



# React Native HTTP Request

**Silver**- Chapter 3 - Topic 5

---

**Selamat datang di Chapter 3 Topic 5 pada course  
React Native dari Binar Academy!**





### Haalloo teman-teman! 🙋

Pada topik sebelumnya kamu telah mempelajari **HTTP Request** dan cara menggunakan **RESTful API** untuk membaca, membuat, mengupdate, dan menghapus data dari server menggunakan **REST Client (POSTMAN)**.

Nah di topik kelima ini, kamu akan belajar melakukan **HTTP Request** didalam React Native dengan menggunakan **Axios**.



**Detailnya, kita bakal bahas hal-hal berikut ini:**

- Axios
- CRUD





Untuk melakukan operasi yang berkaitan dengan RESTful API di aplikasi React Native, kamu akan membutuhkan bantuan sebuah library yang bernama AXIOS.

**Yuk mari, kita cari tahu segala hal tentang AXIOS!**



## Ternyata POSTMAN nggak bisa digunakan di React Native 😱

Di topik sebelumnya, kamu sudah belajar melakukan operasi-operasi yang berhubungan dengan RESTful API menggunakan POSTMAN.

Tapi, sayangnya POSTMAN nggak bisa digunakan di dalam React Native untuk melakukan operasi-operasi tersebut, guys!



# POSTMAN



## Jangan takut! Ada solusinya~

Terus kalau kita mau melakukan operasi-operasi yang berhubungan dengan RESTful API **langsung di dalam React Native** pakai apa dong?

Kalem aja! Untuk melakukan itu, kita bisa **menggunakan sebuah library bernama Axios.**

Apa sih yang dimaksud dengan Axios? Bagaimana cara kerjanya? Yuk langsung kita kepoin aja, guys!

# A X I O S



## Please welcome... Axios!

Axios adalah **sebuah library open source** yang kini lagi booming banget dipakai **untuk melakukan request HTTP** pada bahasa pemrograman JavaScript.

Kok bisa?

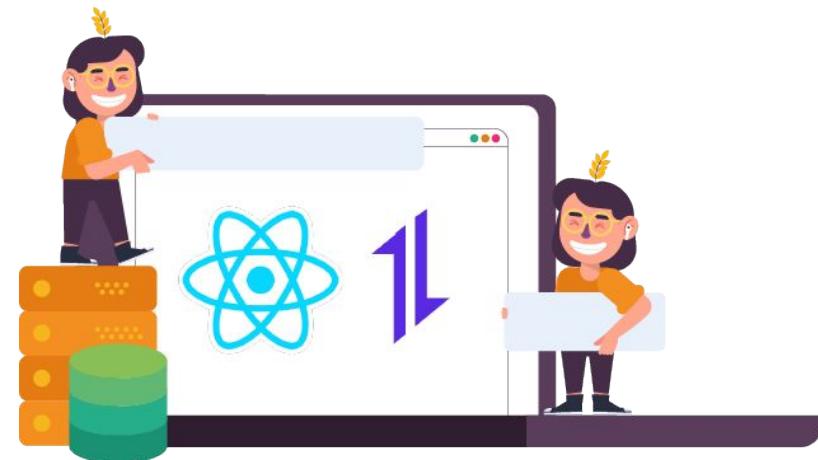
Ya, ini karena Axios dianggap punya banyak kelebihan dan lebih mudah digunakan.



## Penasaran apa bedanya POSTMAN dan Axios? Ini dia jawabannya!

Simpelnya, Axios mirip banget kayak aplikasi POSTMAN, yang tujuannya untuk memproses Request data ke RESTful API pada Server.

**Bedanya adalah letak Axios berada di dalam React Native.**





## Implementasi HTTP Method pakai Axios, hmm bagaimana caranya nih?

Sebelumnya kita udah bahas, ada beberapa HTTP Method yang umum dipakai yaitu GET, POST, PUT/PATCH, DELETE.

Nah kalau di Axios, untuk melakukan request dengan HTTP method yang berbeda, maka berbeda juga cara pemanggilan/penulisan pada method yang akan dipanggil.

HTTP METHOD	PENULISAN PADA AXIOS
GET	axios.get(URL)
POST	axios.post(URL, OBJECTDATA)
PUT	axios.put(URL, OBJECTDATA)
PATCH	axios.patch(URL, OBJECTDATA)
DELETE	axios.delete(URL)



**Biar kamu lebih paham, kita lihat contoh  
syntax tiap methodnya yuk!**





## Contoh penulisan HTTP Method GET pada Axios

Misalnya **kamu mau ngambil data** dari BMKG pakai Axios, maka kamu akan membuat HTTP dengan method GET ke URL

<https://data.bmkg.go.id/DataMKG/TEWS/autogempa.json>

Karena Axios akan menghasilkan data yang berbentuk Promise, maka kamu bisa **menggunakan method then()** sebagai handler, selain itu **kamu juga bisa pakai async/await.**

### HTTP METHOD

GET

### PENULISAN PADA AXIOS

```
import axios from 'axios';

// Membuat GET Request untuk membaca data gempa
// Pada server BMKG
axios.get('https://data.bmkg.go.id/DataMKG/TEWS/autogempa.json').then(function (response) {
    // handle ketika HTTP Request berhasil
    console.log(response);
})
```

Handle **axios** promise dengan **.then()**

```
import axios from 'axios';

async function dataGempa(){
    // Membuat GET Request untuk membaca data gempa
    // Pada server BMKG
    const response = await axios.get('https://data.bmkg.go.id/DataMKG/TEWS/autogempa.json');
    console.log(response)
}
dataGempa()
```

Handle **axios** promise dengan **async/await**



## Contoh penulisan HTTP Method POST pada Axios

Misalnya **kamu mau membuat data** todo list baru pada server 'code.aldipee.com', maka kamu akan membuat HTTP Request ke <http://code.aldipee.com/api/v1/todos> dengan menggunakan method POST

Karena Axios akan menghasilkan data yang berbentuk Promise, maka kamu bisa **menggunakan method then()** sebagai handler, selain itu **kamu juga bisa pakai async/await.**

### HTTP METHOD

POST

### PENULISAN PADA AXIOS

axios.post(URL, OBJECTDATA)

```
import axios from 'axios';

// Membuat POST request untuk membuat data baru
const dataItem = {
  title: 'Hello world',
  status: 'PENDING'
};

axios.post('http://code.aldipee.com/api/v1/todos', dataItem).then(function(response) {
  // handle ketika HTTP Request berhasil
  console.log(response)
})
```

Handle **axios** promise dengan **.then()**

```
import axios from 'axios';

async function createNewData(){
  const dataItem = {
    title: 'Hello world',
    status: 'PENDING'
  };
  // Membuat POST request untuk menambahkan data baru di server
  const response = await axios.post('http://code.aldipee.com/api/v1/todos/622a006c2a0ad93e1e94630a', dataItem);
  console.log(response)
}

createNewData();
```

Handle **axios** promise dengan **async/await**



## Contoh penulisan HTTP Method PUT pada Axios

Misalnya **kamu mau melakukan update** pada sebuah item pada server [code.aldipee.com](http://code.aldipee.com), maka kamu akan membuat **HTTP Request** ke <http://code.aldipee.com/api/v1/todos/622a006c2a0ad93e1e94630a> dengan menggunakan method **PUT**

Karena Axios akan me-return data yang berbentuk Promise, maka kamu bisa **menggunakan method then()** sebagai handler, selain itu **kamu juga bisa pakai async/await**.

### HTTP METHOD

PUT

### PENULISAN PADA AXIOS

axios.put(URL, OBJECTDATA)

```
import axios from 'axios';

// Membuat PUT request untuk mengupdate sebuah data
const dataItem = {
  title: 'Hello world',
  status: 'PENDING'
};

axios.put('http://code.aldipee.com/api/v1/todos/622a006c2a0ad93e1e94630a', dataItem).then(function(response) {
  // handle ketika HTTP Request berhasil
  console.log(response)
})
```

Handle **axios** promise dengan **.then()**

```
import axios from 'axios';

async function updateData(){
  const dataItem = {
    title: 'Hello world',
    status: 'PENDING'
  };
  // Membuat PUT request untuk mengupdate sebuah data
  const response = await axios.put('http://code.aldipee.com/api/v1/todos/622a006c2a0ad93e1e94630a', dataItem);
  console.log(response)
}

updateData();
```

Handle **axios** promise dengan **async/await**



## Contoh penulisan HTTP Method DELETE pada Axios

Misalnya **kamu mau menghapus data** sebuah todo list pada server [code.aldipee.com](http://code.aldipee.com), maka kamu akan membuat HTTP Request ke <http://code.aldipee.com/api/v1/todos/622a006c2a0ad93e1e94630a> dengan menggunakan method **DELETE**.

Karena Axios akan menghasilkan data yang berbentuk Promise, maka kamu bisa **menggunakan method then()** sebagai handler selain itu **kamu juga bisa pakai async/await**.

### HTTP METHOD

#### DELETE

### PENULISAN PADA AXIOS

```
axios.delete(URL)
```

```
import axios from 'axios';

axios.delete('http://code.aldipee.com/api/v1/todos/622a006c2a0ad93e1e94630a').then(function(response) {
  // handle ketika HTTP Request berhasil
  console.log(response)
})
```

Handle **axios** promise dengan **.then()**

```
import axios from 'axios';

async function deleteData(){
  // Menghapus data pada server
  const response = await axios.delete('http://code.aldipee.com/api/v1/todos/622a006c2a0ad93e1e94630a');
  console.log(response)
}

deleteData();
```

Handle **axios** promise dengan **async/await**



## Jangan cuma rajin baca chat gebetan. Baca dokumentasi Axios juga penting!

Ho oh.. sebagai seorang programmer, kamu harus membiasakan diri untuk membaca dokumentasi dari library yang kamu pakai ya. Kenapa begitu?

Karena pemirsa, di sana ada **banyak banget method-method atau fitur-fitur yang tersedia untuk beberapa kasus spesifik tertentu.**

Kamu bisa baca dokumentasi Axios [di sini](#) ya!

The screenshot shows the Axios documentation website. On the left, there's a sidebar with navigation links: English, Getting Started (Introduction, Example, POST Requests), Axios API (Axios API, The Axios Instance, Request Config, Response Schema, Config Defaults, Interceptors, Handling Errors), and a toggle switch. The main content area has a dark header with the Axios logo. Below it, a large heading says "Minimal Example". A sub-section titled "A little example of using axios" contains code snippets for CommonJS usage:

```
const axios = require('axios').default;  
// axios.{method} will now provide autocomplete and parameter typings
```

Another section titled "Example" shows code for performing a GET request:

```
const axios = require('axios');  
  
// Make a request for a user with a given ID  
axios.get('/user?ID=12345')  
.then(function (response) {  
  // handle success  
})
```



Selanjutnya kita bakal belajar tentang **CRUD pada React Native**. Ada yang tahu nggak apa itu **CRUD**?

Oke, kalau begitu langsung aja kita pelajari bersama 😊



# Apa itu CRUD?

Jadi guys, CRUD adalah singkatan dari **create, read, update, and delete**.

Empat poin ini merupakan fungsi-fungsi yang biasanya terdapat dalam sebuah aplikasi.

Jadi kalau orang yang nyuruh kamu bikin aplikasi CRUD, artinya aplikasi kamu harus bisa support keempat fitur ini, yaitu Create (menambahkan), Read (menampilkan data), Update (mengubah data), dan Delete (menghapus data).

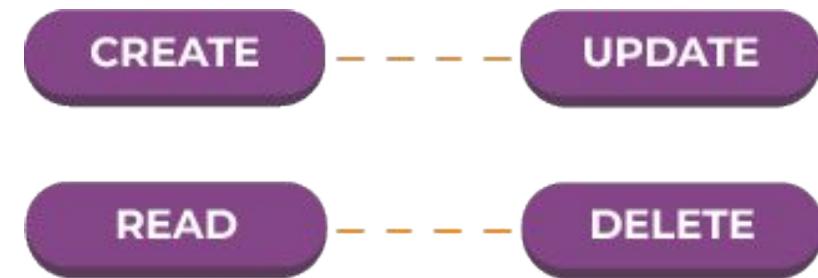




Pada dasarnya cara kerja antara **create dengan update itu sama aja**, guys! Begitu juga dengan cara kerja antara **read dengan delete juga sama aja ya**.

Jadi kalau kamu udah paham cara kerja salah satu di antaranya, maka kamu pasti paham cara kerja yang lainnya.

Makanya, di sesi kali ini kita cukup coba mengimplementasikan Create dan Read aja pada aplikasi React Native kamu.





Kita mulai dengan cara  
mengimplementasikan ‘Read’ pada  
aplikasi React Native kamu. Cekidot~

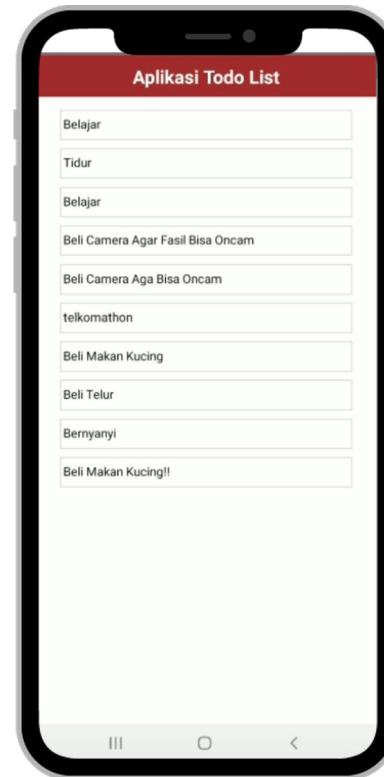




**Briefing itu penting! Yuk simak dulu brief awalnya ya 😊**

Sekarang, kamu akan coba **GET** data menggunakan Axios dan **menampilkan data tersebut** dalam bentuk list item pada React Native. Gambaran hasilnya kamu bisa lihat gambar di samping ya!

Berikut adalah langkah-langkahnya:





1. Langkah pertama, kamu harus **bikin project React Native baru** lebih dulu. Beri nama aplikasimu dengan 'Super Todo App'.

Caranya, ketik perintah:

```
npx react-native init SuperTodoAp.
```



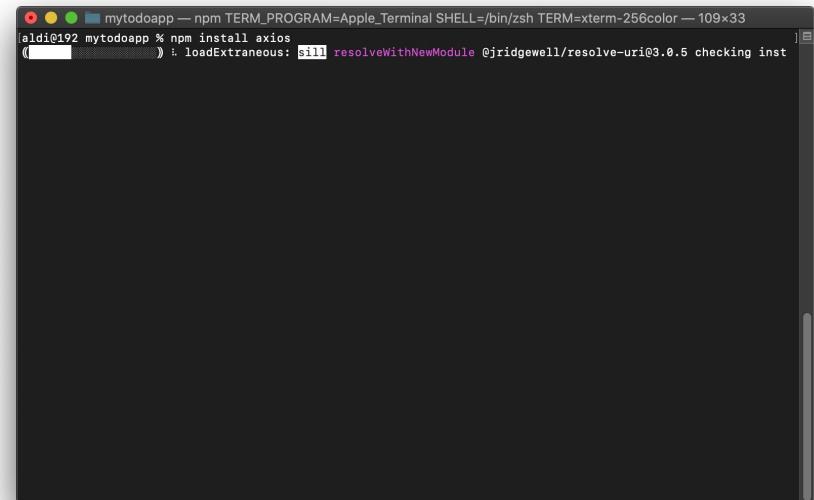
- Setelah project React Native-mu berhasil terinstall, langkah selanjutnya adalah **menginstall Axios pada project React Native-mu** dengan cara mengetik perintah:

```
npm install axios
```

atau kalo kamu pake yarn:

```
yarn add axios
```

Kemudian tunggu hingga proses instalasi Axios pada project React Native-mu selesai.



A screenshot of a terminal window titled 'mytodoapp — npm TERM\_PROGRAM=Apple\_Terminal SHELL=/bin/zsh TERM=xterm-256color — 109x33'. The command 'npm install axios' is being typed. The output shows the progress of the installation, including 'loadExtraneous' and 'silly resolveWithNewModule' logs.



by the way, npm dan yarn adalah sebuah tools package manager yang bisa digunakan pada Javascript untuk mengotomatiskan proses menginstal, memperbarui, mengonfigurasi dan menghapus sebuah package.

Nah di kasus ini, **npm dan yarn berguna untuk mengotomatiskan proses menginstal axios.**





3. Nah, setelah menginstall Axios, sekarang kamu akan **membut component dan tampilan layout** terlebih dahulu.

Pada folder root project, buatlah sebuah folder bernama src, lalu **di dalam folder src tersebut, buatlah sebuah folder lagi bernama components.**

```
App.js
Gemfile
Gemfile.lock
__tests__
_ruby-version
android
app.json
babel.config.js
index.js
ios
metro.config.js
package-lock.json
package.json
src
  components
    TodoList.js
    index.js
.yarn.lock
```



Eits, tunggu dulu!! belum selesai..

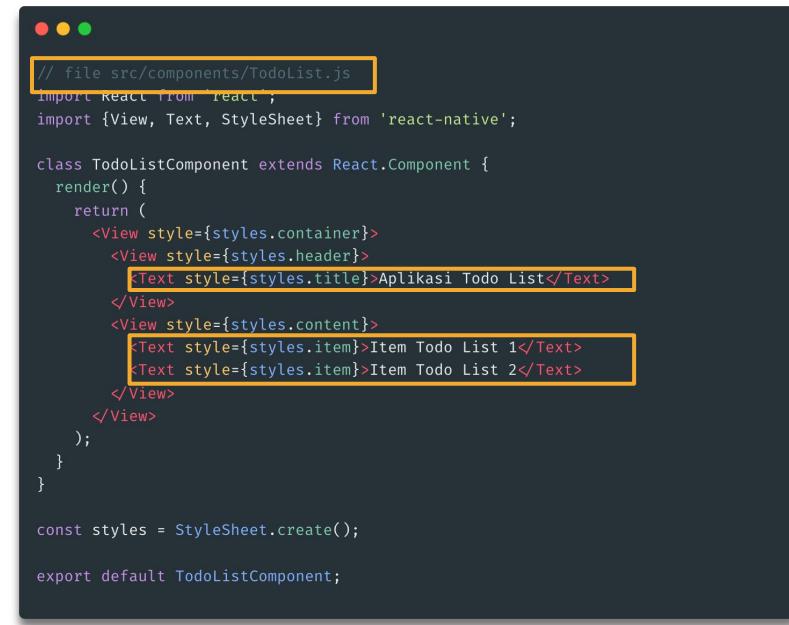
4. Di dalam folder components tadi, **buat sebuah file lagi dengan nama TodoList.js**. Nah, file inilah yang nantinya jadi tempat untuk kamu bikin sebuah **component yang akan menampilkan layout**.



```
App.js
Gemfile
Gemfile.lock
__tests__
_ruby-version
android
app.json
babel.config.js
index.js
ios
metro.config.js
package-lock.json
package.json
src
└── components
    └── TodoList.js
    └── index.js
yarn.lock
```



5. Selanjutnya, pada file `TodoList.js`, buatlah sebuah **component dasar yang akan memunculkan judul dan beberapa list item.**



```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet} from 'react-native';

class TodoListComponent extends React.Component {
  render() {
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>Aplikasi Todo List</Text>
        </View>
        <View style={styles.content}>
          <Text style={styles.item}>Item Todo List 1</Text>
          <Text style={styles.item}>Item Todo List 2</Text>
        </View>
      </View>
    );
  }
}

const styles = StyleSheet.create();

export default TodoListComponent;
```



6. Masih di file yang sama, pada bagian styles di bawah, **tulislah kode styling seperti gambar di samping** untuk membuat tampilan di aplikasi yang sama persis seperti di [slide 21](#) tadi ya, guys!

```
// file src/components/TodoList.js
const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#fff',
  },
  content: {
    paddingHorizontal: 30,
    fontSize: 17,
    marginTop: 10,
  },
  header: {
    backgroundColor: 'brown',
    justifyContent: 'center',
    alignItems: 'center',
    paddingVertical: 10,
  },
  title: {
    fontSize: 20,
    color: '#fff',
    fontWeight: 'bold',
  },
  item: {
    fontSize: 13,
    paddingVertical: 7,
    paddingHorizontal: 5,
    marginVertical: 5,
    color: 'black',
    borderColor: '#d4d4d4',
    borderWidth: 1,
  },
});
```



## Kita balik ke file index.js

7. Lalu pada file `index.js` yang berada di dalam folder `src`, panggil component `TodoListComponent` yang sudah dibuat.

```
// file src/index.js
import React from 'react';
import TodoListComponent from './components/TodoList';

class Index extends React.Component {
  render() {
    return <TodoListComponent />;
  }
}

export default Index;
```



## Habis itu buka file App.js

- Langkah selanjutnya,, pada file App.js yang terdapat pada root project, **import dan panggil component RootIndex** yang telah kamu buat pada step sebelumnya di [slide 29](#) ini..

```
// File App.js
import React from 'react';
import RootIndex from './src/index';

class App extends React.Component {
  render() {
    return <RootIndex />;
  }
}

export default App;
```



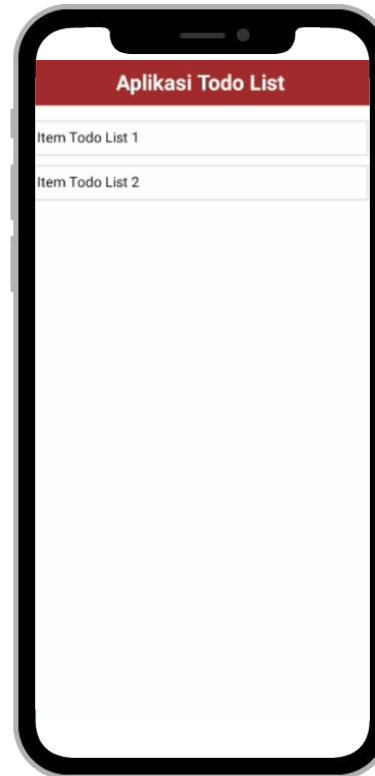
9. Nah, setelah selesai menulis kode stylingnya, **jalankan aplikasimu** dengan cara mengetik perintah berikut pada terminal:

```
npm run android
```

Untuk menjalankan versi iOS, ketikan perintah :

```
npm run ios
```

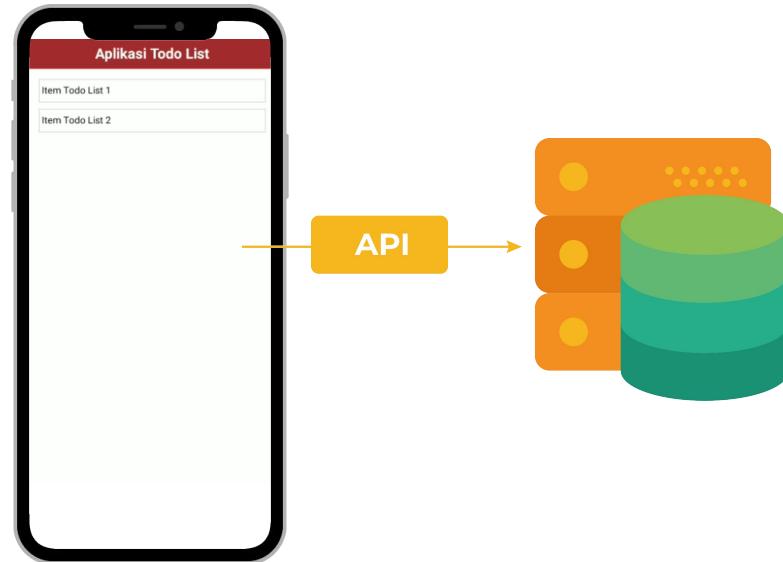
Kalau sudah, maka akan muncul sebuah tampilan persis seperti gambar di samping. Nah, **ini tandanya, kamu udah berhasil bikin layout dasarnya nih, congratssss!!!!**





## Biar aplikasimu nggak kudet, kamu perlu melakukan ini ya~

Langkah selanjutnya adalah membuat agar **item-item pada layout tersebut terkoneksi dengan RESTful API**, supaya kalau ada data yang berubah pada server, maka data yang ditampilkan pada aplikasimu juga akan berubah. Yuk simak caranya berikut ini!





## Panggil Axios di tempat dimana dia dibutuhkan~

Dalam kasus ini, karena tempat membuat list itemnya ada di file `TodoList.js`, jadinya kamu perlu **menggunakan Axios pada file `TodoList.js`. untuk mengambil data dari server.**

```
// file src/components/TodoList.js
import React from "react";
import { View, Text, StyleSheet } from "react-native";
import axios from "axios";

class TodoListComponent extends React.Component {

componentDidMount(){
    // lifecycle mountin componentDidMount
}

render() {
    return (
        <View style={styles.container}>
            <View style={styles.header}>
                <Text style={styles.title}>Aplikasi Todo List</Text>
            </View>
            <View style={styles.content}>
                <Text style={styles.item}>Item Todo List 1</Text>
                <Text style={styles.item}>Item Todo List 2</Text>
            </View>
        </View>
    );
}
const styles = StyleSheet.create({
    // .. kode styling
});
export default TodoListComponent;
```



## Materi Component Lifecycle berkata, “Jangan lupakan aku dong kakak!”

Nah, kamu masih ingat dengan materi Component Lifecycle kan? Kalau lupa, cuss balik pelajari lagi materi tersebut.

Soalnyaaa.. Pada kasus ini kamu akan menggunakan salah satu Component Lifecycle, **yaitu method componentDidMount()**

Terus, kapan ya kita menggunakan method ini? Cekidot~

```
// file src/components/TodoList.js
import React from "react";
import { View, Text, StyleSheet } from "react-native";
import axios from "axios";

class TodoListComponent extends React.Component {
    componentDidMount(){
        // lifecycle mountin componentDidMount
    }

    render() {
        return (
            <View style={styles.container}>
                <View style={styles.header}>
                    <Text style={styles.title}>Aplikasi Todo List</Text>
                </View>
                <View style={styles.content}>
                    <Text style={styles.item}>Item Todo List 1</Text>
                    <Text style={styles.item}>Item Todo List 2</Text>
                </View>
            );
        }
    }

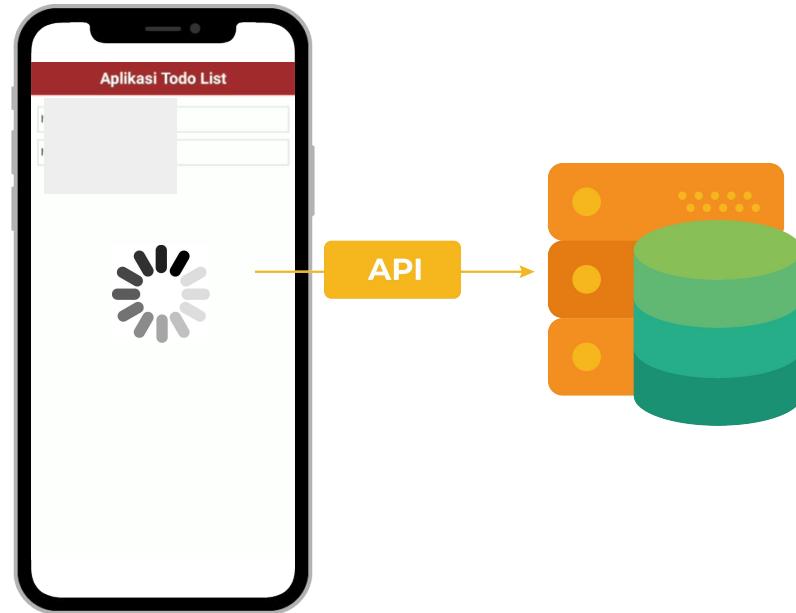
const styles = StyleSheet.create({
    // .. kode styling
});
export default TodoListComponent;
```



### Ini dia jawabannya~

Ketika **aplikasi pertama kali dibuka**, kamu ingin **aplikasi melakukan proses request RESTful API** untuk mendapatkan data paling baru dari server.

Nah, disinilah kamu harus menggunakan salah satu Component Lifecycle, yaitu method `componentDidMount()` untuk bisa melakukan request RESTful API.





## Begini cara penulisan kode nya!

Masukan operasi Axios untuk membaca data pada RESTful API di dalam method `componentDidMount()`

Dengan melakukan ini, aplikasimu akan secara otomatis melakukan HTTP request ke server ketika aplikasi nya dibuka.

```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {

    componentDidMount() {
        axios.get('http://code.aldiipee.com/api/v1/todos').then(responseData => {
            console.log(responseData.data);
        });
    }

    render() {
        return (
            <View style={styles.container}>
                <View style={styles.header}>
                    <Text style={styles.title}>Aplikasi Todo List</Text>
                </View>
                <View style={styles.content}>
                    <Text style={styles.item}>Item Todo List 1</Text>
                    <Text style={styles.item}>Item Todo List 2</Text>
                </View>
            </View>
        );
    }
}

export default TodoListComponent;
```



## Return/hasil dari axios adalah data berbentuk promise

Karena Axios akan me-return/menghasilkan sebuah **data yang berbentuk promise**, maka **kamu bisa menggunakan method .then()** untuk mendapatkan hasil response dari servernya.

```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      console.log(responseData.data);
    });
  }

  render() {
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>Aplikasi Todo List</Text>
        </View>
        <View style={styles.content}>
          <Text style={styles.item}>Item Todo List 1</Text>
          <Text style={styles.item}>Item Todo List 2</Text>
        </View>
      </View>
    );
  }
}

export default TodoListComponent;
```



Nah, **hasil** yang akan didapatkan dari server adalah **data dengan format JSON**.

Struktur data yang didapatkan sama persis dengan yang telah kita lihat pada POSTMAN.





Gambar di samping adalah output dari **hasil variabel responseData.data** yang berasal dari Axios ketika selesai melakukan request ke API.

FYI, tipe data ini adalah tipe data array (dapat dilihat dari **tanda [ ]** di awal dan akhir isi data). informasi ini akan berguna di [slide 51](#)

```
{
  "limit": 10,
  "page": 1,
  "results": [
    {
      "created_at": "2022-03-02T14:04:06.463Z",
      "id": "621f795696e1c8992be7441d",
      "status": "ON_PROGRESS",
      "title": "telkomathon"
    },
    {
      "created_at": "2022-02-26T01:45:23.495Z",
      "id": "62198633c07f275e088dc13e",
      "status": "PENDING",
      "title": "Beli Telur"
    },
    {
      "created_at": "2022-03-02T13:27:00.717Z",
      "id": "621f70a4398b8a991f57cfae",
      "status": "PENDING",
      "title": "Bernyanyi"
    },
    {
      "created_at": "2022-03-02T14:06:05.047Z",
      "id": "621f79cd96e1c8992be74421",
      "status": "PENDING",
      "title": "Beli Makan Kucing!!"
    }
  ],
  "totalPages": 2,
  "totalResults": 16
}
```



Setelah kita dapat data response dari server, langkah selanjutnya kamu harus membuat data ini bisa ditampilkan oleh aplikasi dengan cara rendering.

Bagaimana tuh caranya? Cekidot~





## Biar kamu paham 100%, perhatikan dulu alurnya ya!

Ketika kamu bikin HTTP Request ke server dengan Axios, server akan menerima HTTP Request tersebut dan mengirimkan kembali sebuah response ke Axios.

Gambar di bawah adalah alur ketika server mengirimkan balik sebuah response ke Axios ya!





## Lanjutan alurnya~

Ketika response sudah sampai pada Axios, langkah selanjutnya adalah **mengambil response yang berupa data** tersebut, kemudian **mengupdate state pada component-mu** dengan data yang berasal dari response Axios.

Kenapa harus mengupdate state ya?? Kita pahami lewat analogi di [slide 44](#) yah



## Langkah terakhir~

Setelah state berisi **data response** dari server, langkah yang harus kamu lakukan selanjutnya adalah **merender data-data yang terdapat pada state** tersebut. Kemudian data-data tersebut siap ditampilkan di aplikasimu, guys! Horeeee!!!





## Begini analoginya~

Manajer di perusahaan A (**sebagai client**) meminta persetujuan perjanjian kerja dengan **mengirim sebuah berkas** ke perusahaan B (**sebagai server**) melalui seorang **tukang pos (sebagai Axios)**. Lalu, **perusahaan B menyetujui** perjanjian tersebut dan **mengirimkan kembali berkas** tadi ke perusahaan A **melalui tukang pos yang sama** (Axios).





## Lanjut lagi analoginya~

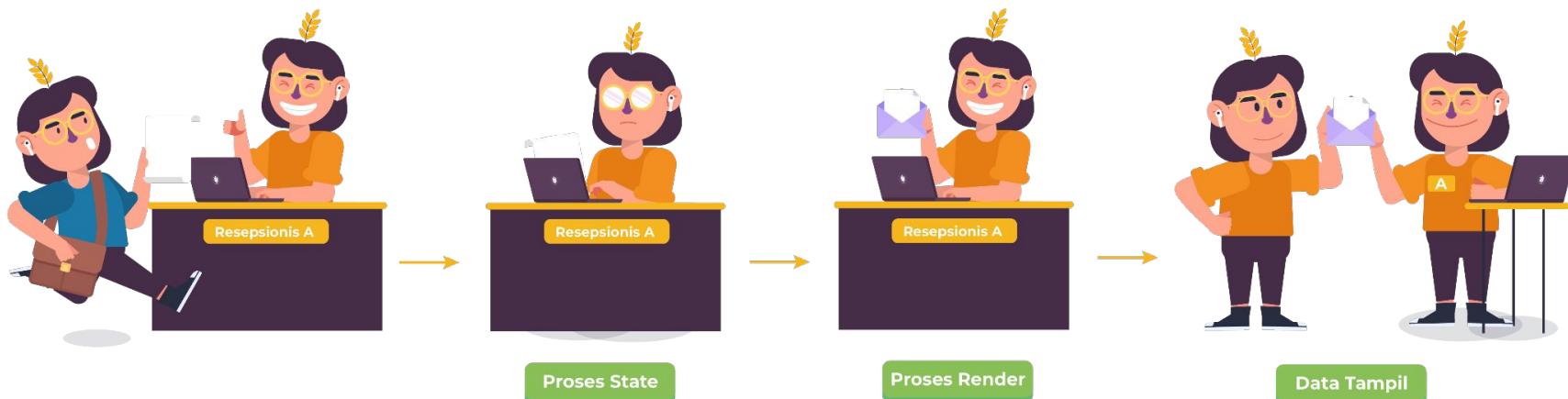
Nah, ketika tukang pos sampai di perusahaan A, **dia nggak bisa langsung memberikan berkas tersebut ke manajer perusahaan A**. Melainkan, berkas tersebut diterima terlebih dahulu oleh **resepsionis perusahaan A (sebagai State)**. **Isi berkas tersebut disalin** oleh resepsionis untuk kemudian **dimasukkan ke dalam amplop agar terlihat rapi** sebelum akhirnya diberikan kepada si manajer.





## Ini dia jawabannya mengapa harus mengupdate state

Nah, **proses menyalin isi berkas oleh resepsionis itu adalah analogi dari proses state**. Sementara proses memasukkan berkas ke dalam amplop adalah analogi dari proses render data di UI. Dan ketika akhirnya berkas itu sampai di tangan si manajer adalah analogi ketika data sudah ditampilkan di dalam sebuah aplikasi.





Tadi kan ketika data response sampai di axios, lalu data tersebut harus diambil dan kita harus mengupdate state pada component. Gimana ya caranya mengupdate state? Yuk simak baik-baik ya!





## Sebelum mulai, ada info penting nih!

FYI, data-data di dalam property results pada gambar di samping **adalah data-data item Task Todo yang perlu kamu tampilkan.**

```
{
  "limit": 10,
  "page": 1,
  "results": [
    {
      "created_at": "2022-03-02T14:04:06.463Z",
      "id": "621f795696e1c8992be7441d",
      "status": "ON_PROGRESS",
      "title": "telkomathon"
    },
    {
      "created_at": "2022-02-26T01:45:23.495Z",
      "id": "62198633c07f275e088dc13e",
      "status": "PENDING",
      "title": "Beli Telur"
    },
    {
      "created_at": "2022-03-02T13:27:00.717Z",
      "id": "621f70a4398b8a991f57cfac",
      "status": "PENDING",
      "title": "Bernyanyi"
    },
    {
      "created_at": "2022-03-02T14:06:05.047Z",
      "id": "621f79cd96e1c8992be74421",
      "status": "PENDING",
      "title": "Beli Makan Kucing!!"
    }
  ],
  "totalPages": 2,
  "totalResults": 16
}
```



## Mari kita mulai~

1. Buat dulu sebuah state pada Component TodoListComponent, lalu beri nama state tersebut todoListData.
2. Atur value dari TodoListData menjadi Array Kosong, yaitu []

```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: [],
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  render() {
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>Applikasi Todo List</Text>
        </View>
        <View style={styles.content}>
          <Text style={styles.item}>Item Todo List 1</Text>
          <Text style={styles.item}>Item Todo List 2</Text>
        </View>
      </View>
    );
  }
}

export default TodoListComponent;
```



## Kenapa value-nya harus array kosong '[ ]'?

Karena **tipe data sebuah state harus seragam** dengan tipe data yang akan dimasukkan pada state tersebut (tipe data yang diterima oleh axios).

```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: [],
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  render() {
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>Aplikasi Todo List</Text>
        </View>
        <View style={styles.content}>
          <Text style={styles.item}>Item Todo List 1</Text>
          <Text style={styles.item}>Item Todo List 2</Text>
        </View>
      );
    }
  }

export default TodoListComponent;
```



Pada kasus ini, property results pada struktur JSON ([lihat FYI di slide 39](#)) memiliki tipe data Array. Oleh karena itu, default value dari state kamu juga harus memiliki tipe data yang sama, yaitu Array '[]'

```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: [],
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  render() {
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>Aplikasi Todo List</Text>
        </View>
        <View style={styles.content}>
          <Text style={styles.item}>Item Todo List 1</Text>
          <Text style={styles.item}>Item Todo List 2</Text>
        </View>
      </View>
    );
  }
}

export default TodoListComponent;
```



## Lanjut ke langkah tiga

3. Selanjutnya, **update data pada state todoListApp** saat Axios berhasil mendapatkan response dari server.

Karena data List yang kamu butuhkan terdapat pada property results, maka kamu harus **memanggil variabel property tersebut dengan responseData.data.results**.

```
// File src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: [],
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      [this.setState({todoListData: responseData.data.results})];
    });
  }

  render() {
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>Aplikasi Todo List</Text>
        </View>
        <View style={styles.content}>
          <Text style={styles.item}>Item Todo List 1</Text>
          <Text style={styles.item}>Item Todo List 2</Text>
        </View>
      </View>
    );
  }
}

export default TodoListComponent;
```



Nah state kamu sudah terupdate dan terisi dengan data yang berasal dari server nih, guys! tapi masih ada satu langkah terakhir yang perlu kamu lakukan nih.

```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: []
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  render() {
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>Aplikasi Todo List</Text>
        </View>
        <View style={styles.content}>
          <Text style={styles.item}>Item Todo List 1</Text>
          <Text style={styles.item}>Item Todo List 2</Text>
        </View>
      </View>
    );
  }
}

export default TodoListComponent;
```



4. Langkah terakhir sebelum data dari server bisa muncul pada aplikasi adalah rendering data/menampilkan data. Begini caranya:

Kamu harus memanggil variable **this.state.todoListData** untuk mengakses property **todoListData**. FYI, property todoListData inilah yang menyimpan data-data yang dari Server.

```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: []
  };

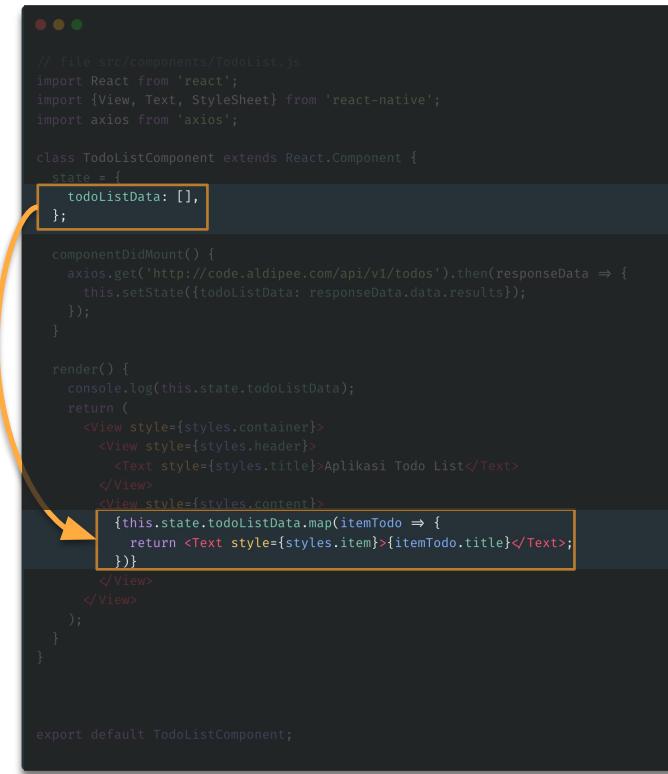
  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  render() {
    console.log(this.state.todoListData);
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>Aplikasi Todo List</Text>
        </View>
        <View style={styles.content}>
          {this.state.todoListData.map(itemTodo => {
            return <Text style={styles.item}>{itemTodo.title}</Text>;
          })}
        </View>
      </View>
    );
  }
}

export default TodoListComponent;
```



5. Karena **bentuk datanya adalah sebuah array**, maka kamu bisa pakai array **method '.map'** untuk melakukan perulangan.



```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = [
    todoListData: []
  ];

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

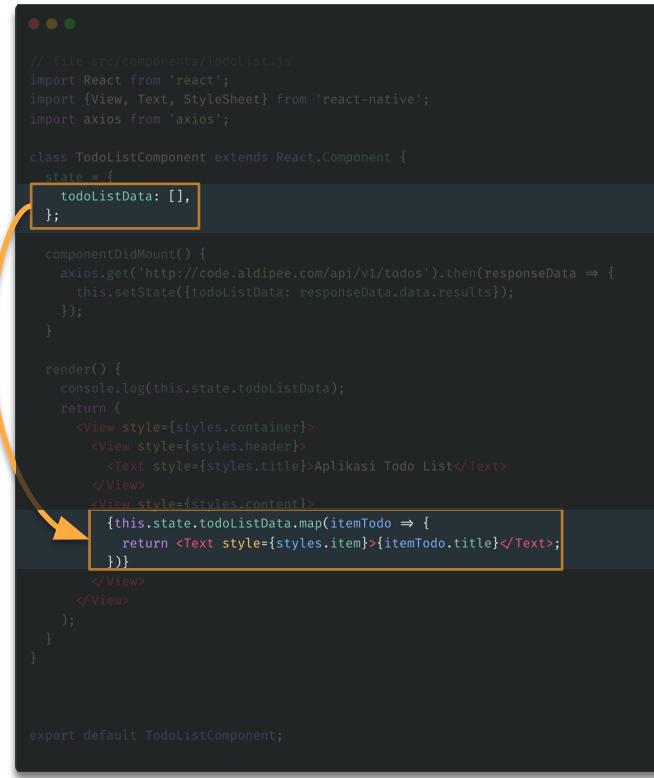
  render() {
    console.log(this.state.todoListData);
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>Aplikasi Todo List</Text>
        </View>
        <View style={styles.content}>
          {this.state.todoListData.map(itemTodo => {
            return <Text style={styles.item}>{itemTodo.title}</Text>;
          })}
        </View>
      </View>
    );
  }
}

export default TodoListComponent;
```



- Setiap data object yang ada di dalam Array akan ditampilkan dengan cara digabung dengan jsx. Kamu akan menggabung variabel tersebut saat perulangan dengan:

```
<Text style={styles.item}>{namaVariabel}</Text>
```



```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: []
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  render() {
    console.log(this.state.todoListData);
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>Aplikasi Todo List</Text>
        </View>
        <View style={styles.content}>
          {this.state.todoListData.map(itemTodo => {
            return <Text style={styles.item}>{itemTodo.title}</Text>;
          })}
        </View>
      </View>
    );
  }
}

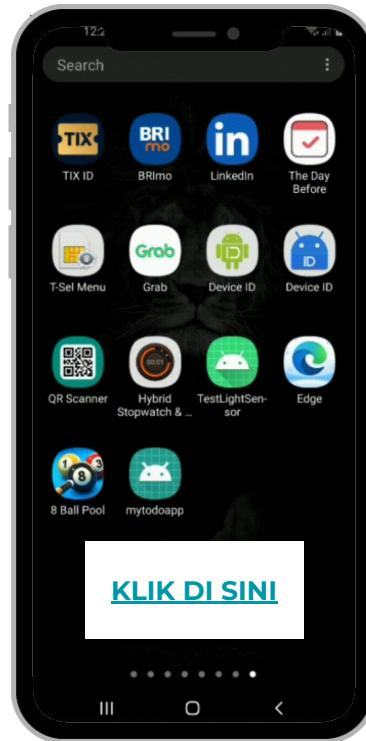
export default TodoListComponent;
```



Nah, setelah berhasil melakukan rendering data/menampilkan data, sekarang aplikasimu berhasil menampilkan data-data yang terdapat pada server.

**Pada tahap inilah kamu sudah dianggap berhasil mengintegrasikan aplikasi dengan API.**

Yeayy, congratss!!





Ke laut liat paus, lanjut teroooss~

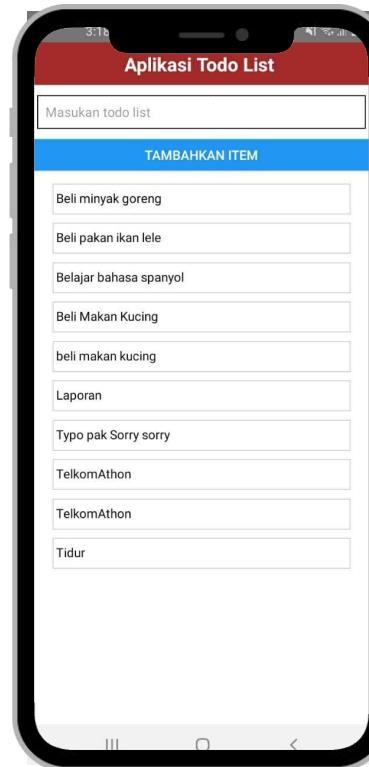
Sekarang kita pelajari cara  
mengimplementasikan 'Create' melalui  
proses bikin data pada server lewat  
aplikasi React Native-mu. Siappp??





## Jadi bagaimana nih caranya bikin data pada server lewat React Native?

Garis besarnya, kamu harus **bikin sebuah kolom Text Input** sederhana dan **sebuah Button** supaya kamu bisa menulis data yang mau kamu kirim ke server.





## Mari kita mulai~

1. Langkah pertama yang harus kamu lakukan adalah **membuat kolom Text Input** terlebih dahulu.

Untuk menampilkan kolom Text Input dan Button, kamu harus **import dulu Component TextInput dan Button** dari React Native, lalu **tulis** kedua component tersebut **di tengah-tengah kode jsx**.

```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet, Alert, TextInput, Button} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: []
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  render() {
    console.log(this.state.todoListData);
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>
            Aplikasi Todo List {this.state.textInput}
          </Text>
        </View>
        <View>
          <TextInput style={styles.input} placeholder="Masukan todo list" />
          <Button title="Tambahkan item" />
        </View>
        <View style={styles.content}>
          {this.state.todoListData.map(itemTodo => {
            return <Text style={styles.item}>{itemTodo.title}</Text>;
          })}
        </View>
      </View>
    );
  }
}

export default TodoListComponent;
```



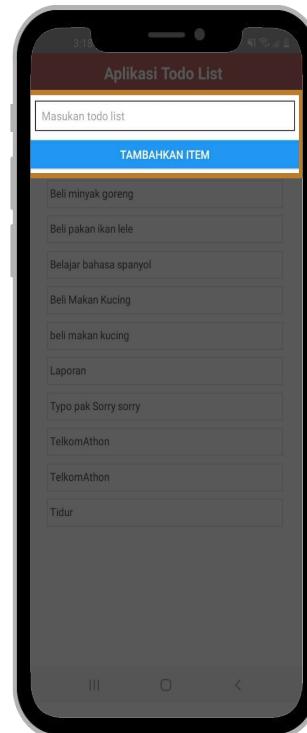
2. Kemudian **pada barisan kode styling**, tambahkan styling yang akan digunakan oleh component TextInput.

```
const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#fff',
  },
  content: {
    paddingHorizontal: 30,
    fontSize: 17,
    marginTop: 10,
  },
  header: {
    backgroundColor: 'brown',
    justifyContent: 'center',
    alignItems: 'center',
    paddingVertical: 10,
  },
  title: {
    fontSize: 20,
    color: '#fff',
    fontWeight: 'bold',
  },
  item: {
    fontSize: 13,
    paddingVertical: 7,
    paddingHorizontal: 4,
    marginVertical: 5,
    color: 'black',
    borderColor: '#d4d4d4',
    borderWidth: 1,
  },
  input: {
    height: 40,
    margin: 12,
    borderWidth: 1,
    padding: 10,
  },
});
```



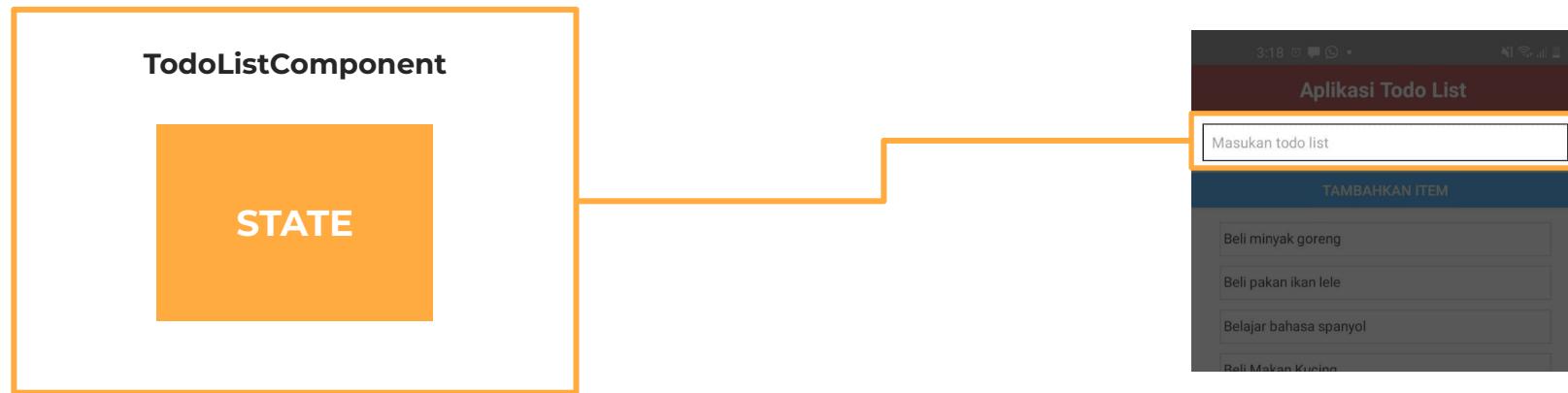
Setelah penambahan styling, maka akan muncul tampilan kolom Text Input dan sebuah Button pada layout aplikasi React Native-mu.

3. Selanjutnya kamu akan coba untuk menangkap inputan text yang akan diketik pada TextInput. Caranya, **kirim hasil ketikan user** tersebut ke server melalui RESTful API dengan menggunakan method POST.



4. Pada dasarnya, setiap teks yang diketik oleh user pada sebuah kolom Text Input, harus kamu simpan di dalam sebuah state terlebih dahulu ya.

Misalnya saat user mengetik text “Meonggg” pada kolom Text Input, maka text tersebut harus kamu simpan di dalam sebuah state pada component.





5. Pada state, **tambahkan satu property** lagi dengan nama `textInput`.

Property ini nantinya yang akan bertanggung jawab untuk **menyimpan teks yang diketik oleh user** pada kolom Text Input di aplikasi.

```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet, Alert, TextInput, Button} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: [],
    textInput: ''
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  render() {
    console.log(this.state.todoListData);
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>
            Aplikasi Todo List {this.state.textInput}
          </Text>
        </View>
        <View>
          <TextInput style={styles.input} placeholder="Masukan todo list" />
          <Button title="Tambahkan item" />
        </View>
        <View style={styles.content}>
          {this.state.todoListData.map(itemTodo => {
            return <Text style={styles.item}>{itemTodo.title}</Text>;
          })}
        </View>
      </View>
    );
  }
}

export default TodoListComponent;
```



## Textinput sudah selesai, lanjut ke fungsi onChangeText

6. Yak, setelah berhasil bikin TextInput, kamu akan bikin sebuah function dengan nama onChangeText. Caranya, Pada Component TextInput, masukan function yang telah kamu buat pada props onChangeText.

```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet, Alert, TextInput, Button} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: [],
    textInput: ''
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  onChangeText = (typingValue) => {
    this.setState({textInput: typingValue});
  }

  render() {
    console.log(this.state.todoListData);
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>
            Aplikasi Todo List {this.state.textInput}
          </Text>
        </View>
        <View>
          <TextInput style={styles.input} placeholder="Masukan todo list" onChangeText={this.onChangeText} />
          <Button title="Tambahkan item" />
        </View>
        <View style={styles.content}>
          {this.state.todoListData.map(itemTodo => {
            return <Text style={styles.item}>{itemTodo.title}</Text>;
          })}
        </View>
      </View>
    );
  }
}

export default TodoListComponent;
```



Function onChangeText ini **bertugas untuk mengupdate state setiap kali user mengetik** sebuah huruf, sehingga setiap teks yang diketik oleh User akan tersimpan pada state TextInput.

```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet, Alert, TextInput, Button} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: [],
    textInput: ''
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  onChangeText = (typingValue) => {
    this.setState({textInput: typingValue});
  }

  render() {
    console.log(this.state.todoListData);
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>
            Aplikasi Todo List {this.state.textInput}
          </Text>
        </View>
        <View>
          <TextInput style={styles.input} placeholder="Masukan todo list" onChangeText={this.onChangeText} />
          <Button title="Tambahkan item" />
        </View>
        <View style={styles.content}>
          {this.state.todoListData.map(itemTodo => {
            return <Text style={styles.item}>{itemTodo.title}</Text>;
          })}
        </View>
      </View>
    );
  }
}

export default TodoListComponent;
```



7. Untuk melakukan tes apakah state yang kamu buat telah sinkron dengan component TextInput atau belum, kamu bisa **memanggil dan menampilkan state TextInput pada kode jsx.**

Render variabel `this.state..textInput`pada kode jsx.

```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet, Alert, TextInput, Button} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: [],
    textInput: ''
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  onChangeText = (typingValue) => {
    this.setState({textInput: typingValue});
  }

  render() {
    console.log(this.state.todoListData);
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>
            Aplikasi Todo List [this.state.textInput]
          </Text>
        </View>
        <View>
          <TextInput style={styles.input} placeholder="Masukan todo list" onChangeText={this.onChangeText} />
          <Button title="Tambahkan item" />
        </View>
        <View style={styles.content}>
          {this.state.todoListData.map(itemTodo => {
            return <text style={styles.item}>{itemTodo.title}</text>;
          })}
        </View>
      </View>
    );
  }
}

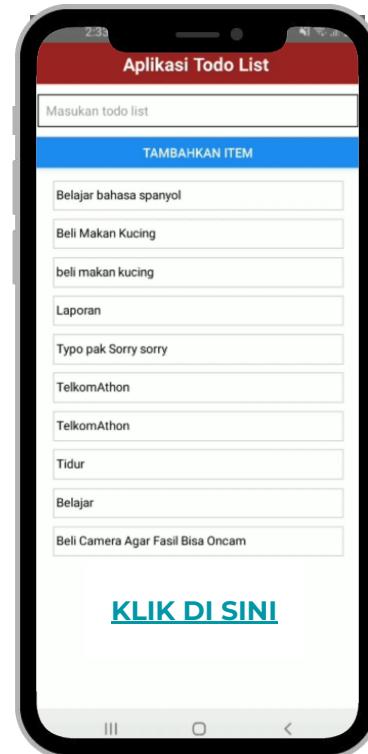
export default TodoListComponent;
```



## Yeaayy jadiii!

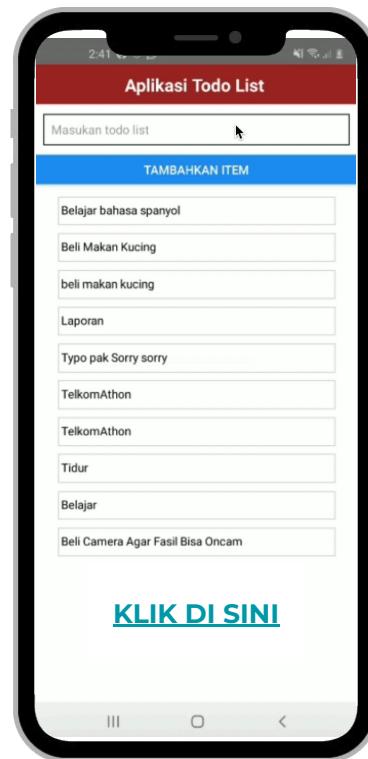
Nah, sekarang saat user mengetik sesuatu pada TextInput, maka setiap teks yang diketik akan muncul pada tampilan UI deh.

Tapi kalau ketikan user nggak tampil pada tampilan UI aplikasimu, berarti ada yang salah dengan source code mu.





Langkah selanjutnya adalah mengirim hasil ketikan user ke server, melalui RESTful API dengan menggunakan HTTP Method POST. Agar hasil ketikan user bisa masuk ke tampilan list item pada aplikasimu.





- Kamu akan membuat function yang bertugas mengambil data hasil ketikan user pada state, dan mengirimkan data tersebut ke server dengan menggunakan Axios.

Kita kasih nama function ini '**onKirimData**'.

```
... ...
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet, Alert, TextInput, Button} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: [],
    textInput: ''
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  onChangeText = (typingValue) => {
    this.setState({textInput: typingValue});
  }

  onKirimData = () =>{
    const dataObjectYangAkanDikirim = {
      title: this.state.textInput,
      description: this.state.textInput,
      status: 'PENDING'
    }
    axios.post('http://code.aldipee.com/api/v1/todos', dataObjectYangAkanDikirim).then(responseServer =>{
      console.log(responseServer)
    })
  }

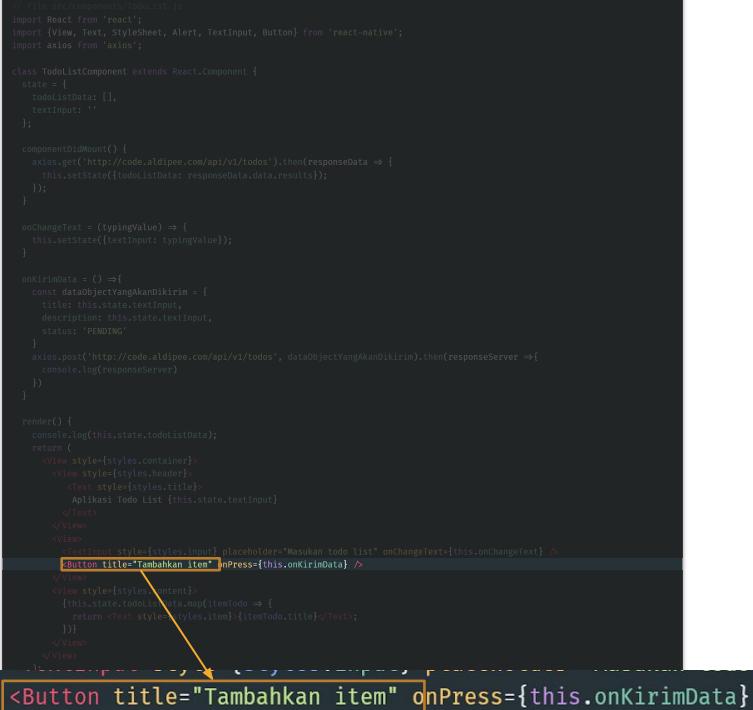
  render() {
    ...
  }
}

onKirimData = () =>{
  const dataObjectYangAkanDikirim = {
    title: this.state.textInput,
    description: this.state.textInput,
    status: 'PENDING'
  }
  axios.post('http://code.aldipee.com/api/v1/todos', dataObjectYangAkanDikirim).then(responseServer =>{
    console.log(responseServer)
  })
}

export default TodoListComponent;
```



12. Karena kamu ingin pengiriman data dilakukan saat user menekan button “Tambahkan Item”, maka **masukan function yang bertugas mengirimkan data ke server** yaitu function `onKirimData` pada component `<Button />`.



```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet, Alert, TextInput, Button} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: [],
    textInput: ''
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  onChangeText = (typingValue) => {
    this.setState({textInput: typingValue});
  }

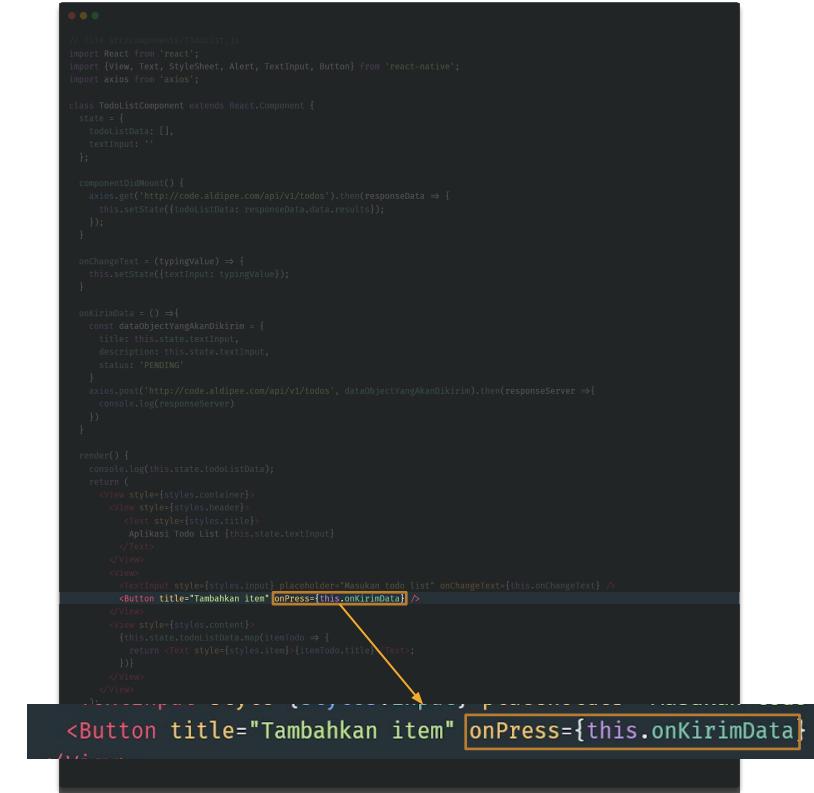
  onKirimData = () => {
    const dataObjectYangAkanDiririm = {
      title: this.state.textInput,
      description: this.state.textInput,
      status: 'PENDING'
    }
    axios.post('http://code.aldipee.com/api/v1/todos', dataObjectYangAkanDiririm)
    .then(responseServer => {
      console.log(responseServer)
    })
  }

  render() {
    console.log(this.state.todoListData);
    return (
      <View style={styles.container}>
        <View style={styles.header}>
          <Text style={styles.title}>
            Aplikasi Todo List {this.state.textInput}
          </Text>
        </View>
        <View>
          <TextInput style={styles.input1} placeholder="Masukan todo list" onChangeText={this.onChangeText} />
          <Button title="Tambahkan item" onPress={this.onKirimData} />
        </View>
        <View style={styles.content}>
          {this.state.todoListData.map(itemTodo => {
            return <Text style={styles.item}>{itemTodo.title}</Text>;
          })}
        </View>
      </View>
    );
  }
}
```



### 13. Masukan function tersebut sebagai value pada props

**onPress.** Artinya, function onKirimData hanya akan dieksekusi ketika user menekan button “Tambahkan Item”



```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet, Alert, TextInput, Button} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: [],
    textInput: ''
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  onChangeText = (typingValue) => {
    this.setState({textInput: typingValue});
  }

  onKirimData = () => {
    const dataObjectYangAkanDiririm = {
      title: this.state.textInput,
      description: this.state.textInput,
      status: 'PENDING'
    }
    axios.post('http://code.aldipee.com/api/v1/todos', dataObjectYangAkanDiririm).then(responseServer => {
      console.log(responseServer)
    })
  }

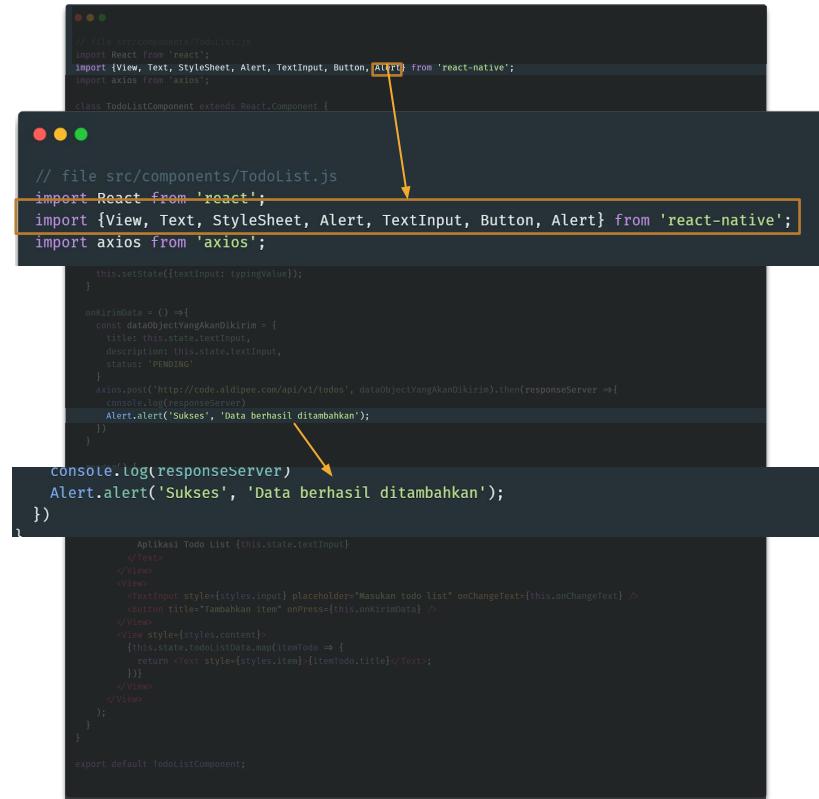
  render() {
    console.log(this.state.todoListData);
    return (
      <View style={styles.container}>
        <Text style={styles.header}>
          <Text style={styles.title}>
            Aplikasi Todo List {this.state.textInput}
          </Text>
        </Text>
        <View style={styles.content}>
          {this.state.todoListData.map(itemTodo => {
            return <Text style={styles.item}>{itemTodo.title}</Text>;
          })}
        </View>
      </View>
    );
  }
}

export default TodoListComponent;
```

<Button title="Tambahkan item" onPress={this.onKirimData}></Button>

14. Setelah itu kamu akan **menambahkan tampilan Alert**. Alert berfungsi untuk menampilkan pop up jika data kita berhasil dan tersimpan pada server.

Untuk menggunakan Alert, kamu harus mengimport Alert dulu dari React Native.



```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet, Alert, TextInput, Button, Axios} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
    // file src/components/TodoList.js
    import React from 'react';
    import {View, Text, StyleSheet, Alert, TextInput, Button} from 'react-native';
    import axios from 'axios';

    this.setState({textInput: typingValue});
}

onKirimData = () => {
    const dataObjectYangAkanDikirim = {
        title: this.state.textInput,
        description: this.state.textInput,
        status: 'PENDING'
    }
    axios.post('http://code.alidpee.com/api/v1/todos', dataObjectYangAkanDikirim).then(responseServer => {
        console.log(responseServer);
        Alert.alert('Sukses', 'Data berhasil ditambahkan');
    })
}

console.log(responseServer)
Alert.alert('Sukses', 'Data berhasil ditambahkan');

Applikasi Todo List {this.state.textInput}
</Text>
</View>
<View>
<TextInput style={styles.input} placeholder="Masukan todo list" onChangeText={this.onChangeText} />
<Button title="Tambahkan item" onPress={this.onKirimData} />
</View>
<View style={styles.content}>
{this.state.todoListData.map(itemTodo => [
    return <text style={styles.item}>{itemTodo.title}</Text>;
])
}
</View>
</View>
};

export default TodoListComponent;
```



## 15. Jika sudah, panggil method Alert.alert di dalam method .then() pada request POST Axios.

Nantinya kode-kode di dalam method .then() hanya akan tereksekusi jika Request POST-mu berhasil, alias datamu sudah tersimpan di server.

```
file:///Users/aldi/Downloads/TodoList.js
import React from 'react';
import {View, Text, StyleSheet, Alert, TextInput, Button, Alert} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todolistData: [],
    textInput: ''
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todolistData: responseData.data.results});
    });
  }

  onChangeText = (typingValue) => {
    this.setState({textInput: typingValue});
  }

  onKirimData = () => {
    const dataObjectYangAkanDikirim = {
      title: this.state.textInput,
      description: this.state.textInput,
      status: 'PENDING'
    }
    axios.post('http://code.aldipee.com/api/v1/todos', dataObjectYangAkanDikirim).then(responseServer =>{
      console.log(responseServer)
      Alert.alert('Sukses', 'Data berhasil ditambahkan');
    })
  }

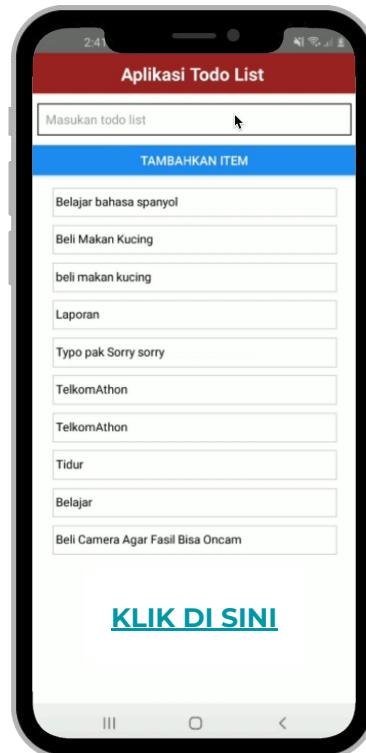
  render() {
    console.log(this.state.todolistData);
    return (
      <TextInput style={styles.input} placeholder="Masukan todo list" onChangeText={this.onChangeText} />
      <Button title="Tambahkan item" onPress={this.onKirimData} />
      <View style={styles.content}>
        {this.state.todolistData.map(itemTodo => [
          return <text style={styles.item}>{itemTodo.title}</text>;
        ])}
      </View>
    );
  }
}

export default TodoListComponent;
```



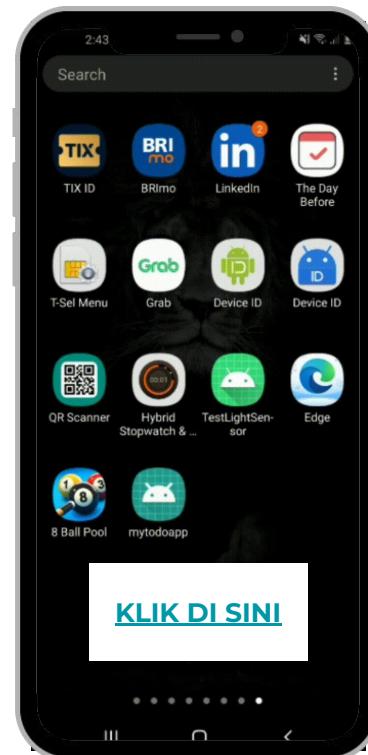
16. Jangan lupa save file kodingannya yaaa..

Nah, sekarang kamu bisa tes aplikasimu dengan bikin sebuah data.





Kalau kamu buka kembali aplikasi dari awal, maka data yang kamu input pada TextInput sebelumnya akan muncul pada list yang ada di bawah Button.





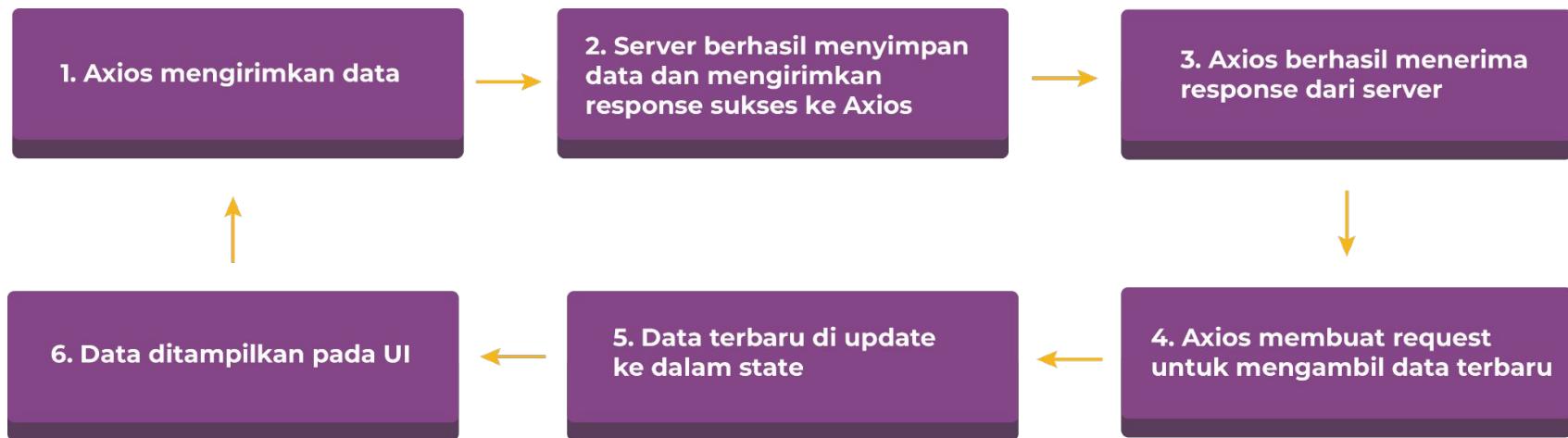
**Meskipun sudah berhasil bikin data, tapi masih ada problem ini guys.**

**Masa iya setiap kali user menambahkan data, user harus tutup aplikasi dulu, baru data yang baru bisa muncul?**

**Nggak user friendly banget, ah. Terus bagaimana dong solusinya?**



Nah untuk mengatasi hal ini, solusinya adalah kamu harus mengambil data dari server lagi setiap kali kita berhasil menambahkan data baru. Kayak gini nih flow nya:





Nah, sekarang kamu akan menambahkan request baru untuk mendapatkan update data terbaru setiap kali kamu berhasil mengirim data pada server.

Kode yang akan kamu tuliskan sama persis dengan saat kamu request data pada awal aplikasi dibuka.

Kamu tinggal **copy saja kode yang ada di dalam `componentDidMount`, lalu paste di dalam method `then()`, pada `axios.post`.**

```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet, Alert, TextInput, Button, Alert} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: [],
    textInput: ''
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  onChangeText = (typingValue) => {
    this.setState({textInput: typingValue});
  }

  onKirimData = () => {
    const dataObjectYangAkanDikirim = {
      title: this.state.textInput,
      description: this.state.textInput,
      status: 'PENDING'
    }

    axios.post('http://code.aldipee.com/api/v1/todos', dataObjectYangAkanDikirim).then(responseServer => {
      console.log(responseServer);
      Alert.alert('Sukses', 'Data berhasil ditambahkan');
      axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
        this.setState({todoListData: responseData.data.results});
      });
    })
  }

  render() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }
}

<TextInput style={styles.input} placeholder="Masukan todo list" onChangeText={this.onChangeText} />
<Button title="Tambahkan item" onPress={this.onKirimData} />
<View>
  <View style={styles.content}>
    {this.state.todoListData.map(itemTodo => {
      return <Text style={styles.item}>{itemTodo.title}</Text>;
    })}
  </View>
</View>
}

export default TodoListComponent;
```



Sederhananya, kamu akan melakukan Request GET data setiap kali kamu berhasil melakukan POST data.

Apa tujuannya? Untuk memperbarui data yang tersimpan pada state setiap kali kamu berhasil mengirimkan data baru.

```
// file src/components/TodoList.js
import React from 'react';
import {View, Text, StyleSheet, Alert, TextInput, Button, Alert} from 'react-native';
import axios from 'axios';

class TodoListComponent extends React.Component {
  state = {
    todoListData: [],
    textInput: ''
  };

  componentDidMount() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }

  onChangeText = (typingValue) => {
    this.setState({textInput: typingValue});
  }

  onKirimData = () => {
    const dataObjectYangAkanDikirim = {
      title: this.state.textInput,
      description: this.state.textInput,
      status: 'PENDING'
    }

    axios.post('http://code.aldipee.com/api/v1/todos', dataObjectYangAkanDikirim).then(responseServer => {
      console.log(responseServer);
      Alert.alert('Sukces', 'Data berhasil ditambahkan');
      axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
        this.setState({todoListData: responseData.data.results});
      });
    })
  }

  render() {
    axios.get('http://code.aldipee.com/api/v1/todos').then(responseData => {
      this.setState({todoListData: responseData.data.results});
    });
  }
}

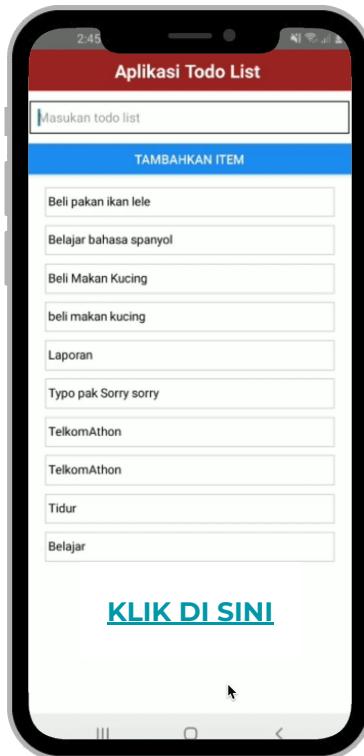
<TextInput style={styles.input} placeholder="Masukan todo list" onChangeText={this.onChangeText} />
<Button title="Tambahkan item" onPress={this.onKirimData} />
</View>
<View style={styles.content}>
  {this.state.todoListData.map(itemTodo => {
    return <Text style={styles.item}>{itemTodo.title}</Text>;
  })}
</View>
</View>
}

export default TodoListComponent;
```



Nah, sekarang setiap kali menambahkan data baru, list data akan terupdate dengan data terbaru dehhhhh..

**FINISH!**





**Yeaayy!! Selamat yaa, guys! Akhirnya kamu berhasil menuntaskan Topik 5 tentang CRUD ini. Biar makin jago, yuk kerjain quiz berikut ini!**





# Saatnya kita Quiz!





## 1. Apa yang dimaksud dengan Integrasi Aplikasi dengan API?

- A. Membuat Aplikasi kita terkoneksi dengan server melalui RESTful API
- B. Install Axios di aplikasi
- C. Membuat animasi pada aplikasi yang terkoneksi data dinamis



## 1. Apa yang dimaksud dengan Integrasi Aplikasi dengan API?

- A. Membuat Aplikasi kita terkoneksi dengan server melalui RESTFul API
- B. Install Axios di aplikasi
- C. Membuat animasi pada aplikasi yang terkoneksi data dinamis

Integrasi API maksudnya adalah membuat aplikasi kita dapat berinteraksi dengan server melalui RESTFul API



## 2. Apa perbedaan antara Postman dan Axios?

- A. Postman adalah HTTP Client yang di install dan digunakan pada Desktop, sedangkan Axios adalah HTTP Client ang di install dan digunakan pada source code sebuah project
- B. Postman adalah HTTP Client yang di install dan digunakan pada source code sebuah project, sedangkan Axios adalah HTTP Client yang di install dan digunakan pada Desktop
- C. Postman dapat digunakan pada Desktop dan Source code sebuah project, sedangkan axios hanya bisa digunakan di Source Code Project saja.



## 2. Apa perbedaan antara Postman dan Axios?

- A. Postman adalah HTTP Client yang di install dan digunakan pada Desktop, sedangkan Axios adalah HTTP Client ang di install dan digunakan pada source code sebuah project
- B. Postman adalah HTTP Client yang di install dan digunakan pada source code sebuah project, sedangkan Axios adalah HTTP Client yang di install dan digunakan pada Desktop
- C. Postman dapat digunakan pada Desktop dan Source code sebuah project, sedangkan axios hanya bisa digunakan di Source Code Project saja.



3. Jono ingin mengirimkan data penjualan dari aplikasi react native yang telah di buat olehnya, method HTTP yang jono harus gunakan agar bisa mengirimkan data tersebut ke server melalui RESTFul API adalah?
- A. PUT
  - B. PATCH
  - C. POST



**3. Jono ingin mengirimkan data penjualan dari aplikasi react native yang telah di buat olehnya, method HTTP yang jono harus gunakan agar bisa mengirimkan data tersebut ke server melalui RESTFul API adalah?**

- A. PUT
- B. PATCH
- C. POST

Method POST selalu digunakan untuk menambahkan/memasukan data baru pada Server melalui RESTFul API



**4. Setelah pada operasi axios.get, setelah axios berhasil mendapatkan data dari server, Langkah apa yang diperlukan selanjutnya agar data tersebut dapat ditampilkan di UI Aplikasi?**

- A. Data tersebut langsung di tampilkan saja dari response axios
- B. Data tersebut disimpan didalam state terlebih dahulu, kemudian JSX akan mengambil data tersebut Ketika sudah disimpan di dalam state
- C. Data tersebut akan langsung di render oleh JSX dalam bentuk Promise



**4. Setelah pada operasi axios.get, setelah axios berhasil mendapatkan data dari server, Langkah apa yang diperlukan selanjutnya agar data tersebut dapat ditampilkan di UI Aplikasi?**

- A. Data tersebut langsung di tampilkan saja dari response axios
- B. Data tersebut disimpan didalam state terlebih dahulu, kemudian JSX akan mengambil data tersebut Ketika sudah disimpan di dalam state
- C. Data tersebut akan langsung di render oleh JSX dalam bentuk Promise

**Setiap hasil data yang didapatkan oleh server akan disimpan di state komponen tersebut terlebih dahulu, lalu akan di render oleh jsx. Jsx akan mengambil data yang berasal dari state.**

# Terima Kasih!



Next Topic

loading...