WEB PROGRAMMING

MODUL PRAKTIKUM Express

DAFTAR ISI

[1 Pendahuluan 2](#_Toc55118471)

[1.1 Pengenalan Express 2](#_Toc55118472)

[1.2 Instalasi 2](#_Toc55118473)

[1.3 Create Basic Express Server 3](#_Toc55118474)

[2 Create Basic HTTP Express Server 5](#_Toc55118475)

[2.1 Create Main File 5](#_Toc55118476)

[2.2 Instalasi nodemon 6](#_Toc55118477)

[2.2.1 Pengenalan nodemon 6](#_Toc55118478)

[2.2.2 Instalasi nodemon 7](#_Toc55118479)

[2.2.3 Menjalankan nodemon 7](#_Toc55118480)

[3 HTTP Request Methods 9](#_Toc55118481)

[3.1 Pengenalan 9](#_Toc55118482)

[3.2 HTTP Response Status Codes 9](#_Toc55118483)

[4 Static Content 11](#_Toc55118484)

[4.1 Module 11](#_Toc55118485)

[4.2 Middlewares 11](#_Toc55118486)

[4.3 Serving Static Files 11](#_Toc55118487)

[5 Routing 13](#_Toc55118488)

[5.1 Route definition structure 13](#_Toc55118489)

[5.2 Defining basic routes 14](#_Toc55118490)

[5.3 Testing Basic Routes 15](#_Toc55118491)

# Pendahuluan

## Pengenalan Express

Express adalah framework aplikasi web Node.js minimal dan fleksibel yang menyediakan serangkaian fitur canggih untuk aplikasi web dan seluler.

Express memiliki segudang metode utilitas HTTP dan middleware yang Anda inginkan, membuat API yang kuat cepat dan mudah.

Beberapa keunggulan yang dimiliki oleh Express.js antara lain:

1. Dukungan pembuatan *middleware*
2. Dukungan terhadap berbagai HTTP *verb* seperti POST, GET, PUT, DELETE, OPTION, HEAD, dan lainnya
3. Sudah terpasang *template engine* Jade
4. manajemen *file* statik seperti CSS dan Javascript
5. Sangat bebas untuk dikostumisasi

## Instalasi

Untuk menggunakan Express.js, pastikan Anda memasang Node.js terlebih dahulu dan pastikan NPM sudah dapat digunakan.

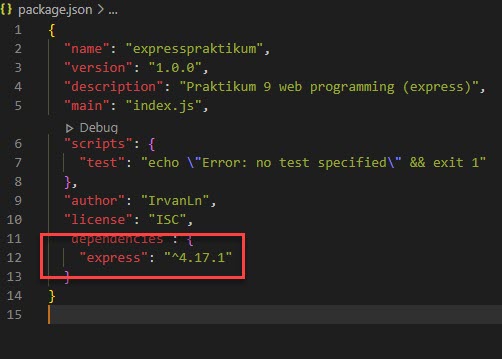
Melalui konsol atau terminal silahkan eksekusi perintah berikut. Pastikan pilih folder yang Kita Tentukan:



Jika selesai maka konsol akan menampilkan informasi sebagai berikut :



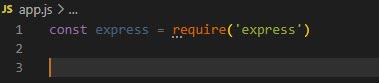
Dan Kita dapat mengecek apakah express sudah berhasil di instal pada file package.json seperti gambar dibawah ini :



## Create Basic Express Server

Pada praktikum sebelumnya Kita sudah membuat sebuah HTTP server menggunakan NodeJS maka sekarang ini Kita akan membuat versi express sebagai berikut :

1. Buat sebuah file dengan nama “app.js”
2. Jika kita akan menggunakan express pada sebuah file maka kita harus mengimport express terlebih dahulu pada sebuah file tersebut sebagai berikut :



1. Kemudian tambahkan sebuah fungsi untuk memberitahukan port yang akan dipakai dan pemanggilan fungsinya :



1. Pada konsol panggil nama filenya menggunakan node dengan perintah **“node app.js”**

****

1. Maka pada layar konsol akan muncul sesuatu yang sudah Kita tuliskan pada console.log sebelumnya.

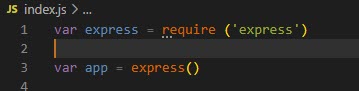
Modul express akan menangani tiap request dari user dan kemudian akan memberikan response berupa file yang diinginkan.

# Create Basic HTTP Express Server

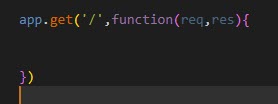
Pada bab ini Kita akan membuat sebuah web server dengan menggunakan NodeJs seperti pada praktikum sebelumnya. dan kemudian Kita akan menambahkan penggunaan express.

## Create Main File

1. Pada Folder yang telah terinisialisasi Node install express seperti yang telah dilakukan di bab sebelumnya.
2. Pertama Buat sebuah file baru dengan nama “index.js”
3. Kemudian import express pada folder tersebut.
4. Buat sebuah instances dengan nama express()



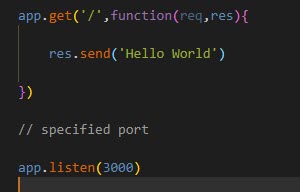
1. buat callback untuk memanggil HTTP get request dengan path utama root (‘/’)



Pada asynchronous Callback diatas terdapat fungsi dengan dua buah parameter (req dan res) yang digunakan untuk me-request data dan menampung hasil (result / response) pada fungsi yang akan dibuat di tahap selanjutnya.

app.get() merupakan sebuah method untuk mendefinisikan route dan menetapkan fungsi callback yang akan dipanggil setiap kali ada permintaan (request) get HTTP dengan path (‘/’)

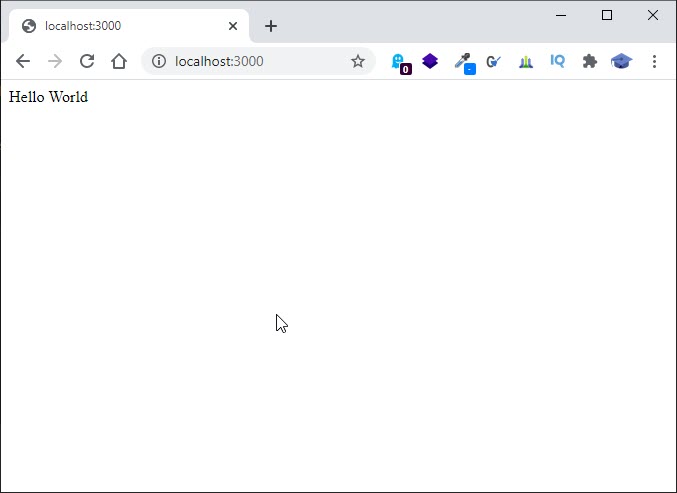
1. Kemudian gunakan method send() untuk menuliskan sebuah data yang akan ditampilkan pada browser. Contoh Kita akan menuliskan kata “Hello World” pada browser dengan listen port 3000



1. Kemudian jalankan node pada terminal yang sudah dibuka dengan perintah “node index.js”



1. Pada browser ketikan alamat localhost beserta port yang sedang dijalankan “localhost:3000”



Maka akan tampil tulisan yang Kita tulis pada method res.send( ).

## Instalasi nodemon

### Pengenalan nodemon

Nodemon adalah sebuah utilitas yang akan memantau setiap perubahan dalam source Kita dan secara otomatis merestart server Kita.

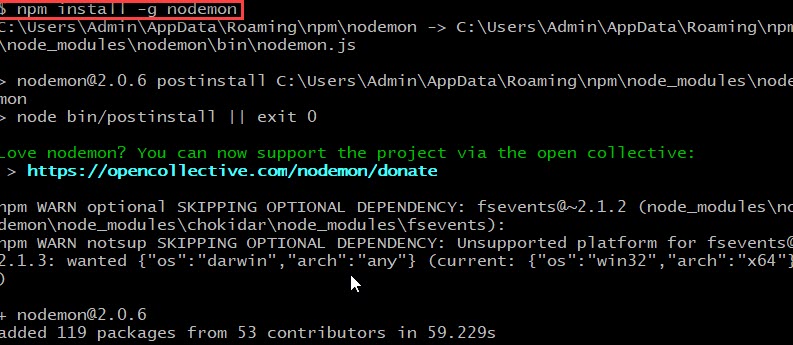
fitur - fitur yang ada pada nodemon :

1. Restart aplikasi secara otomatis
2. Mendeteksi ekstensi file default untuk dipantau
3. Support default untuk node & coffeescript
4. Mengabaikan (ignores) file atau direktori tertentu
5. Monitoring direktori tertentu
6. Bekerja dengan aplikasi server atau utilitas non-time run dan REPL (Read-Eval-Print-Loop ),
7. Membutuhkan node untuk menggunakan nodemon.

### Instalasi nodemon

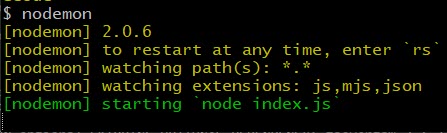
Pada terminal konsol ketik perintah “npm install -g nodemon”

tunggu proses instalasi sampai selesai seperti gambar berikut :



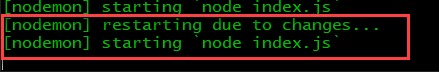
### Menjalankan nodemon

Untuk menjalankan nodemon cukup mengentikan “nodemon” pada folder yang sudah terinstall nodemon.

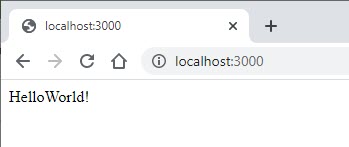


Maka ketika server berjalan maka akan terlihat seperti gambar diatas. Setiap melakukan perubahan maka nodemon akan melakukan restart server dan menyesuaikan isi dari folder source yang sedang di monitor.

Contoh Kita akan melakukan perubahan kata “Hello World” menjadi “Helloworld!”, ketika file disave maka nodemon akan secara otomatis merespon perubahan tersebut tanpa harus menjalankan kembali perintah nodenya.



Pada gambar diatas terlihat 2 baris baru yang muncul ketika Kita melakukan save pada file yang kita ubah.

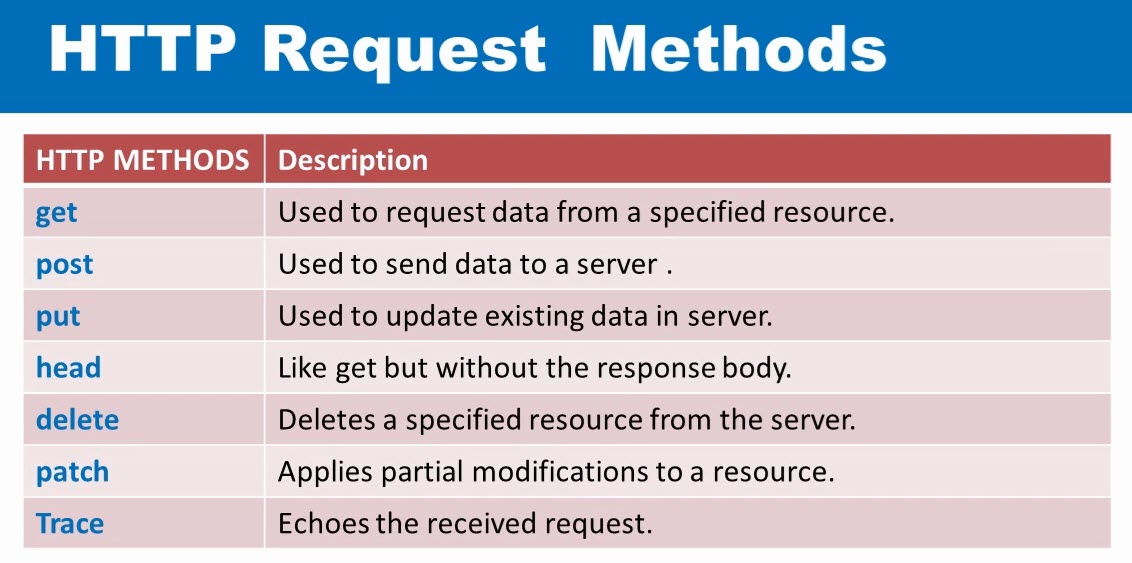


Hasil yang terlihat ketika browser direfresh.

# HTTP Request Methods

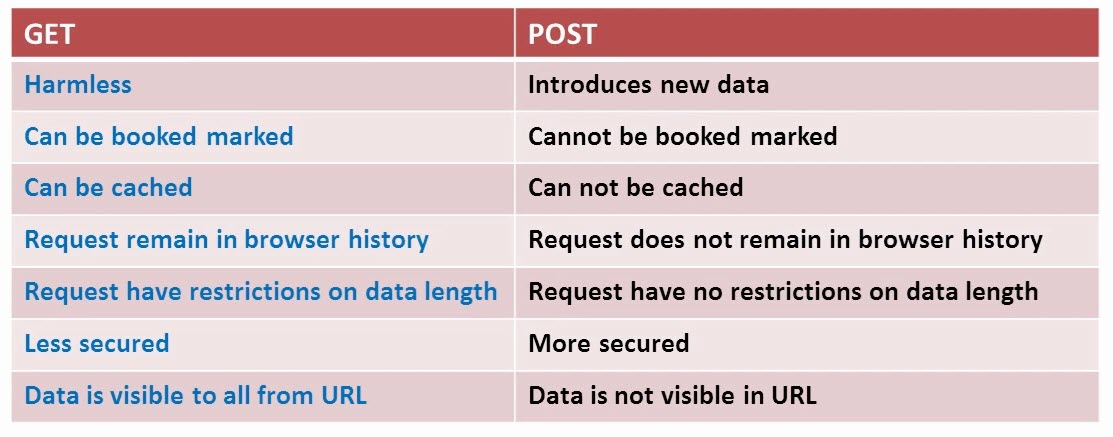
## Pengenalan

HTTP Method merupakan sebuah pekerjaan yang menunjukan tindakan yang diinginkan untuk dilakukan pada sumber daya (resource) yang diidentifikasikan.



Pada gambar diatas terdapat beberapa HTTP Methods yang dapat digunakan.

Request method yang biasa digunakan adalah get dan post. Berikut beberapa perbedaan yang ada pada method get dan post :

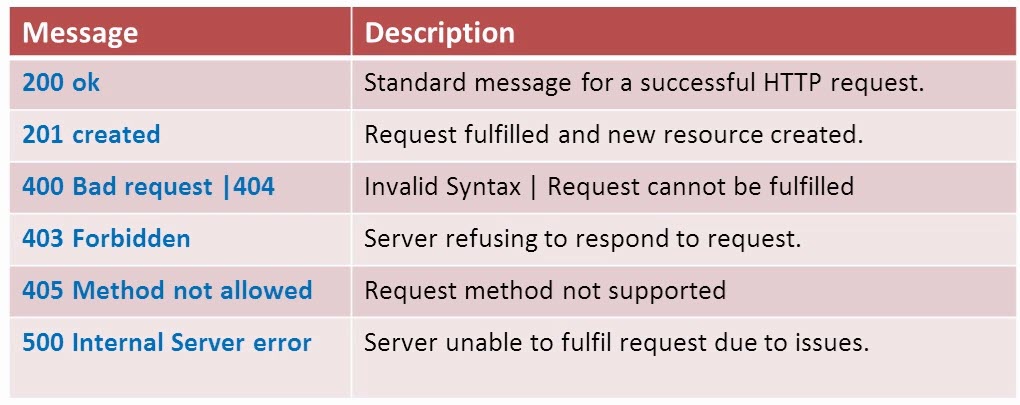


## HTTP Response Status Codes

Merupakan pesan yang dikembalikan setelah permintaan ke server. Terdapat 5 tingkatan status codes seperti gambar dibawah ini :



Pada praktikum ini Kita tidak akan mencoba seluruh status codes yang ada pada http request methods. Hanya beberapa yang paling sering digunakan yang akan Kita bahas diantaranya :

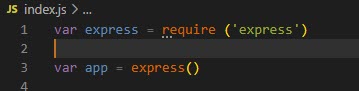


# Static Content

Untuk menyajikan file statis (*Static files*) seperti gambar, file CSS, dan file JavaScript, gunakan fungsi *middleware* bawaan *express.static* di Express. Sebelum masuk pada serving static content maka Kita perlu mengetahui beberapa hal sebagai berikut.

## Module

Modul adalah sebuah *library javascript* atau *file* yang dapat diimport kedalam sebuah code lain menggunakan fungsi ***require().*** Contoh modul adalah express yang sudah Kita gunakan pada bab sebelumnya.



## Middlewares

Middlewares merupakan sebuah fungsi yang melakukan beberapa operasi pada permintaan atau respons dan kemudian memanggil fungsi berikutnya dalam tumpukan yang bisa berupa middlelware lain atau pengendali rute (*route handler*)

Middleware dapat melakukan operasi apa pun, menjalankan kode apa pun, membuat perubahan pada objek permintaan dan respons dan juga dapat mengirim siklus permintaan-respons

Ada beberapa 3rd party middleware yang membantu mempermudah tugas dari web development. Dan dapat diinstall menggunakan node **npm**.

Contoh fungsi middleware yang sudah ada pada express (bawaan / *built in*) adalah **express.static( )** digunakan untuk melayani file statis

## Serving Static Files

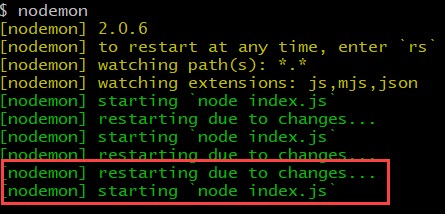
Pada pendahuluan bab diatas sudah disinggung mengenai file static. Express dapat menggunakan middleware **express.static,** namun sebelumnya Kita perlu membuat sebuah folder baru untuk menampung file-file static tersebut.

1. Buat sebuah folder baru dengan nama “public”
2. Didalam folder public buat folder baru dengan nama “images” dan masukan beberapa gambar yang nantinya akan dibuka pada browser menggunakan express.
3. Tambahkan fungsi **app.use** untuk menggunakan middleware express.static seperti berikut :

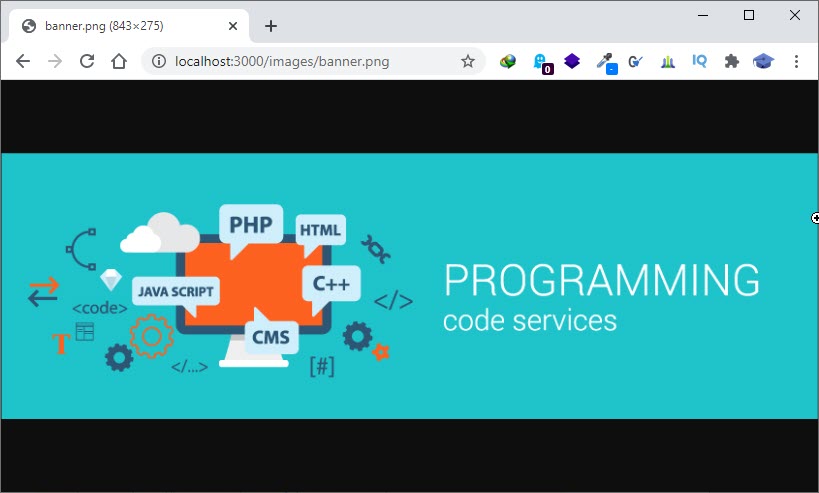


‘public’ adalah nama folder yang digunakan untuk menampung file static.

1. pastikan nodemon masih berjalan maka akan terlihat perubahan setelah file disimpan.



1. Kemudian pada browser coba arahkan URL pada folder public/images/namafilegambar seperti gambar berikut nama file images-nya adalah “banner.png”



# Routing

Pada bab ini kita akan membahas dasar dari routing. routing mengacu pada menentukan bagaimana aplikasi menanggapi permintaan klien ke titik akhir (endpoints) tertentu, yang merupakan URI (path) dan HTTP request methods tertentu (GET, POST, dll).

Setiap route dapat memiliki satu atau lebih *Handler function*, yang dijalankan ketika route cocok dengan handler yang sudah dibuat sebelumnya.

Berikut adalah struktur URL pada umumnya :



Baris pertama <https://www.google.com> dapat dikatakan sebagai *root route*, pada baris kedua yang menjadi root route adalah <https://www.bbc.co.uk> kemudian untuk <https://www.bbc.co.uk/news> dan <https://www.bbc.co.uk/news/uk-43864933> merupakan halaman - halaman yang berbeda dari *root route*.

## Route definition structure

Pada bab ini Kita akan melihat lebih jauh mengenai routing, dasar sturktur route dapat dilihat pada sebagai berikut :

app.METHOD(PATH, HANDLER)

**app** pada fungsi diatas merupakan sebuah instance yang Kita buat sebelumnya yang berisi modul express **require(‘express’).**

**METHOD** pada fungsi diatas merupakan fungsi untuk memanggil HTTP Request Methods biasanya menggunakan huruf kecil seperti **get,post**,**delete**, dll

app.get(PATH,HANDLER), app.post(PATH, HANDLER)

**PATH** pada fungsi diatas mereferensikan jalur yang ada pada server contoh :

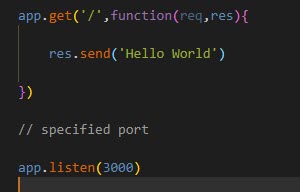
<https://www.google.com>

<https://www.bbc.co.uk/news/uk-43864933>

**HANDLER** merupakan sebuah fungsi yang akan dieksekusi ketika path dan route cocok.

## Defining basic routes

Pada bagian ini Kita akan mempelajari dasar dari pembuatan routes. Pada bagian sebelumnya Kita sudah membuat routing menggunakan method get.



dibagian gambar diatas PATH mengarah pada root route dan fungsi “function(req,res)” merupakan HANDLER yang dipakai.

Kita akan mencoba beberapa method yang dapat dibuat pada express seperti POST, PUT, dan DELETE.

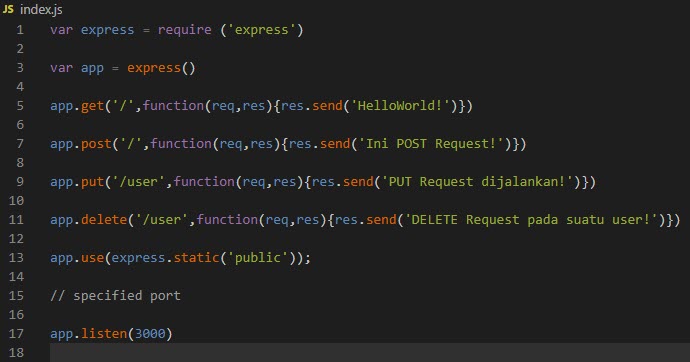
POST biasa digunakan untuk memasukan data baru.

PUT biasa digunakan untuk memperbaharui data yang sudah ada.

DELETE biasa digunakan untuk menghapus data.

Kita akan mencoba HTTP request ini pada route tertentu untuk mengetahui bagaimana routes dan http request methods bekerja.

Kita akan membuat HTTP Request method seperti gambar berikut :



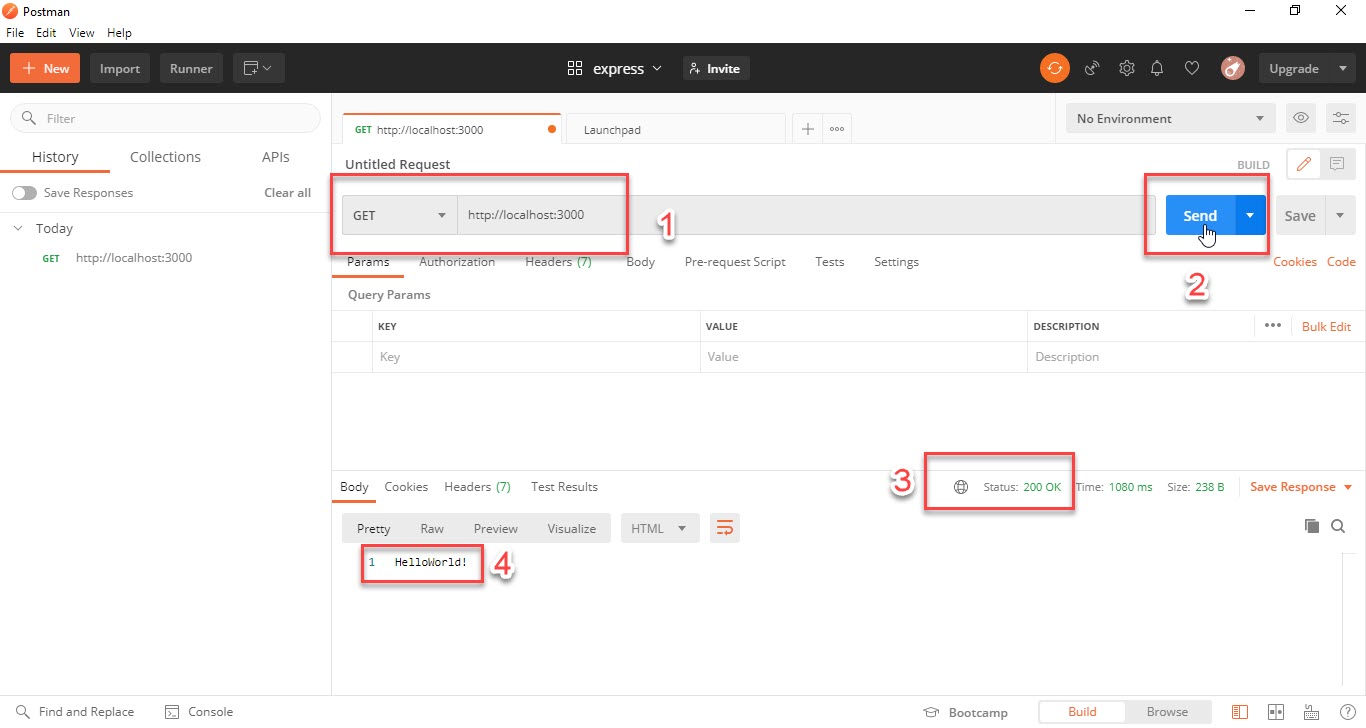
Kita dapat melakukan percobaan / testing untuk method POST, PUT atau DELETE pada aplikasi POSTMAN untuk melihat respon code dan content data yang direquest.

aplikasi postman dapat didownload pada link berikut : <https://www.postman.com/downloads/>

## Testing Basic Routes

Pada sub-bab diatas Kita sudah membahas mengenai postman, bagian ini kita akan mencoba menggunakan POSTMAN untuk mengimplementasikan request method tersebut.

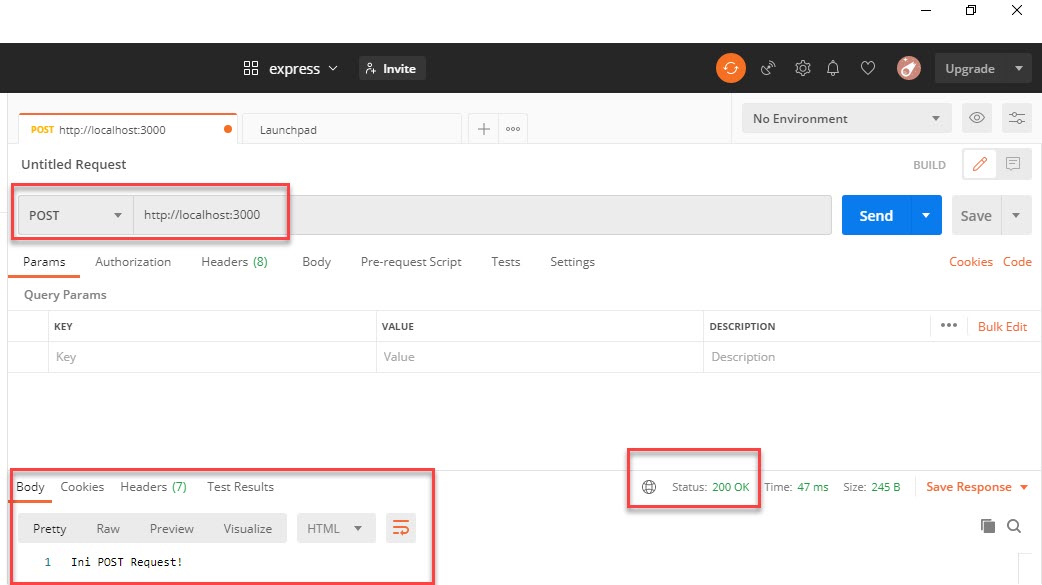
1. Buka aplikasi postman dan pilih method GET kemudian tuliskan URL <http://localhost:3000> pada URL parameter
2. Jangan lupa untuk menjalankan nodemon sebelum mencoba mengirim request get pada postman.

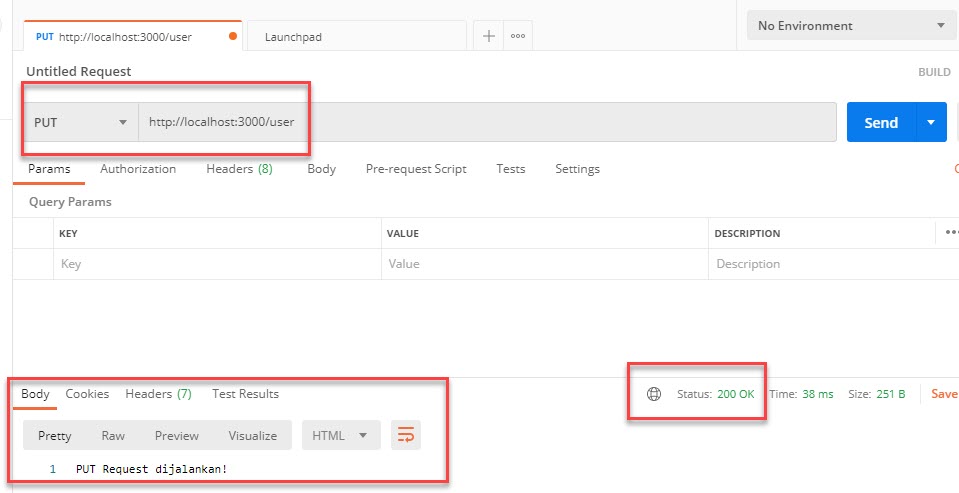


Pada postman kita dapat melihat status code yang merespon ketika Kita mengirimkan permintaan pada server (request server), pada gambar diatas nomor 3 status code nya adalah 200 OK yang artinya server menerima dengan baik dan memberikan response pada permintaan dari browser client.

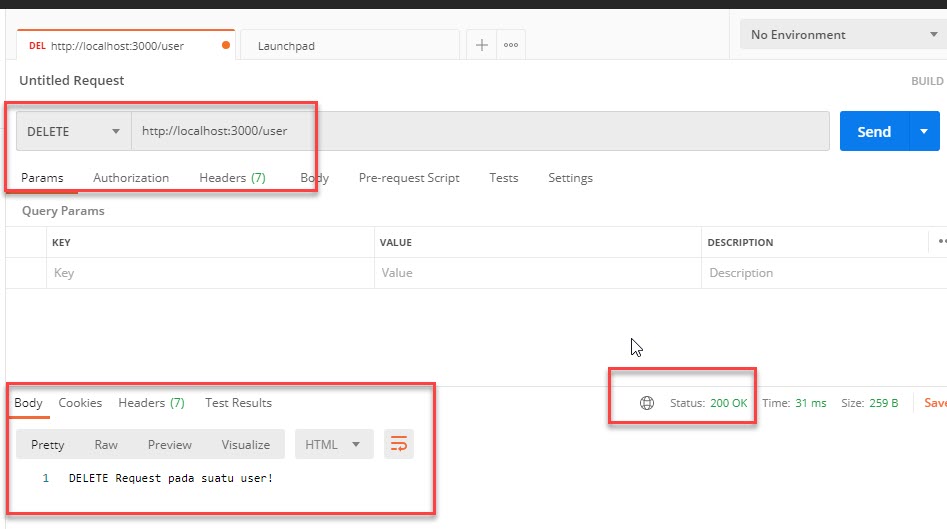
Hasil nya dapat Kita lihat pada nomor 4 di dalam gambar.

1. Kemudian Ganti method GET dengan POST, PUT, DELETE maka seharusnya Kita akan mendapatkan hasil yang berbeda sesuai dengan fungsi yang sudah kita buat di code Kita.





Pada method PUT Kita mengarahkan route pada path “/user” sehingga penulisannya adalah sebagai berikut “http:/localhost:3000/user”. Demikian juga dengan method DELETE.



Seluruh HANDLER pada method dapat diganti menjadi sebuah fungsi yang Kita butuhkan untuk membuat aplikasi berjalan sesuai dengan kebutuhan.