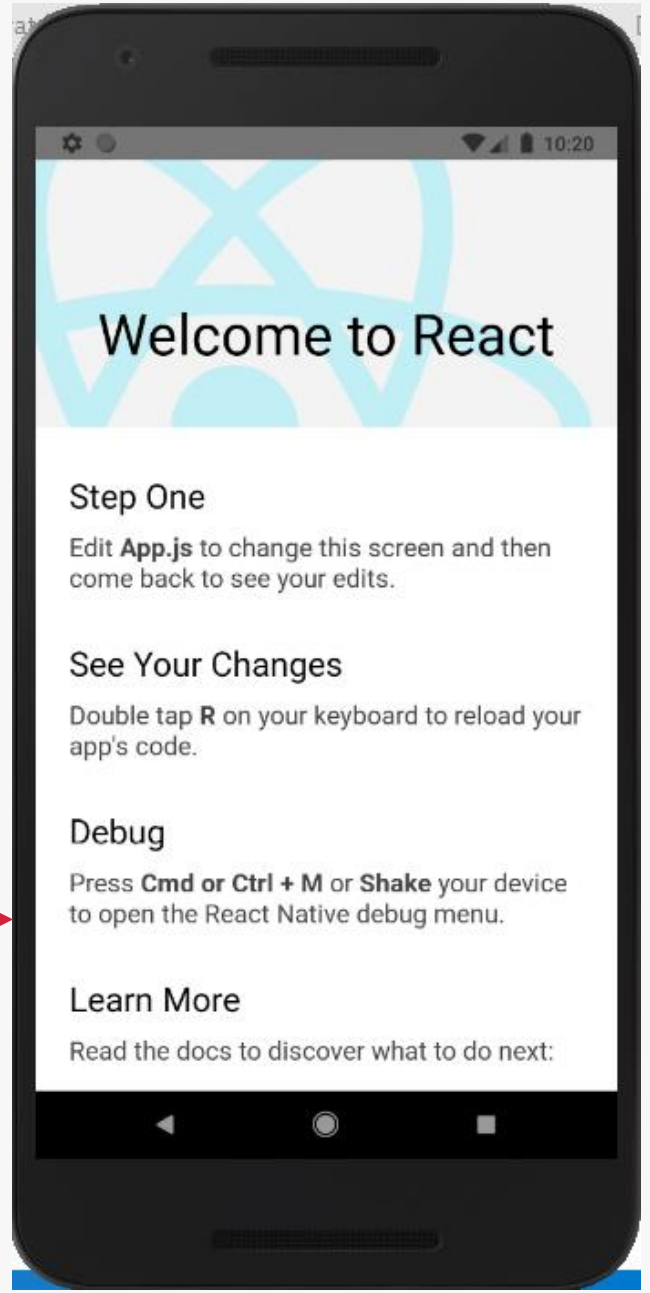
 # npx react-native run-android

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

Microsoft Windows [Version 6.2.9200]
(c) 2012 Microsoft Corporation. All rights reserved.

E:\reactnativeworkshop\firstreactnative>npx react-native run-android
```

Menjalankan aplikasi di Emulator Android (4)

```
BUILD SUCCESSFUL in 2m 14s
27 actionable tasks: 27 executed
info Connecting to the development server...
info Starting the app...
Starting: Intent { cmp=com.firstreactnative/.MainActivity }
```

Proses berhasil maka aplikasi dijalankan di emulator



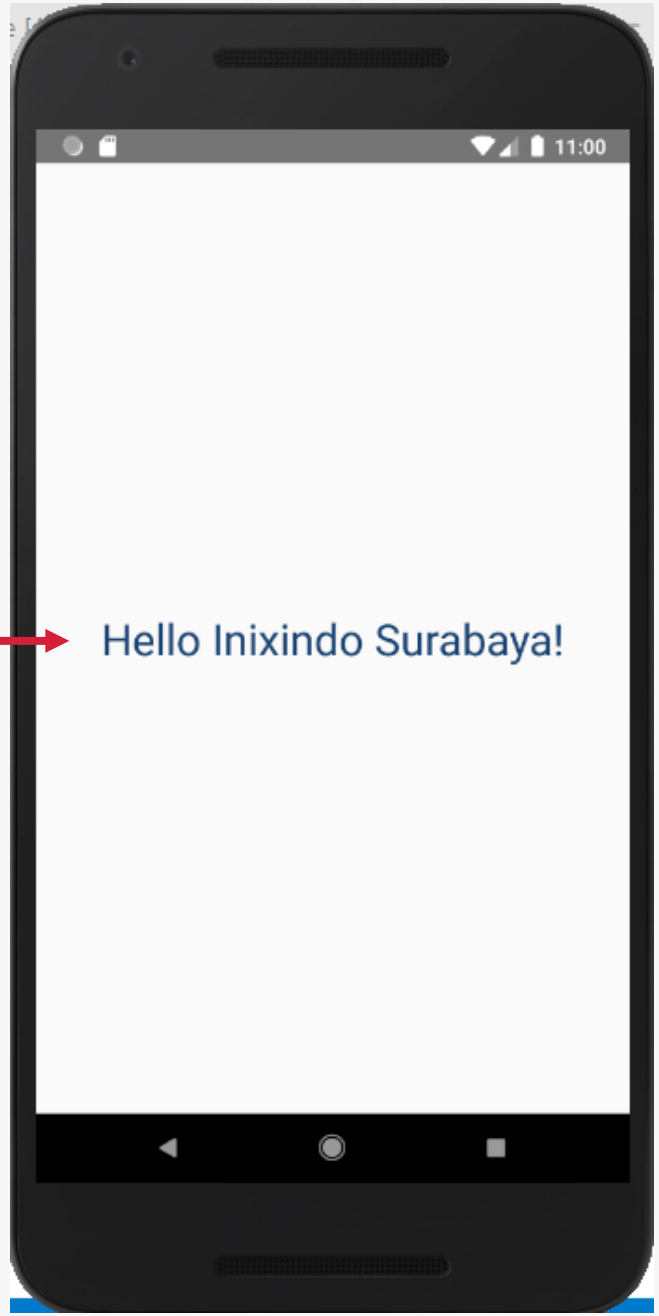
Mengubah tampilan pertama

Buka file App.js dan ubah sesuai kode disamping

```
const App: () => React$Node = () => {
  return (
    <>
      <View style={styles.container}>
        <Text style={styles.simpletext}>Hello Inixindo Surabaya!</Text>
      </View>
    </>
  );
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
    textAlign: 'center',
    alignItems: 'center',
    justifyContent: 'center',
  },
  simpletext: {
    textAlign: 'center',
    alignItems: 'center',
    justifyContent: 'center',
    fontSize: 30,
  },
});
```

Jalankan aplikasi

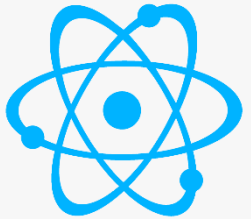
Muncul tulisan 'Hello Inixindo Surabaya!'
Dengan posisi teks berada ditengah layar



Mengidentifikasi dan mendeteksi kesalahan

Tekan tombol Ctrl + M untuk membuka menu **Debug**

Membedakan State dan Props



State

State merupakan data yang tersimpan dalam sebuah component. State bersifat private dan hanya relevan terhadap component itu sendiri, tidak bisa di akses dari component lain. Berbeda dengan props yang valuenya dilempar dari component lain, state justru dapat menyimpan dan mengubah datanya sendiri dari dalam.

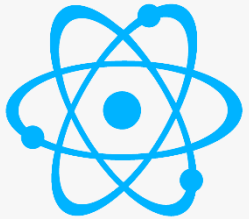
- Dimiliki oleh komponennya sendiri
- Bisa diubah dari dalam classnya sendiri (mutable)
- Memiliki setState() untuk mengubah nilainya

Props

Prop singkatan dari Property. Ini mirip dengan atribut pada tag HTML, dalam pembuatannya value dari props bisa dilempar dan diakses dari atau ke component lain.

- Bersifat immutable (tidak bisa diubah)
- Bisa diakses dan dilempar dari dan ke component lain

Membedakan State dan Props



Buka lagi file App.js untuk memahami **State**

Ubah sesuai petunjuk kode berikut ini:

```
export default class App extends Component {
  constructor(props) {
    super(props);
    // buat state nilai awal disini
    this.state = {
      value: 'Hello Inixindo Surabaya',
    };
  }

  // fungsi untuk mengubah isi text input
  onChangeTextInput = (text) => {
    // buat setState untuk mengubah state nilai awal sebelumnya
    this.setState({value: text});
  };
```

```
render() {
  let {container, textStyle, textInput} = styles;
  let {value} = this.state;
  return (
    <View style={container}>
      {/* <Text style={textStyle}>Hello Inixindo Surabaya</Text> */}
      <Text style={textStyle}>{this.state.value}</Text>
      <TextInput
        style={textInput}
        onChangeText={this.onChangeTextInput}
        // value={'value text input'}
        // ubah value ditas sesuai dengan state saat ini
        value={value}
      />
    </View>
  );
}
```

Membedakan State dan Props

Buka lagi file App.js untuk memahami **Props**

Buat sebuah file baru **ChildComponent.js**

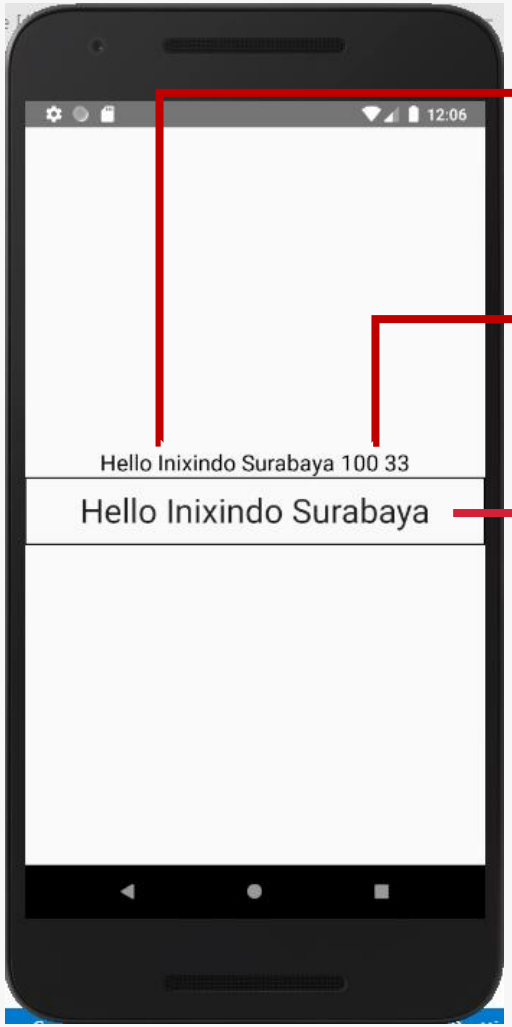
```
import React, {Component} from 'react';
import {View, Text} from 'react-native';

export default class ChildComponent extends Component {
  constructor(props) {
    super(props);
  }

  render() {
    let {data1} = this.props;
    return (
      <View>
        <Text style={{fontSize: 20}}>{data1}</Text>
      </View>
    );
  }
}
```

Membedakan State dan Props

Buka kembali file App.js dan tambahkan baris kode seperti pada gambar berikut:

```
import {ChildComponent} from './ChildComponent';

render() {
  let {container, textStyle, textInput} = styles;
  let {value} = this.state;
  return (
    <View style={container}>
      <ChildComponent data1={value} data2={'100'} data3={'33'} />

      {/* <Text style={textStyle}>Hello Inixindo Surabaya</Text> */}
      {/* <Text style={textStyle}>{this.state.value}</Text> */}

      <TextInput
        style={textInput}
        onChangeText={this.onChangeTextInput}
        // value={'value text input'}
        // ubah value ditas sesuai dengan state saat ini
        value={value}
      />
    </View>
  );
}
```

Hasil State dan Props

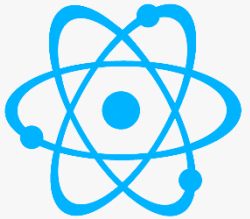


State => nilai awal yang ditentukan dan bisa diubah menggunakan setState() dari dalam component itu sendiri (mutable)

Props => nilai asli dari component dan tidak bisa diubah (immutable) tapi dapat dilempar ke component yang lain

Text Input untuk menuliskan teks

Menggunakan React Navigation

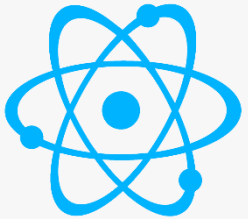


Untuk menggunakan React Navigation, harus mengunduhnya terlebih dahulu, tuliskan perintah dibawah ini:

 # yarn add react-navigation

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

Microsoft Windows [Version 6.2.9200]
(c) 2012 Microsoft Corporation. All rights reserved.

E:\reactnativeworkshop\firstreactnative>yarn add react-navigation
```

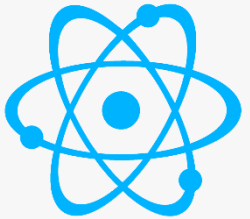
Lihat Package.json



```
{
  "name": "firstreactnative",
  "version": "0.0.1",
  "private": true,
  "scripts": {
    "android": "react-native run-android",
    "ios": "react-native run-ios",
    "start": "react-native start",
    "test": "jest",
    "lint": "eslint ."
  },
  "dependencies": {
    "react": "16.11.0",
    "react-native": "0.62.1",
    "react-navigation": "^4.3.7"
  },
```

React Navigation berhasil ditambahkan

Membaca data dari Web API

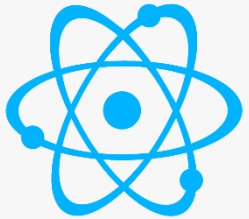


Pada bagian ini kita akan membuat sebuah aplikasi React Native yang digunakan untuk membaca data dari Web API berbentuk json.

Buat sebuah folder baru dengan nama **components**

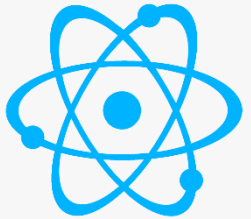
Didalam folder tersebut buat 2 file baru dengan nama masing-masing **HomeScreen.js** dan **DetailScreen.js**

Buka file HomeScreen.js



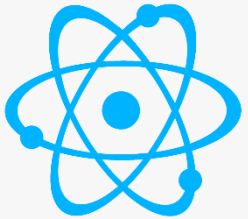
```
import React, { Component } from 'react';
import { Button, View, Text } from 'react-native';

class HomeScreen extends Component {
  static navigationOptions = {
    title: 'Home',
  };
  render() {
    return (
      <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
        <Text>Home Screen</Text>
        <Button
          title="Go to Details"
          onPress={() => this.props.navigation.navigate('Details')}
        />
      </View>
    );
  }
}

export default HomeScreen;
```

Buka file DetailScreen.js



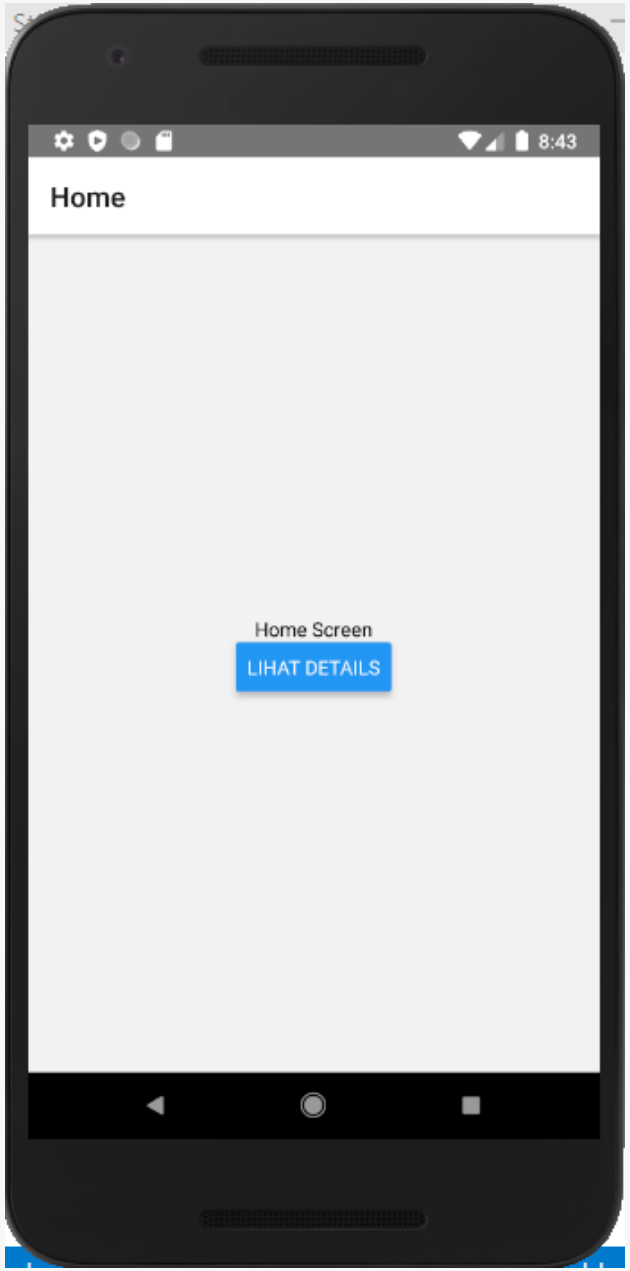
```
import React, { Component } from "react";
import { Button, View, Text } from "react-native";

class DetailsScreen extends Component {
  static navigationOptions = {
    title: "Details"
  };
  render() {
    return (
      <View style={{ flex: 1, alignItems: "center", justifyContent: "center" }}>
        <Text>Details Screen</Text>
        <Button
          title="Buka Details... lagi"
          onPress={() => this.props.navigation.push("Details")}
        />
        <Button
          title="Kembali ke Home"
          onPress={() => this.props.navigation.navigate("Home")}
        />
        <Button
          title="Kembali"
          onPress={() => this.props.navigation.goBack()}
        />
      </View>
    );
  }
}

export default DetailsScreen;
```

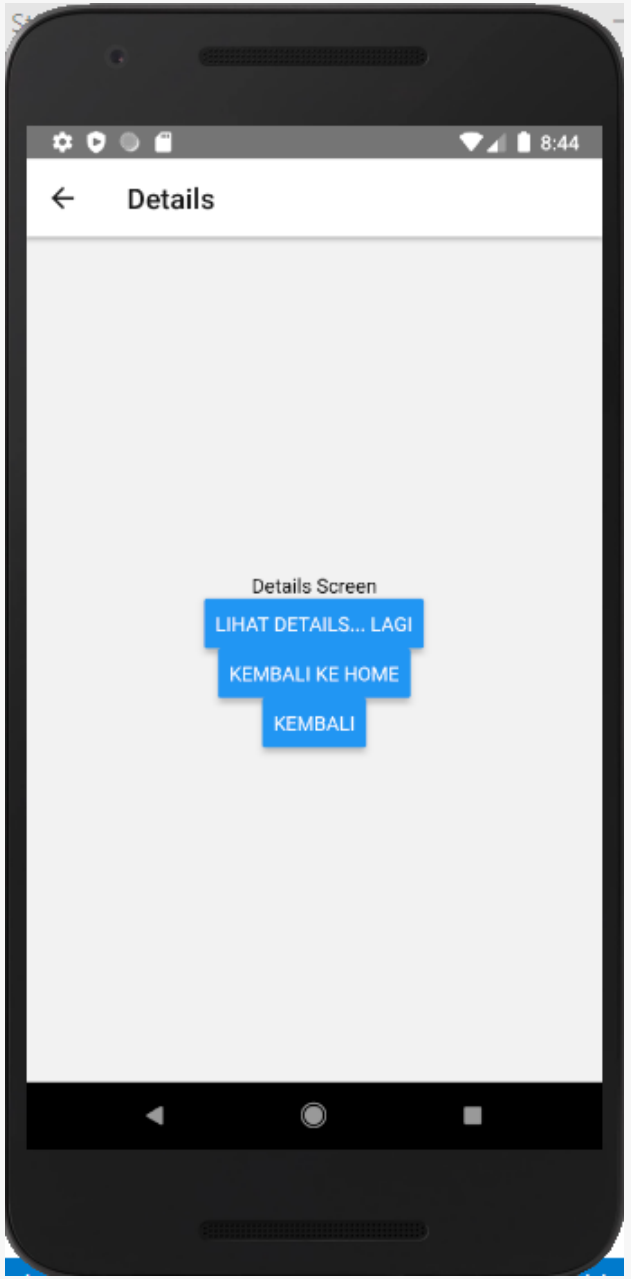
Buka file App.js



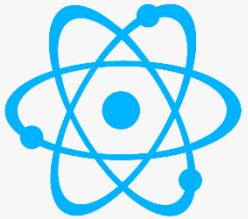
```
import React from "react";
import { StyleSheet, Text, View } from "react-native";

import { createAppContainer, createSwitchNavigator } from "react-navigation";
import { createStackNavigator } from "react-navigation-stack";

import HomeScreen from "./components/HomeScreen";
import DetailsScreen from "./components/DetailsScreen";

const StackNavigator = createStackNavigator(
  {
    Home: {screen: HomeScreen},
    Details: {screen: DetailsScreen}
  },
  {
    initialRouteName: "Home",
    navigationOptions: {
      headerStyle: {
        backgroundColor: "#f4511e"
      },
      headerTintColor: "#fff",
      headerTitleStyle: {
        fontWeight: "bold"
      }
    }
  }
);

export default createAppContainer(StackNavigator);
```

Jalankan aplikasi di Emulator





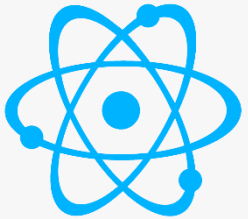
Menampilkan data Web API World Cup 2018



Pada components/HomeScreen.js tambahkan constructor sebelum fungsi rendering data

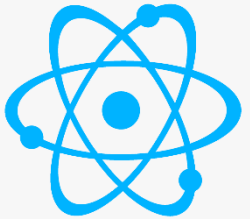
```
constructor(props){
  super(props);
  this.state ={ isLoading: true}
}
```

Fungsi menampilkan data API json



```
componentDidMount(){
  return fetch('https://raw.githubusercontent.com/openfootball/world-cup.json/master/2018/worldcup.json')
    .then((response) => response.json())
    .then((responseJson) => {
      // console.log(responseJson);
      this.setState({
        isLoading: false,
        dataSource: responseJson.rounds,
      }, function(){

      });

    })
    .catch((error) =>{
      console.error(error);
    });
}
```

Instalasi React Native Elements



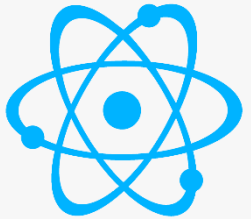
Kita bisa menggunakan komponen List dan ListItem yang ada pada react-native-elements, cara instalasinya sebagai berikut:

```
# yarn add react-native-elements
```

Buka kembali HomeScreen.js dan tambahkan importnya

```
import { List, ListItem } from 'react-native-elements';
```

Render HomeScreen.js

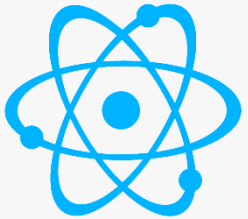


```
render() {
  if(this.state.isLoading){
    return(
      <View style={styles.activity}
        <ActivityIndicator/>
      </View>
    )
  }
```

```
return(
  <ScrollView style={styles.container}>
    <List>
      {
        this.state.dataSource.map((item, i) => (
          <ListItem
            key={i}
            title={item.name}
            leftIcon={{name: 'soccer-ball-o', type: 'font-awesome'}}
            onPress={() => {
              this.props.navigation.navigate('Details', {
                matches: `${JSON.stringify(item.matches)}`,
              });
            }}
          />
        ))
      }
    </List>
  </ScrollView>
);
}
```

```
const styles = StyleSheet.create({
  container: {
   flex: 1,
   paddingBottom: 22
  },
  item: {
    padding: 10,
    fontSize: 18,
    height: 44,
  },
  activity: {
    flex: 1,
    padding: 20,
  }
})
```

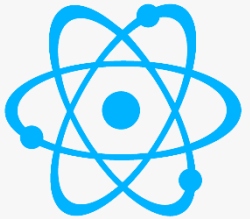
Menampilkan Details Pertandingan



Buka kembali file DetailsScreen.js dan tambahkan importnya:

```
import React, { Component } from 'react';
import { StyleSheet, ScrollView, View } from 'react-native';
import { List, ListItem, Text, Card } from 'react-native-elements';
```

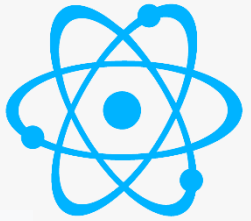
Render DetailScreen.js



```
render() {
  const { navigation } = this.props;
  const matches = JSON.parse(navigation.getParam('matches', 'No matches found'));
  console.log(matches);
```

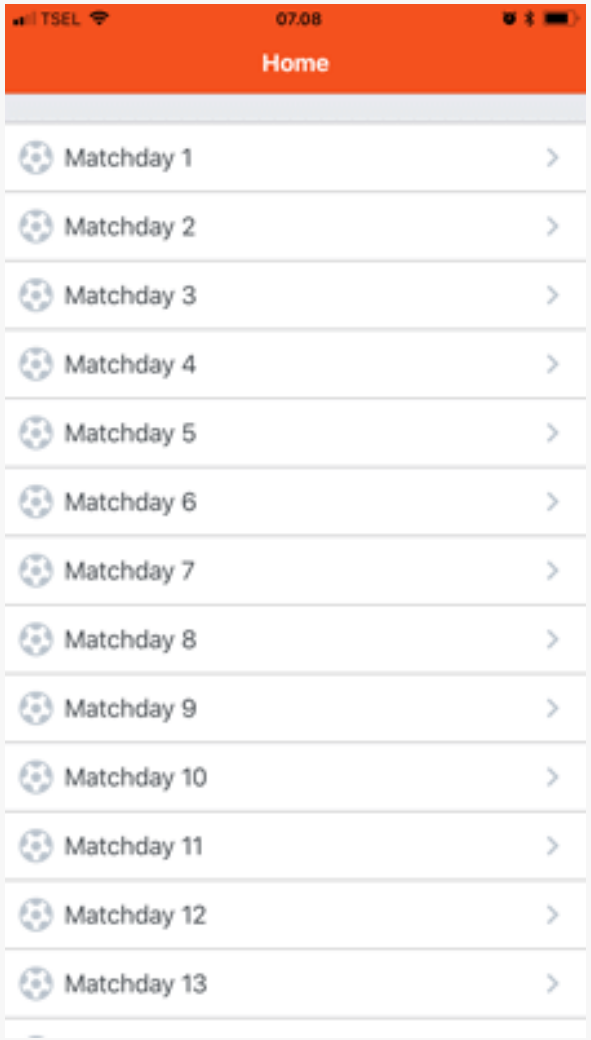
```
const styles = StyleSheet.create({
  container: {
    flex: 1,
    padding: 20
  },
  subContainer: {
    flex: 1,
    paddingBottom: 20,
    borderBottomWidth: 2,
    borderBottomColor: '#CCCCCC',
  }
})
```

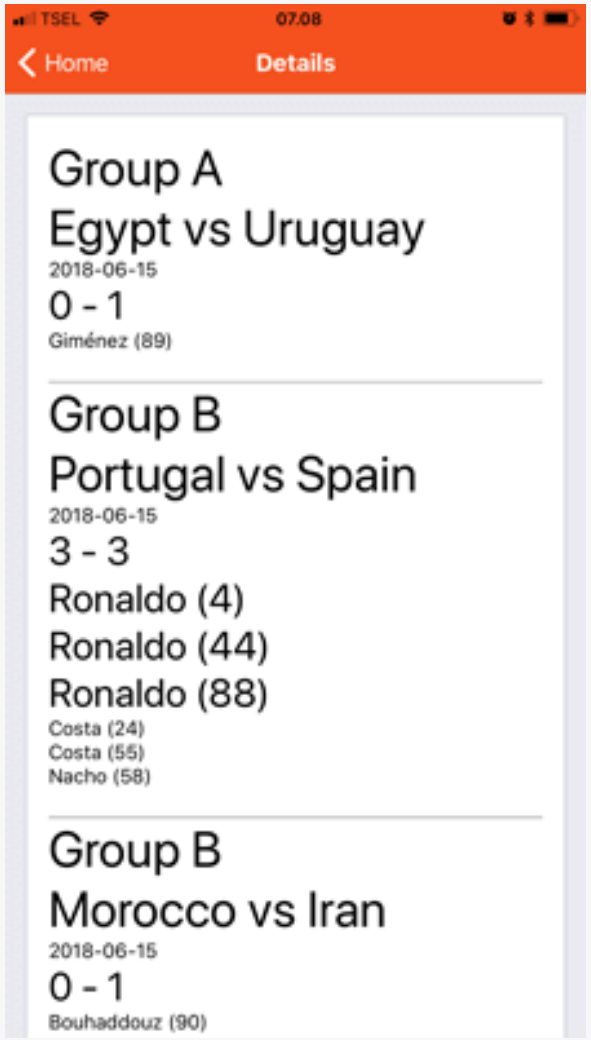
Render DetailScreen.js (2)



```
return (
  <ScrollView>
    <Card style={styles.container}>
      {
        matches.map((item, key) => (
          <View key={key} style={styles.subContainer}>
            if(item.group) {
              <View>
                <Text h3>{item.group}</Text>
              </View>
            }
            <View>
              <Text h3>{item.team1.name} vs {item.team2.name}</Text>
            </View>
            <View>
              <Text h5>{item.date}</Text>
            </View>
            <View>
              <Text h4>{item.score1} - {item.score2}</Text>
            </View>
            if(item.goals1.length > 0) {
              item.goals1.map((item2, key2) => (
                <View key={key2}>
                  <Text h4>{item2.name} ({item2.minute})</Text>
                </View>
              ))
            }
            if(item.goals2.length > 0) {
              item.goals2.map((item3, key3) => (
                <View key={key3}>
                  <Text h5>{item3.name} ({item3.minute})</Text>
                </View>
              ))
            }
          </View>
        ))
      }
    </Card>
  </ScrollView>
);
}
```

Hasil


TSEL
07.08
Home
Matchday 1
Matchday 2
Matchday 3
Matchday 4
Matchday 5
Matchday 6
Matchday 7
Matchday 8
Matchday 9
Matchday 10
Matchday 11
Matchday 12
Matchday 13


TSEL
07.08
Home
Details
Group A
Egypt vs Uruguay
2018-06-15
0 - 1
Giménez (89)
Group B
Portugal vs Spain
2018-06-15
3 - 3
Ronaldo (4)
Ronaldo (44)
Ronaldo (88)
Costa (24)
Costa (55)
Nacho (58)
Group B
Morocco vs Iran
2018-06-15
0 - 1
Bouhaddouz (90)


inixindo
Surabaya