

Tutorial RESTful API Node JS + Express + MySQL — Part 1

Belajar RESTful API Node JS + Express + MySQL (Read)



Kiddy

Follow

Jun 19, 2018 · 5 min read



Hello coders! Kali ini gue mau ngajarin kalian belajar mengenai cara membuat RESTful API di Node JS dengan menggunakan salah satu framework Node JS yaitu **Express** dan database-nya yang simple aja dulu, **MySQL**

Mungkin beberapa pembaca berfikir bahwa “Kenapa gak Mongo DB aja sih?”, ya I know Mongo DB bagus dan gue juga udah nyoba. Kita belajar yang basic aja dulu, tutorial yang pake Mongo DB juga banyak yang bagus-bagus, tapi giliran tutorial yang pake MySQL sedikit banget yang tulisannya jelas, & mudah dimengerti. Apalagi mereka pake English ahahaha.

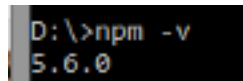
Di tutorial ini, gue anggep lo adalah orang yang udah paham Node JS itu apa, Express itu apa, dan RESTful API itu apa. Jadi kalo lo belum paham lebih baik lo belajar dulu ditempat lain karena gue engga akan jelasin hal tersebut lagi.

Oke langsung aja yuk cus!

Pertama, pastiin lo punya Node JS nya dulu lah. Kalo engga ada download disini. Kalo udah langsung install ya.

Kedua, setelah install lo tulis di command prompt lu “npm -v” untuk cek versinya.

```
npm -v
```



```
D:\>npm -v
5.6.0
```

Sip kalo gini berarti udah ready.

Ketiga, masuk ke folder manapun tempat lo ngebuat projek. Buat dulu foldernya abis itu masuk ke foldernya di CMD dan ketik “npm init”

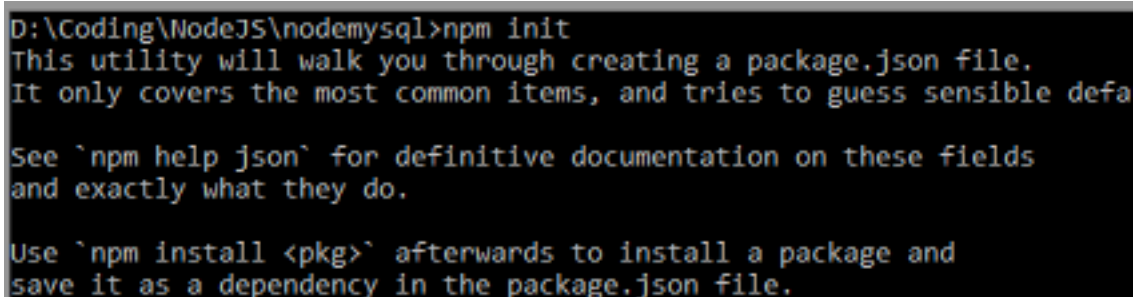
```
npm init
```

Tujuannya adalah dia akan generate file package.json, sebelum lu nanya gue jelasin nih.

*npm uses the **package.json** file to specify the version of a **package** that your app depends on.*

artinya dilarang ngerokok.

NPM menggunakan package.json untuk menspesifikasikan versi dari paket (library) yang dibutuhkan di aplikasi buatan lu.



```
D:\Coding\NodeJS\nodemysql>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help json` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.
```

```
Press ^C at any time to quit.
package name: (nodemysql)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to D:\Coding\NodeJS\nodemysql\package.json:

{
  "name": "nodemysql",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}

Is this ok? (yes)
```

nah diatas adalah hasil dari package.json yang akan dimasukkan, lo bisa ganti versi, nama, deskripsi, file utama yang dijalankan, script test, author, bahkan licensenya. Tapi karena ini buat belajar ya enter enter enter aja biar cepet hehehe.

oke sekarang install package yang kita perlukan, kita akan pake

1. express package
2. mysql package
3. body-parser package

Package body-parser ini dipake untuk ngeparsing setiap request yang masuk melalui HTTP, baik itu dengan x-www-form-urlencoded, raw json, dan form data. Package ini ngebuat kita gampang untuk make method parsing dari apa saja yang disubmit melalui rest api kita.

nah cara installnya gini

```
npm install --save express mysql body-parser
```

cakep, kalo udah saatnya menunggu. Ini agak lama jadi mendingan lu tinggal ngopi aja atau pup (poop), tergantung kecepatan internet lu.

Setelah kedownload semuanya, sekarang saatnya kita ngoding.

Buat dulu tablenya yang akan kita pake, tablenya kaya gini aja biar simple.

```
CREATE TABLE IF NOT EXISTS `person` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `first_name` varchar(50) NOT NULL DEFAULT '0',  
  `last_name` varchar(50) NOT NULL DEFAULT '0',  
  PRIMARY KEY (`id`)  
)
```

Lalu, Buat file **server.js**, ini adalah file utama kita dimana bisa dibilang kaya index.php gitu deh.

```
var express = require('express'),  
    app = express(),  
    port = process.env.PORT || 3000,  
    bodyParser = require('body-parser'),  
    controller = require('./controller');  
  
app.use(bodyParser.urlencoded({ extended: true }));  
app.use(bodyParser.json());  
  
var routes = require('./routes');  
routes(app);  
  
app.listen(port);  
console.log('Learn Node JS With Kiddy, RESTful API server started  
on: ' + port);
```

Disini kita juga ga akan masukin routes, controller, serta response value langsung kedalam server.js (ga dimasukin satu file), biar apa?

biar rapih aja, jadi sembari membiasakan lo ngoding terstruktur dan dengan file yang dipisah-pisah sesuai kebutuhannya ^.^

Oke setelah server.js dijalankan, kita akan ngebuat file koneksi antara si database dan aplikasi kita.

Buat sebuah file dan beri nama **conn.js** sebagai tempat mengkoneksikan aplikasi ke database ketika kita butuh koneksi ke database.

```
var mysql = require('mysql');

var con = mysql.createConnection({
  host: "localhost",
  user: "root",
  password: "",
  database: "namadatabase"
});

con.connect(function (err) {
  if(err) throw err;
});

module.exports = con;
```

Oke kalo udah sekarang saatnya buat file controller tempat kita ngoding. Sekarang buat file **controller.js** dan copy code dibawah~

```
'use strict';

var response = require('./res');
var connection = require('./conn');

exports.users = function(req, res) {
  connection.query('SELECT * FROM person', function (error, rows, fields) {
    if(error) {
      console.log(error)
    } else {
      response.ok(rows, res)
    }
  });
};

exports.index = function(req, res) {
  response.ok("Hello from the Node JS RESTful side!", res)
};
```

Kalo udah sekarang langsung buat **res.js**

loh fungsinya apa? pasti ada yang bertanya-tanya gitu kenapa gue harus buat **res.js**? Oke sedikit penjelasan ya.

Kalo kalian pernah baca tutorial yang gue tulis baik itu di Golang, Lumen. Pasti ada standarisasi respons yang akan dikembalikan. Contohnya gini

```
{
  "status": 1,
  "message": "Success",
  "Data": [
    {
      "id": "1",
      "firstname": "adsasdas",
      "lastname": "asdadas"
    }
  ]
}
```

Nah standarisasi respon ini butuh banget cuy, buat si frontend developer. Fungsinya biar dia udah tau parameter apa yang dikembaliin dan valuenya seperti apa aja. Karena team yang baik adalah team yang saling memahami satu sama lain, kaya couple idaman gitu (yang jomblo kaya gue jgn baper) ahahaha.

Oke setelah intermezzo, lanjut ya buat file **res.js** dan copy paste kode dibawah.

```
'use strict';

exports.ok = function(values, res) {
  var data = {
    'status': 200,
    'values': values
  };
  res.json(data);
  res.end();
};
```

oke disini gue gue tulis **res.json(data)** nah data inilah yang akan dikembalikan jadi json dan fungsi **res.end()** adalah fungsi yang dipake untuk nutup koneksi ke database setelah data ditampilkan ke json.

Sekarang saatnya buat file **routes.js**

Ya file ini dipake buat nulis routes atau endpoint apa saja yang ada di rest api kita.

```
'use strict';

module.exports = function(app) {
  var todoList = require('./controller');

  app.route('/')
    .get(todoList.index);

  app.route('/users')
    .get(todoList.users);
};
```

nah mantep betul, kita udah berhasil ngebuat REST API yang baru Read aja. CUD nya menyusul di part 2.

Sekarang saatnya mencoba! Yak coba jalanin dengan tulis di CMD kita **npm start** kalo misalnya error tulis **node server.js** di command prompt.

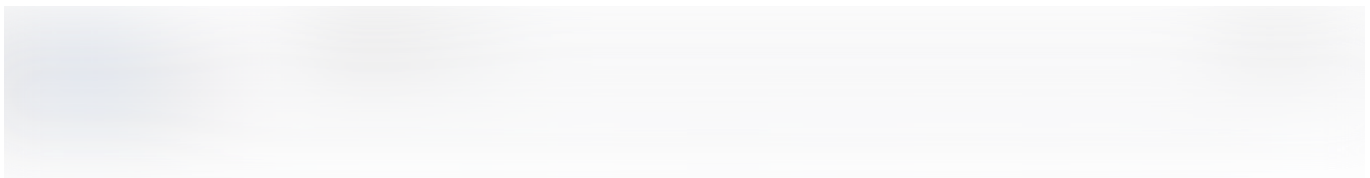
```
node server.js
```



Nah kalo udah gitu berarti udah jalan, akses ke **localhost:3000** dan lihat hasilnya.



Nah kalo gini udah berhasil berarti. Sekarang akses ke **localhost:3000/users**



Mantul, mantap betul! Sekarang lo udah siap untuk ngebuat REST API pake Node JS dengan framework Express dan database MySQL ^.^

Sekian dari gue, tunggu part-2 nya ya ^^

Semoga bermanfaat ya, dan happy coding!

[Nodejs](#) [Expressjs](#) [MySQL](#) [Restful Api](#) [Rest Api](#)

[About](#) [Help](#) [Legal](#)