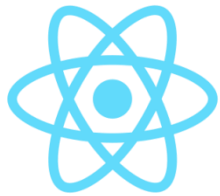


Front-End Development



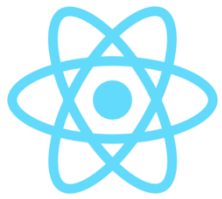
React

#3 Learn Once, Write Anywhere



API

- **API (Application Programming Interface)** is a software intermediary which allows applications to talk to each other.
- When you use an application on your mobile phone, the application connects to the Internet and sends data to a server. The server then retrieves that data, interprets it, performs the necessary actions and sends it back to your phone. The application then interprets that data and presents to you with the information that you wanted, in a readable way. This is what an API is - all of this happens via API.



API





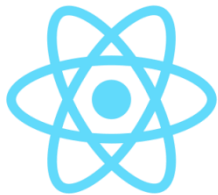
HTTP Methods

- ***HTTP (The Hypertext Transfer Protocol)*** is designed to enable communications between clients & servers. It works as a request & response protocol between a client & server.
- A web browser may be the client, and an application on a computer that hosts a web site may be the server.
- The most commonly used HTTP Methods are POST, GET, PUT, PATCH & DELETE.



RESTful APIs

- ***RESTful (Representational State Transfer)*** web services is a way of providing interoperability between computer system on the internet.
- A RESTful API is an application program interface that uses HTTP requests to GET, PUT, POST or UPDATE data, based on representational state transfer (RESTful) architecture technology.



JSON

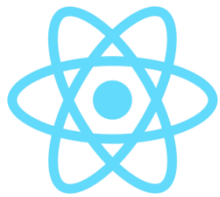
■ JSON (*Javascript Object Notation*)

is a lightweight data-interchange format that based on a subset of JS. It's easy to read, write and can be used with any modern language.

JSON is not a JavaScript Object!

JSON rules:

- Uses property/value pairs = `{"nama" : "Andi"}`
- Uses dblquote on its prop & val (ex number)
- Must use specified data type
- File type is `".json"`
- Mime type is `"application/json"`



Validate JSON

jsonlint.com

```
1 {  
2   "nama": "Andi",  
3   "usia": 24,  
4   "job": "PNS"  
5 }
```

Validate JSON Clear

Results

valid JSON

```
1 {  
2   nama: "Andi",  
3   usia: 24,  
4   job: "PNS"  
5 }
```

Validate JSON Clear

Results

Error: Parse error on line 1:
{ nama: "Andi", usia:
--^
Expecting 'STRING', '}', got 'undefined'

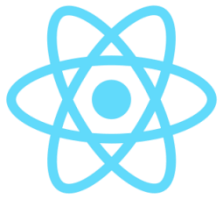


Object → JSON

```
var orang = {  
  nama: "Andi",  
  usia: 21,  
  asal: "Lampung"  
}
```

```
orang = JSON.stringify(orang);
```

```
console.log(orang);  
console.log(orang.nama);
```

JSON → Object

```
var orang = {  
  nama: "Andi",  
  usia: 21,  
  asal: "Lampung"  
}
```

```
orang = JSON.stringify(orang);  
orang = JSON.parse(orang);
```

```
console.log(orang);  
console.log(orang.nama);
```



- A powerful GUI platform to make API development faster & easier, from building API requests through testing, documentation & sharing.
- Postman has features for every API developer: request building, tests & pre-request scripts, variables, environments & request descriptions, designed to work seamlessly together.
- Download from getpostman.com

GET all data from *<https://facebook.github.io/react-native/movies.json>*



```
{
  "title": "The Basics - Networking",
  "description": "Your app fetched this from a remote endpoint!",
  "movies": [
    { "title": "Star Wars", "releaseYear": "1977"},
    { "title": "Back to the Future", "releaseYear": "1985"},
    { "title": "The Matrix", "releaseYear": "1999"},
    { "title": "Inception", "releaseYear": "2010"},
    { "title": "Interstellar", "releaseYear": "2014"}
  ]
}
```

Try on Postman!



Using Axios Package

Axios is an NPM built-in package that allows us to:

- Make http requests from node.js
- Supports the Promise API
- Intercept request and response
- Transform request and response data
- Cancel requests
- Automatic transforms for JSON data

Install Axios package:

```
$ npm install axios --save
```

```
import React, { Component } from 'react';  
import axios from 'axios';
```

```
class App extends Component {
```

```
  componentDidMount(){
```

```
    axios.get('https://facebook.github.io/react-native/movies.json')  
      .then((ambilData) => {  
        console.log(ambilData);  
      })  
  };
```

```
  render() {  
    return (  
      <div>  
        <h1>Coba Get Data</h1>  
      </div>  
    );  
  }
```

```
}  
export default App;
```

Axios

#1 Get All Data

Axios

#1 Get All Data

← → ↻ ⓘ localhost:3000

Coba Get Data

Elements Console Sources Network Performance Memory

top ▼ Filter Default levels ▼

```
▼ {data: {...}, status: 200, statusText: "", headers: {...}, config: {...}, ...}
  ► config: {adapter: f, transformRequest: {...}, transformResponse: {...}, ...}
  ▼ data:
    description: "Your app fetched this from a remote endpoint!"
    ▼ movies: Array(5)
      ► 0: {title: "Star Wars", releaseYear: "1977"}
      ► 1: {title: "Back to the Future", releaseYear: "1985"}
      ► 2: {title: "The Matrix", releaseYear: "1999"}
      ► 3: {title: "Inception", releaseYear: "2010"}
      ► 4: {title: "Interstellar", releaseYear: "2014"}
      length: 5
      ► __proto__: Array(0)
    title: "The Basics - Networking"
    ► __proto__: Object
  ► headers: {last-modified: "Thu, 28 Dec 2017 23:06:47 GMT", content-t...}
  ► request: XMLHttpRequest {onreadystatechange: f, readyState: 4, time...
```

```
import React, { Component } from 'react';
import axios from 'axios';
```

```
class App extends Component {
```

```
  componentDidMount(){
```

```
    axios.get('https://facebook.github.io/react-native/movies.json')
```

```
    .then((ambilData) => {
```

```
      console.log(ambilData.data.title);
```

```
      console.log(ambilData.data.description);
```

```
      console.log(ambilData.data.movies);
```

```
    })
```

```
  };
```

```
  render() {
```

```
    return (
```

```
      <div>
```

```
        <h1>Coba Get Data</h1>
```

```
      </div>
```

```
    );
```

```
  }
```

```
}
```

```
export default App;
```

Axios

#2 Get Specific Data

Axios

#2 Get Specific Data

← → ↻ ⓘ localhost:3000

Coba Get Data

🔍 📄 | Elements Console Sources Network Performance

🚫 | top ▼ | Filter Default

The Basics - Networking

Your app fetched this from a remote endpoint!

▼ (5) [{...}, {...}, {...}, {...}, {...}] ⓘ

- ▶ 0: {title: "Star Wars", releaseYear: "1977"}
- ▶ 1: {title: "Back to the Future", releaseYear: "1985"}
- ▶ 2: {title: "The Matrix", releaseYear: "1999"}
- ▶ 3: {title: "Inception", releaseYear: "2010"}
- ▶ 4: {title: "Interstellar", releaseYear: "2014"}

length: 5

▶ __proto__: Array(0)

> |

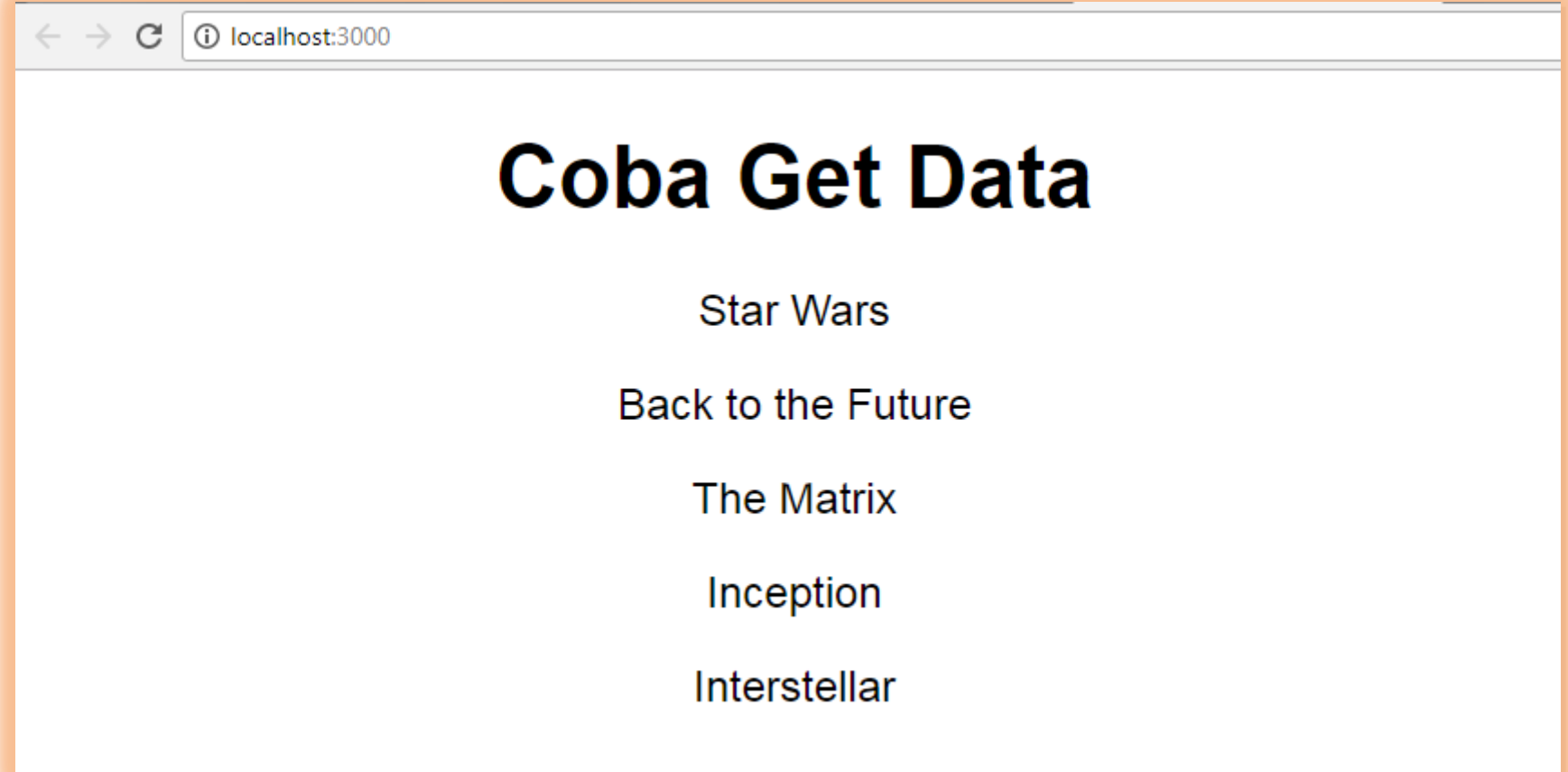
Axios

#3 Axios API

```
axios.get('https://URL.json')
  .then((ambilData) => {
    console.log(ambilData.data.title);
    console.log(ambilData.data.description);
    console.log(ambilData.data.movies);
  })
};
```

===== Can be written as

```
axios({
  url: 'https://URL.json',
  method: 'GET',
}).then((ambilData) => {
  console.log(ambilData.data.title);
  console.log(ambilData.data.description);
  console.log(ambilData.data.movies);
})
```



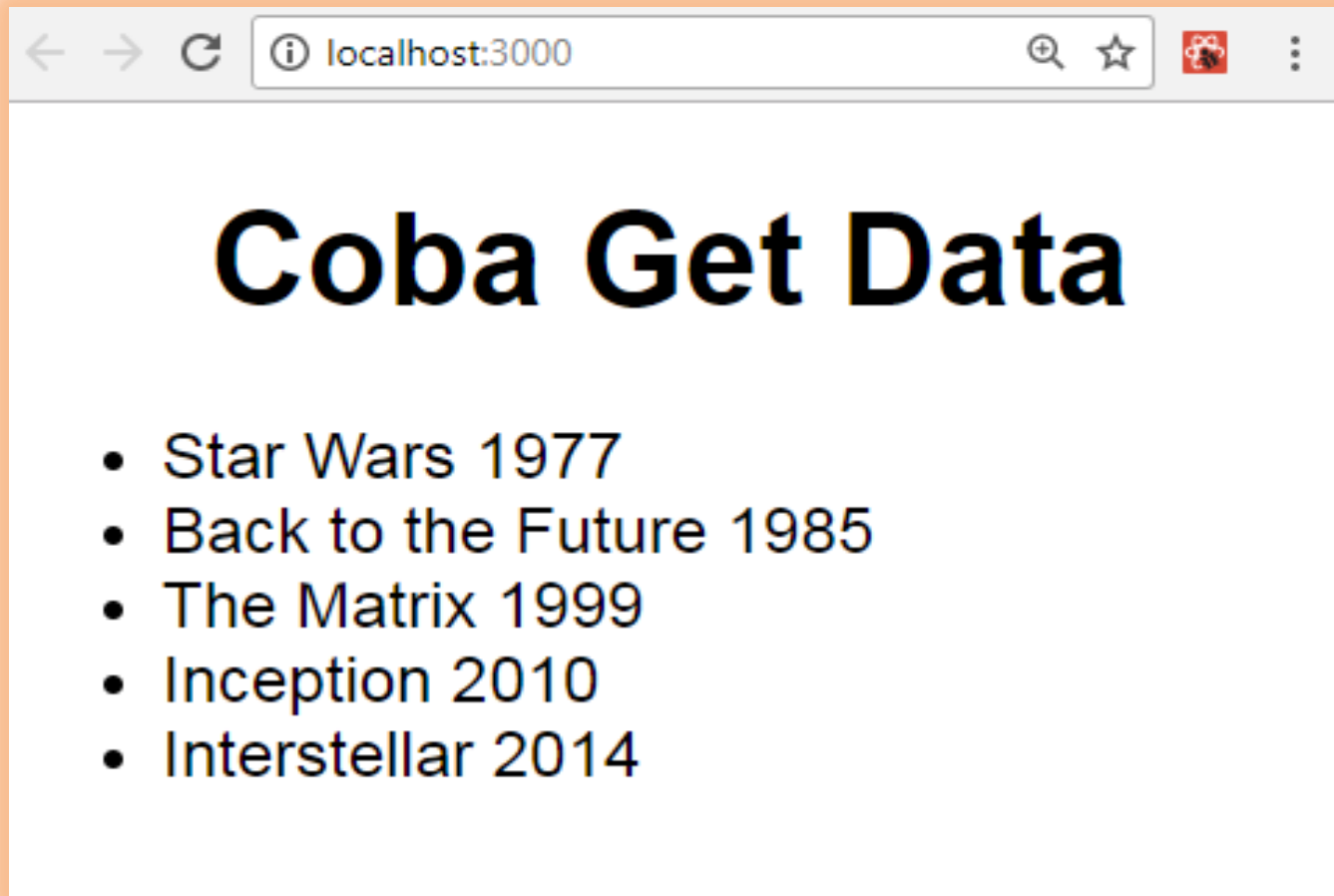
```
import React, { Component } from 'react';
import axios from 'axios';
```

```
class App extends Component {
  constructor() {
    super();
    this.state = {
      judul: '',
    };
  }
  componentDidMount(){
    axios.get('https://facebook.github.io/react-native/movies.json')
      .then((ambilData) => {
        console.log(ambilData);
        this.setState({
          judul0: ambilData.data.movies[0].title,
          judul1: ambilData.data.movies[1].title,
          judul2: ambilData.data.movies[2].title,
          judul3: ambilData.data.movies[3].title,
          judul4: ambilData.data.movies[4].title
        });
      })
  };
};
```

Axios

#4 Render Data part 1

```
render() {  
  return (  
    <div>  
      <center>  
        <h1>Coba Get Data</h1>  
        <p>{this.state.judul0}</p>  
        <p>{this.state.judul1}</p>  
        <p>{this.state.judul2}</p>  
        <p>{this.state.judul3}</p>  
        <p>{this.state.judul4}</p>  
      </center>  
    </div>  
  );  
}  
}  
  
export default App;
```



```
import React, { Component } from 'react';
import axios from 'axios';
class App extends Component {
  constructor() {
    super();
    this.state = {
      judul: [],
    };
  }
  componentDidMount(){
    axios.get('https://facebook.github.io/react-native/movies.json')
      .then((ambilData) => {
        console.log(ambilData);
        this.setState({
          judul: ambilData.data.movies,
        })
      })
  };
};
```


Axios

#5 Render Data part 1

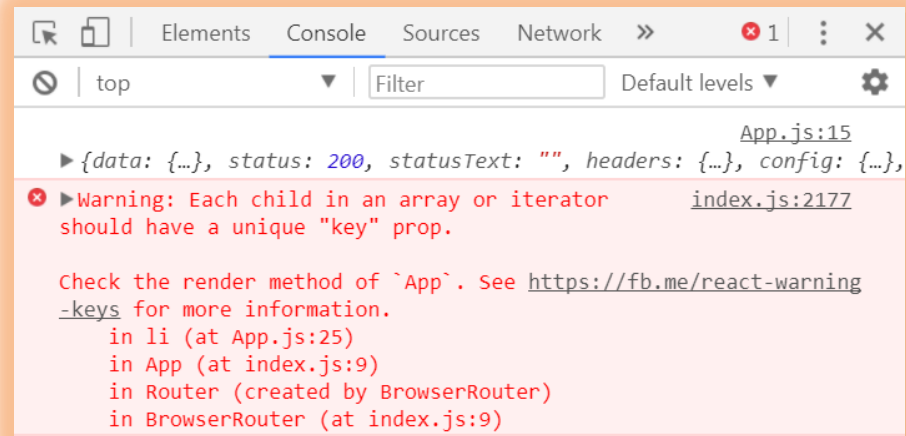
Axios

#5 Render Data part 2

```
render() {  
  const data = this.state.judul.map((item, index)=>{  
    var fullfilm = [item.title,item.releaseYear].join(" ");  
    return <li key={index}>{fullfilm}</li>;  
  })  
  return (  
    <div>  
      <center>  
        <h1>Coba Get Data</h1>  
      </center>  
      <ul>  
        { data }  
      </ul>  
    </div>  
  );  
}  
}  
export default App;
```



Tanpa Keys, output program tetap berhasil, namun di Console muncul Warning



<https://jsonplaceholder.typicode.com/>



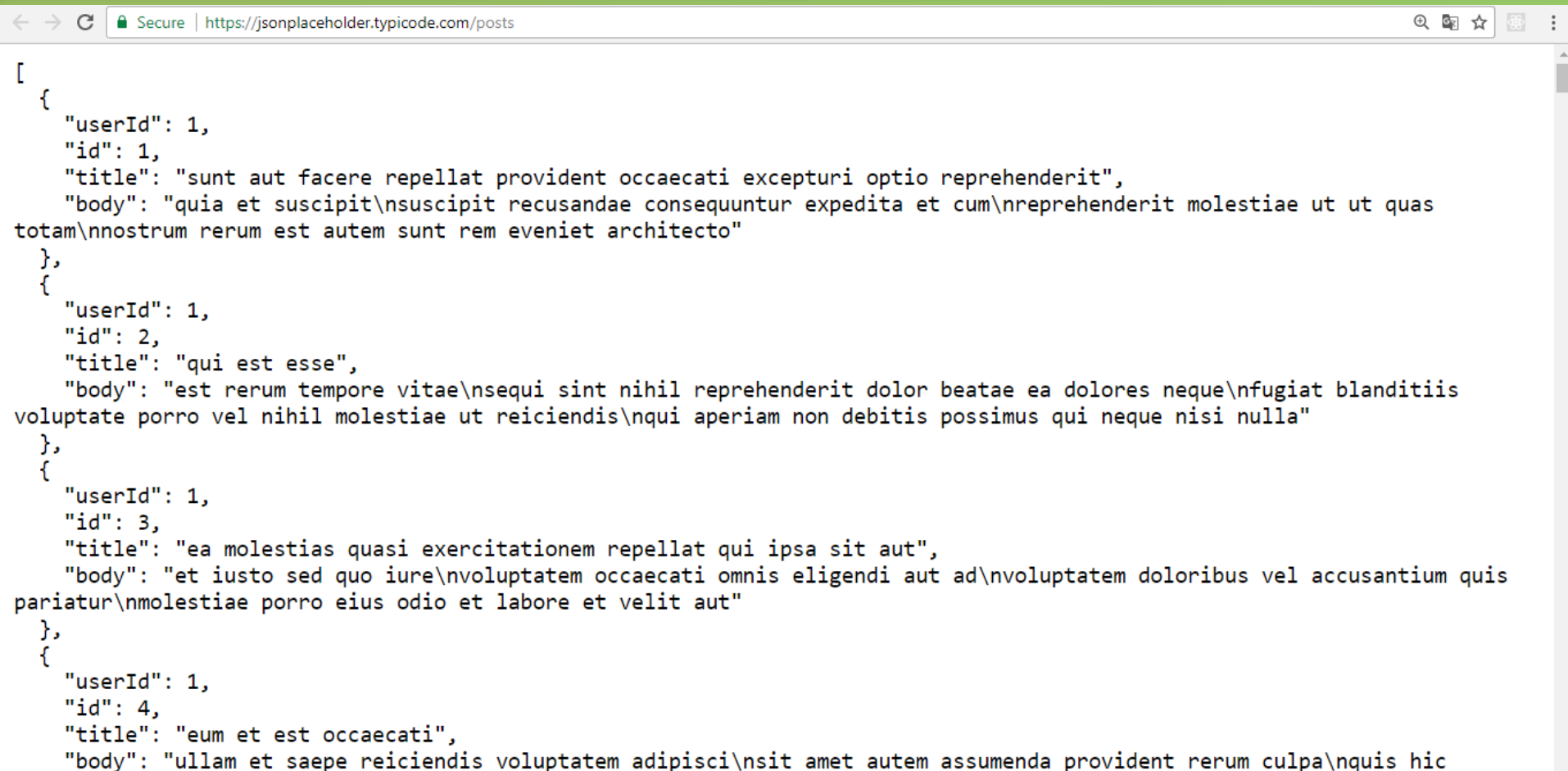
JSONPlaceholder

Fake Online REST API for Testing and Prototyping

powered by [JSON Server](#) and [lowdb](#)

JSONPlaceholder is a free online REST service that you can use whenever you need some fake data.

GET all data from <https://jsonplaceholder.typicode.com/posts>



```
[
  {
    "userId": 1,
    "id": 1,
    "title": "sunt aut facere repellat provident occaecati excepturi optio reprehenderit",
    "body": "quia et suscipit\nsuscipit recusandae consequuntur expedita et cum\nreprehenderit molestiae ut ut quas totam\nnostrum rerum est autem sunt rem eveniet architecto"
  },
  {
    "userId": 1,
    "id": 2,
    "title": "qui est esse",
    "body": "est rerum tempore vitae\nsequi sint nihil reprehenderit dolor beatae ea dolores neque\nfugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis\nqui aperiam non debitis possimus qui neque nisi nulla"
  },
  {
    "userId": 1,
    "id": 3,
    "title": "ea molestias quasi exercitationem repellat qui ipsa sit aut",
    "body": "et iusto sed quo iure\nvoluptatem occaecati omnis eligendi aut ad\nvoluptatem doloribus vel accusantium quis pariatur\nmolestiae porro eius odio et labore et velit aut"
  },
  {
    "userId": 1,
    "id": 4,
    "title": "eum et est occaecati",
    "body": "ullam et saepe reiciendis voluptatem adipisci\nsit amet autem assumenda provident rerum culpa\nquis hic"
  }
]
```

Try on Postman!

Get all 100 data (id & title) from
<https://jsonplaceholder.typicode.com/posts>

Coba Get Data

- 1 - sunt aut facere repellat provident occaecati excepturi optio reprehenderit
- 2 - qui est esse
- 3 - ea molestias quasi exercitationem repellat qui ipsa sit aut
- 4 - eum et est occaecati
- 5 - nesciunt quas odio
- 6 - dolore eum magni eos aperiam quia
- 7 - magnam facilis autem
- 8 - dolore dolore est ipsam
- 9 - nesciunt iure omnis dolore tempora et accusantium
- 10 - optio molestias id quia eum
- 11 - et ea vero quia laudantium autem
- 12 - in quibusdam tempore odit est dolore
- 13 - dolorum ut in voluptas mollitia et saepe quo animi
- 14 - voluptatem eligendi optio
- 15 - eveniet quod temporibus
- 16 - sint suscipit perspiciatis velit dolorum rerum ipsa laboriosam odio
- 17 - fugit voluptas sed molestias voluptatem provident
- 18 - voluptate et itaque vero tempora molestias

```
import React, { Component } from 'react';
import axios from 'axios';
```

```
class App extends Component {
  constructor() {
    super();
    this.state = {
      dataKu: [],
    };
  }
```

```
  componentDidMount(){
    axios.get('https://jsonplaceholder.typicode.com/posts')
      .then((ambilData) => {
        console.log(ambilData);
        this.setState({
          dataKu: ambilData.data,
        })
      })
  }
};
```

```
render() {  
  const data = this.state.dataKu.map((item, index)=>{  
    var id_title = [item.id,item.title].join(" - ");  
    return <li key={index}>{id_title}</li>;  
  })  
  return (  
    <div>  
      <h1>Coba Get Data</h1>  
      { data }  
    </div>  
  );  
}  
}  
  
export default App;
```

GET all data from <https://jsonplaceholder.typicode.com/users>



```
[
  {
    "id": 1,
    "name": "Leanne Graham",
    "username": "Bret",
    "email": "Sincere@april.biz",
    "address": {
      "street": "Kulas Light",
      "suite": "Apt. 556",
      "city": "Gwenborough",
      "zipcode": "92998-3874",
      "geo": {
        "lat": "-37.3159",
        "lng": "81.1496"
      }
    },
    "phone": "1-770-736-8031 x56442",
    "website": "hildegard.org",
    "company": {
      "name": "Romaguera-Crona",
      "catchPhrase": "Multi-layered client-server neural-net",
      "bs": "harness real-time e-markets"
    }
  },
]
```

Try on Postman!

Get all 10 data (id, name, email & address) from <https://jsonplaceholder.typicode.com/users>

Coba Get Data

No	Nama	Email	Alamat
1	Leanne Graham	Sincere@april.biz	Apt. 556, Kulas Light, Gwenborough
2	Ervin Howell	Shanna@melissa.tv	Suite 879, Victor Plains, Wisokyburgh
3	Clementine Bauch	Nathan@yesenia.net	Suite 847, Douglas Extension, McKenziehaven
4	Patricia Lebsack	Julianne.OConner@kory.org	Apt. 692, Hoeger Mall, South Elvis
5	Chelsey Dietrich	Lucio_Hettinger@annie.ca	Suite 351, Skiles Walks, Roscoeview
6	Mrs. Dennis Schulist	Karley_Dach@jasper.info	Apt. 950, Norberto Crossing, South Christy
7	Kurtis Weissnat	Telly.Hoeger@billy.biz	Suite 280, Rex Trail, Howemouth
8	Nicholas Runolfsdottir V	Sherwood@rosamond.me	Suite 729, Ellsworth Summit, Aliyaview
9	Glenna Reichert	Chaim_McDermott@dana.io	Suite 449, Dayna Park, Bartholomebury
10	Clementina DuBuque	Rey.Padberg@karina.biz	Suite 198, Kattie Turnpike, Lebsackbury

```
import React, { Component } from 'react';
import axios from 'axios';
```

```
class App extends Component {
  constructor() {
    super();
    this.state = {
      dataKu: [],
    };
  }
```

```
  componentDidMount(){
    axios.get('https://jsonplaceholder.typicode.com/users')
      .then((ambilData) => {
        console.log(ambilData);
        this.setState({
          dataKu: ambilData.data,
        })
      })
  };
};
```

```
render() {  
  var css = {border:'1px solid black', padding:'12px'}  
  const data = this.state.dataKu.map((item, index)=>{  
    var id = item.id;  
    var name = item.name;  
    var mail = item.email;  
    var alamat = [  
      item.address.suite,  
      item.address.street,  
      item.address.city].join(", ");  
    return <tr style={css} key={index}>  
      <td style={css}>{id}</td>  
      <td style={css}>{name}</td>  
      <td style={css}>{mail}</td>  
      <td style={css}>{alamat}</td>  
    </tr>;  
  })  
}
```



```
return (  
<div><center>  
<h1>Coba Get Data</h1>  
<table style={css}>  
<tbody>  
<tr>  
<th>No</th>  
<th>Nama</th>  
<th>Email</th>  
<th>Alamat</th>  
</tr>  
{ data }  
</tbody>  
</table>  
</center></div>  
>;  
>>  
export default App;
```

Create Fake REST API with JSON-Server #1

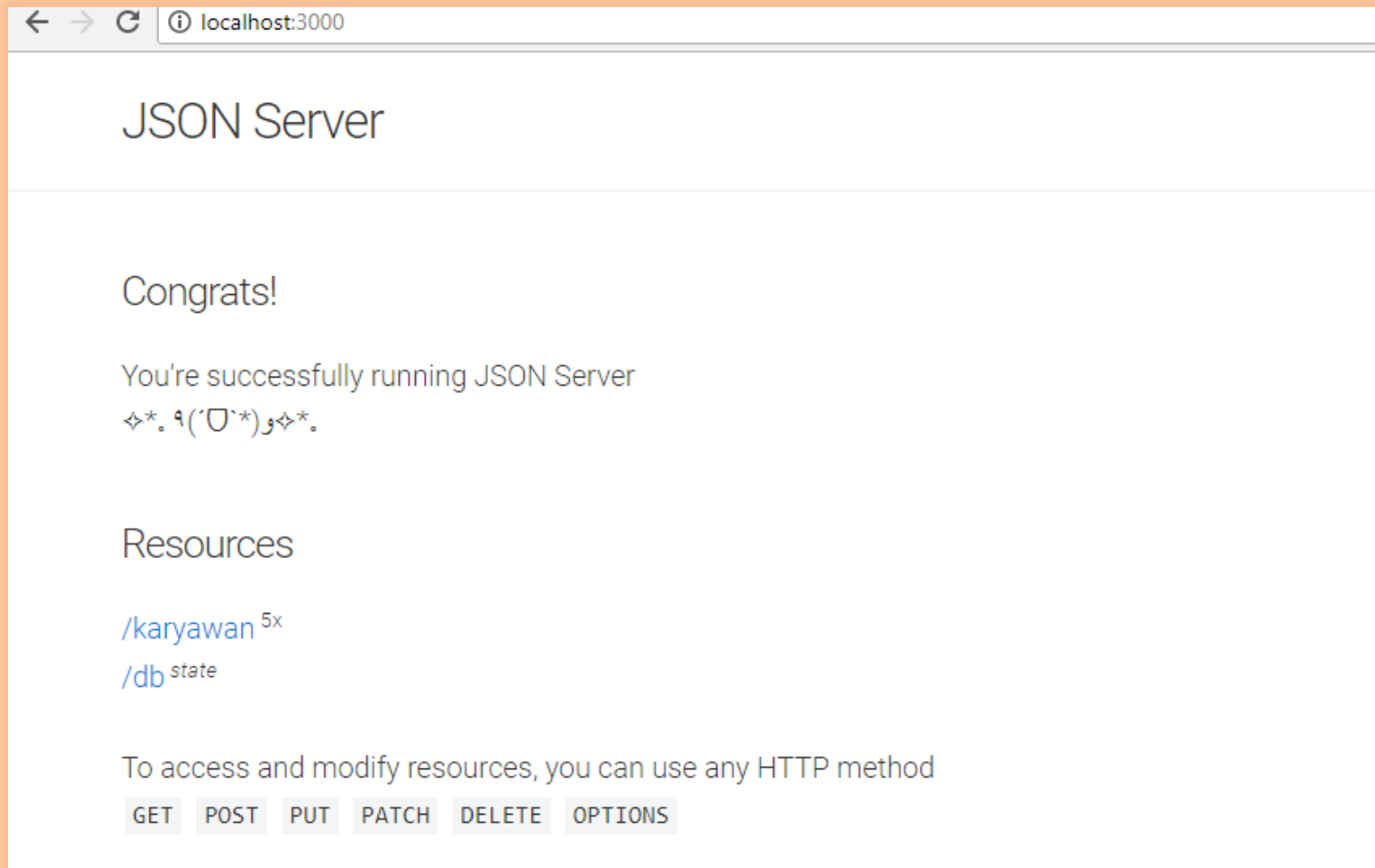
- ❖ Create a project folder, for eg *lin_fake_api*
- ❖ Create a json file on project folder, for eg *db.json*

```
{
  "karyawan": [
    {"id":1, "nama": "Andi", "usia":25, "kota": "Jakarta"},
    {"id":2, "nama": "Budi", "usia":27, "kota": "Yogya"},
    {"id":3, "nama": "Caca", "usia":25, "kota": "Bandung"},
    {"id":4, "nama": "Dedi", "usia":31, "kota": "Jakarta"},
    {"id":5, "nama": "Euis", "usia":22, "kota": "Bandung"}
  ]
}
```

Create Fake REST API with JSON-Server #2

- ❖ On project folder, type:
`$ npm init`
- ❖ Install json-server:
`$ npm install -g json-server --save`
- ❖ Activate json-server:
`$ json-server --watch db.json`
- ❖ Done, see on browser
`http://localhost:3000`

http://localhost:3000



http://localhost:3000/karyawan

```
← → ↺ ⓘ localhost:3000/karyawan

[
  {
    "id": 1,
    "nama": "Andi",
    "usia": 25,
    "kota": "Jakarta"
  },
  {
    "id": 2,
    "nama": "Budi",
    "usia": 27,
    "kota": "Yogyakarta"
  },
  {
    "id": 3,
    "nama": "Caca",
    "usia": 25,
    "kota": "Bandung"
  },
  {
    "id": 4,
    "nama": "Dedi",
    "usia": 31,
    "kota": "Jakarta"
  },
  {
    "id": 5,
    "nama": "Euis",
    "usia": 22,
    "kota": "Bandung"
  }
]
```

GET on Postman

localhost:3000/db

localhost:3000/karyawan

localhost:3000/karyawan/3

localhost:3000/karyawan?id=3

localhost:3000/karyawan?usia=25

localhost:3000/karyawan?_limit=3

localhost:3000/karyawan?_sort=nama&_order=desc

localhost:3000/karyawan?usia_gte=26

localhost:3000/karyawan?usia_gte=23&usia_lte=27

localhost:3000/karyawan?q=Jakarta

POST on Postman

- Set **POST** to **http://localhost:3000/karyawan**
- Set **Headers** Key: **Content-Type** & value: **app/json**
- Insert data json on **Body raw** & send it, for eg:

```
{  
  "id":6,  
  "nama" : "Fifi",  
  "usia" : 28,  
  "kota" : "Medan"  
}
```

- Data on db.json is updated!

DELETE on Postman

- Set **DELETE** to <http://localhost:3000/karyawan/5> to delete data karyawan number 5.

UPDATE on Postman

- Edit **PATCH** to <http://localhost:3000/karyawan/2> to update data karyawan number 2.
- Set **Headers** Key: **Content-Type** & value: **app/json**. Insert data json on **Body raw** & send it, for eg:

```
{  
  "kota" : "Solo"  
}
```


**Try to connect
React Project to
Fake REST API!**

الأذان

aladhan.com



alquran.cloud/api

Free API

Part 1 CoPas only

QUOTES on DESIGN

quotesondesign.com



BLOCKCHAIN

[blockchain.info/api/
exchange_rates_api](http://blockchain.info/api/exchange_rates_api)

COUNTRY API



[fabian7593.github.io/
CountryAPI/](http://fabian7593.github.io/CountryAPI/)

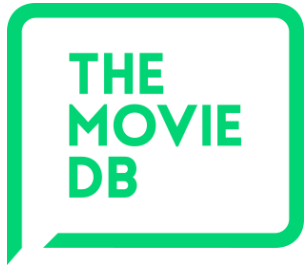
TheSportsDB

thesportsdb.com/api.php

 **Purwadhika**
Startup and Coding School

Free API

Part 2 Auth Key



themoviedb.org



openweathermap.org



api.nasa.gov



wunderground.com



food2fork.com



last.fm



xsight

telkomxsight.com



mainAPI

mainapi.net

Free API
Indonesian!



JAKARTA
SMART CITY

api.jakarta.go.id

 **Purwadhika**
Startup and Coding School

Free API

Part 3 Header Key

The Zomato logo consists of a solid red rounded square with the word "zomato" written in white lowercase letters in the center.

zomato

zomato.com