Modul 03

HTTP Request and API

TUJUAN PEMBELAJARAN

1. Mampu memahami dan menjelaskan tentang HTTP Request
2. Mampu memahami dan menjelaskan tentang API

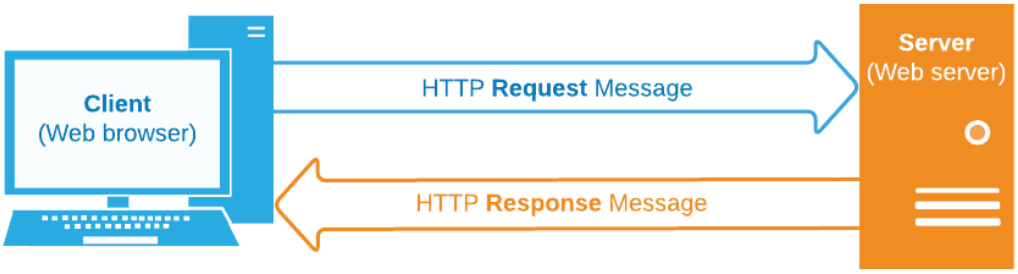
Hardware & Software

1. PC (*Personal Computer)* dengan akses Internet
2. Visual Studio Code
3. Node.js
4. NPM (Node Package Manager)
5. API Access key

URAIAN MATERI

A. HTTP Request

HTTP (Hypertext Transfer Protocol) Request adalah cara utama untuk mengirim permintaan dari klien (biasanya web browser atau aplikasi) ke server web untuk mengakses halaman web atau sumber daya lainnya.



Gambar 1. Ilustrasi HTTP Request

Proses kerja HTTP Request dapat dijelaskan dalam beberapa langkah utama:

1. **Inisiasi Permintaan**:
   - Klien (biasanya web browser) menginisiasi permintaan dengan membuat koneksi ke server web. Ini bisa dilakukan dengan menggunakan protokol TCP/IP (Transmission Control Protocol/Internet Protocol).
   - Klien kemudian menentukan jenis permintaan yang ingin dilakukan, seperti GET (mengambil data), POST (mengirim data), PUT (mengganti data), DELETE (menghapus data), dll.
   - Klien juga menyediakan alamat URL atau URI (Uniform Resource Identifier) yang menunjuk ke sumber daya yang diinginkan di server.
2. **Pembentukan Request**:
   - Klien membuat pesan HTTP Request yang berisi informasi yang diperlukan untuk permintaan tersebut. Pesan ini terdiri dari beberapa komponen utama:
   - Metode (GET, POST, dll.).
   - URI (alamat sumber daya yang diminta).
   - Versi protokol HTTP (misalnya, HTTP/1.1).
   - Header HTTP (informasi tambahan seperti tipe konten yang diinginkan, cookie, dan lain-lain).
   - Tubuh permintaan (data tambahan jika diperlukan, seperti data formulir dalam kasus POST).
3. **Kirim Request**:
   Klien mengirim pesan HTTP Request ke server melalui koneksi yang telah dibuat sebelumnya. Ini melibatkan pengiriman pesan melalui protokol TCP/IP ke alamat IP server yang sesuai dengan URI yang diminta.
4. **Pengolahan Request Server**:
   - Server web menerima pesan HTTP Request dari klien.
   - Server kemudian memproses permintaan sesuai dengan metode yang ditentukan (GET, POST, dll.) dan URI yang disediakan.
   - Server dapat melakukan berbagai tugas, seperti mengambil data dari basis data, menjalankan aplikasi, atau menghasilkan halaman web dinamis.
5. **Pengiriman Respons**:
   - Setelah selesai memproses permintaan, server mengirim pesan HTTP Response kembali ke klien.
   - Pesan HTTP Response juga terdiri dari beberapa komponen utama:
   - Kode status (misalnya, 200 OK untuk permintaan yang berhasil atau 404 Not Found jika sumber daya tidak ditemukan).
   - Header HTTP (informasi tambahan seperti tipe konten respons, cookie, dan lain-lain).
   - Tubuh respons (data yang diminta atau respons lainnya).
6. **Penerimaan dan Tindak Lanjut Klien:**
   - Klien menerima pesan HTTP Response dari server.

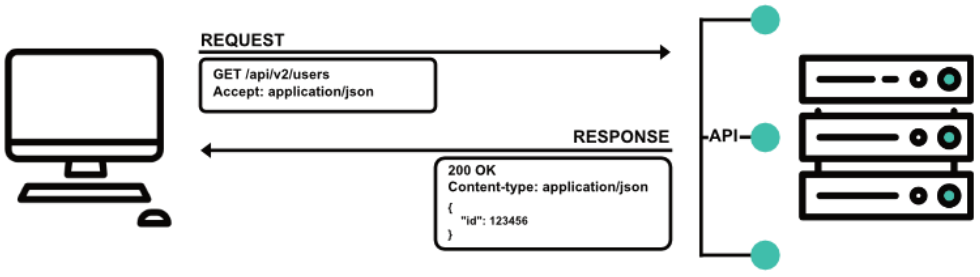
- Klien kemudian menginterpretasikan respons tersebut, memproses data (jika ada) dan menampilkan halaman web atau tindakan lainnya sesuai dengan respons tersebut.
- Klien juga dapat memutuskan untuk mengirim permintaan tambahan (seperti mengambil gambar atau file tambahan) jika diperlukan untuk menampilkan halaman sepenuhnya.

7. Penutupan Koneksi
   Setelah selesai, koneksi antara klien dan server bisa ditutup, tergantung pada aturan dan kepemrograman aplikasi.

B. API (Application Programming Interface)

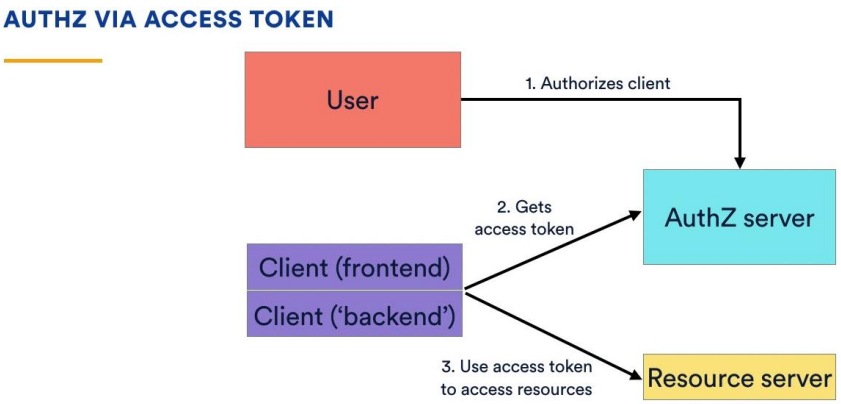
API adalah sebuah set aturan dan protokol yang memungkinkan berbagai perangkat lunak atau aplikasi untuk berkomunikasi satu sama lain. API adalah cara bagi pengembang perangkat lunak untuk mengintegrasikan fungsionalitas dari satu aplikasi ke dalam aplikasi lainnya, membuat aplikasi tersebut dapat bekerja bersama dan berbagi data atau layanan. Berikut adalah beberapa konsep dasar tentang bagaimana API bekerja:

1. **Abstraksi**: API menyediakan tingkat abstraksi atau antarmuka yang didefinisikan dengan jelas untuk berinteraksi dengan suatu sistem atau layanan. Ini berarti pengembang tidak perlu tahu detail internal dari sistem atau layanan yang mereka gunakan; mereka hanya perlu tahu cara menggunakan API tersebut.
2. **Kegunaan**: API digunakan untuk berbagai tujuan, termasuk mengakses data, berkomunikasi dengan perangkat keras, mengintegrasikan layanan pihak ketiga, dan banyak lagi. Sebagai contoh, API Google Maps memungkinkan pengembang untuk mengintegrasikan peta dan data lokasi ke dalam aplikasi mereka.
3. **Request-Response Model**: Sebagian besar API bekerja berdasarkan model permintaan-respons. Pengembang mengirimkan permintaan (request) kepada API dengan parameter yang sesuai, dan API memberikan respons (response) dengan data atau hasil yang diminta. Contoh umum adalah API REST (Representational State Transfer) yang menggunakan HTTP sebagai protokol komunikasi



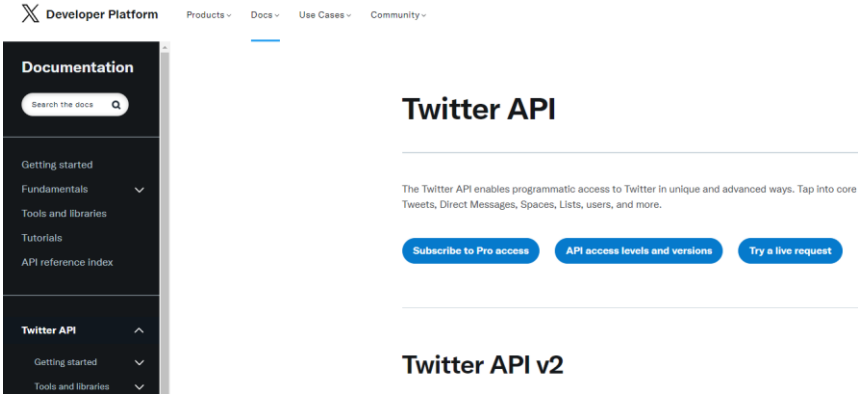
Gambar 2. Ilustrasi Request-responde model pada API

4. **Endpoint**: API memiliki berbagai endpoint, yang merupakan URL atau alamat yang spesifik untuk setiap fungsi atau aksi yang dapat diambil oleh API. Misalnya, dalam API Twitter, Anda mungkin memiliki endpoint untuk mengirim tweet, mengambil daftar pengikut, atau mencari tweet.
5. **Format Data:** Data yang dikirimkan dan diterima melalui API biasanya dalam format tertentu, seperti JSON (JavaScript Object Notation) atau XML (eXtensible Markup Language). Ini memungkinkan data untuk disusun secara terstruktur dan mudah dipahami oleh aplikasi.
6. **Autentikasi dan Otorisasi**: Untuk melindungi data dan layanan, API biasanya memiliki mekanisme autentikasi dan otorisasi. Autentikasi memverifikasi identitas pengguna atau aplikasi yang mengakses API, sementara otorisasi mengendalikan akses ke sumber daya atau layanan tertentu berdasarkan izin yang diberikan



Gambar 3. Ilustrasi Autentikasi dan Otorisasi API

7. **Dokumentasi**: Pengembang API menyediakan dokumentasi yang jelas untuk pengguna API. Dokumentasi ini menjelaskan bagaimana API berfungsi, endpoint yang tersedia, format permintaan dan respons, dan cara mengautentikasi.



Gambar 4. Dokumentasi API Twittter melalui https://developer.twitter.com/en/docs/twitter-api

Dalam dunia pengembangan perangkat lunak, API memiliki peran yang sangat penting karena memungkinkan aplikasi yang berbeda dibangun secara modular,

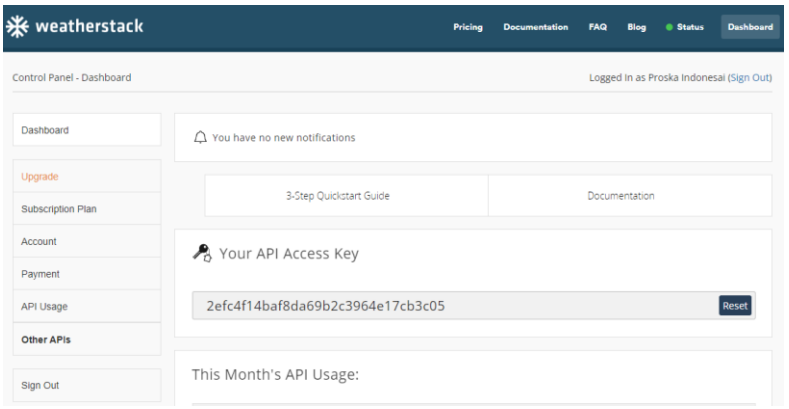
mempercepat pengembangan, dan memungkinkan integrasi yang lebih mudah antara berbagai sistem. Pengembang dapat menggunakan API dari penyedia layanan lain untuk memperluas fungsionalitas aplikasi mereka tanpa harus membangun semuanya dari awal.

LATIHAN

A. HTTP Request & API

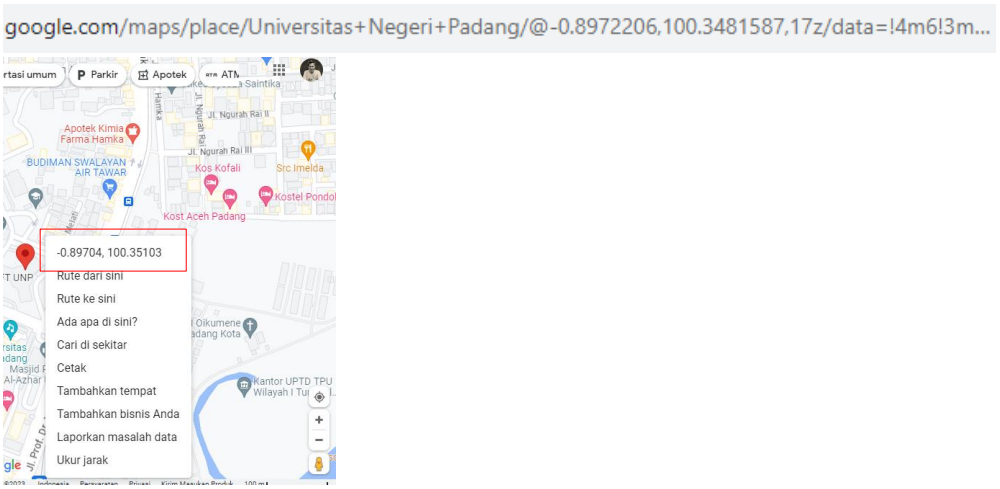
a. Mendapatkan API Access Key

1. Buatlah akun di https://weatherstack.com/. Bagian *company details* dapat dikosongkan.
2. Setelah itu silakan login dan anda akan masuk ke dashboard seperti gambar dibawah ini. Silakan perhatikan kode pada bagian **"Your API Secret key"** karena anda akan menggunakannya didalam program.



3. Klik **3-Step Quickstart Guide** dan kemudian perhatikan base URL http://api.weatherstack.com/ pada **Step 2: API Endpoints**. URL ini akan digunakan sebagai base URL untuk melakukan *request* ke API Weatherstack. Weatherstack menyediakan beberapa API Endpoints untuk mendapatkan data terkait cuaca diantaranya:
   a) Current Weather: Mendapatkan data cuaca terbaru
   b) Historical Weather: Mendapatkan data historis cuaca
   c) Historical Time-Series: Mendapatkan data historis cuaca dalam rangkaian waktu tertentu.
   d) Weather Forecast: Mendapatkan data ramalan cuaca hingga 14 hari.
   e) Location Lookup: Mencari satu atau beberapa lokasi.
4. Bukalah google map dan carilah **Universitas Negeri Padang**. Klik kana dibagian map dan akan tampil informasi terkait latitude dan longitude. Informasi ini juga terdapat pada url google maps. Latitude dan longitude pada URL dan

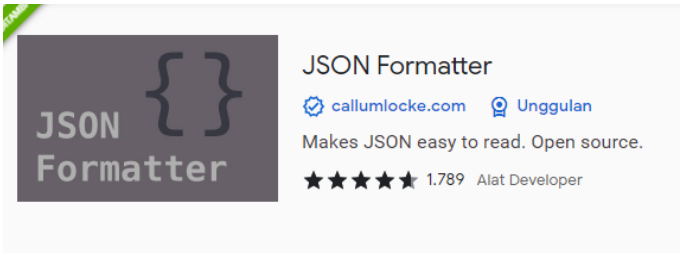
pada maps bisa saja berbeda tergantung dimana anda memposisikan klik kanan anda pada maps



5. Ketikan URL berikut di tab baru browser anda http://api.weatherstack.com/current?access_key=*****&query=*****,****** Gantilah tandan bintang (*) berwarna **merah** dengan kode access API key anda, warna **hijau** dengan latitude dan warna **cokelat** dengan longitude. Pastikan tidak ada angka atau karakter yang tertinggal termasuk tanda minus (-) jika ada
6. Tekan enter dan akan tampil seperti gambar dibawah ini

{"request":{"type":"LatLon","query":"Lat -0.90 and Lon 100.35","language":"en","unit":"m"},'
Sumatra","lat":"-0.949","lon":"100.354","timezone_id":"Asia\/Jakarta","localtime":"2023-09-1
AM","temperature":26,"weather_code":113,"weather_icons":["https:\/\/cdn.worldweatheronline.c
["Sunny"],"wind_speed":5,"wind_degree":212,"wind_dir":"SSW","pressure":1012,"precip":0,"humi

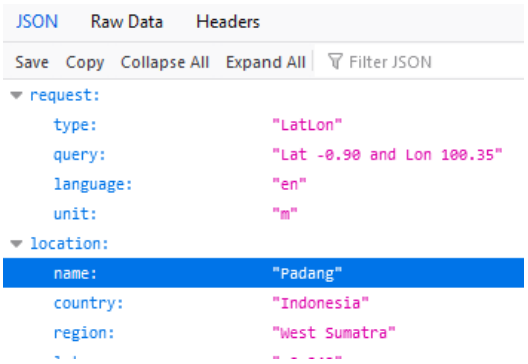
7. Tampilan diatas menggunakan format JSON. Agar tampilan menjadi lebih rapi, bukalah https://chrome.google.com/webstore/category/extensions kemudian cari JSON Formatters dan install ekstensi tersebut



8. Tampilan file JSON dari API Weatherstack akan menjadi seperti berikut ini

{
"request": {
"type": "LatLon",
"query": "Lat -0.90 and Lon 100.35",
"language": "en",
"unit": "m"
},

9. Jika anda menggunakan browser **Firefox**, maka **tidak perlu** menginstall extension atau add-on untuk tampilan seperti ini



10. Anda dapat menemukan informasi lengkap terkait cuaca di Universitas Negeri Padang melalui data yang ditampilkan seperti informasi lokasi, cuaca terkini, kapan cuaca di observasi, temperature, deskripsi dan lainnya. **PENTING!** Anda akan mengakses data ini melalui program

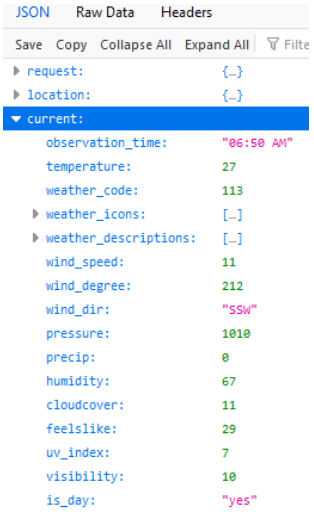
b. HTTP Request

1. Buatlah folder project baru pada visual studio code dengan nama **aplikasiCuaca**, lalu buatlah file javascript baru dalam folder tersebut dengan nama **app.js**.
2. Akses terminal melalui visual studio code, pastikan anda berada pada direktori folder yang baru anda buat. Lalu install **npm** dengan perintah **npm init** (sama seperti pada modul 2)
3. Setelah itu, silakan install module **postman-request** https://www.npmjs.com/package/postman-request dengan perintah **`npm i postman-request`**
4. Ketikanlah kode berikut di file app.js

```
1. const request = require('postman-request')
2. const url =
   'http://api.weatherstack.com/current?access_key=*****
   &query=*****,*****'
3. request({ url: url }, (error, response) => {
4. console.log(response)
5. // const data = JSON.parse(response.body)
6. // console.log(data)
7. // console.log(data.current)
8. // console.log(data.current.temperature)
9. })
```

5. Gantilah tanda bintang (*) seperti yang anda lakukan pada Langkah A no.5
6. Jalankan program tersebut melalui terminal dan pahami apa yang ditampilkan.
7. Jadikan kode pada baris ke-4 menjadi komentar dan kemudian **uncomment** baris ke-5 dan ke-6. Jalankan dan pahami apa yang ditampilkan
8. Jadikan kode pada baris ke-6 sebagai komentar, lalu **Uncomment** lagi kode pada baris ke 7 dan pahami apa yang ditampilkan

9. Jadikan kode pada baris ke-7 sebagai komentar, lalu **Uncomment** lagi kode pada baris ke-8 dan pahami apa yang ditampilkan.
10. Perlu diperhatikan pada const url bahwa **base url** untuk akses API menggunakan HTTP bukan HTTPS. Untuk menggunakan versi HTTPS maka dikenakan biaya. Pastikan ketika anda mengetikannya di browser juga menggunakan HTTP.
11. Perhatikan struktur file API dalam format JSON pada saat anda mengaksesnya melalui browser



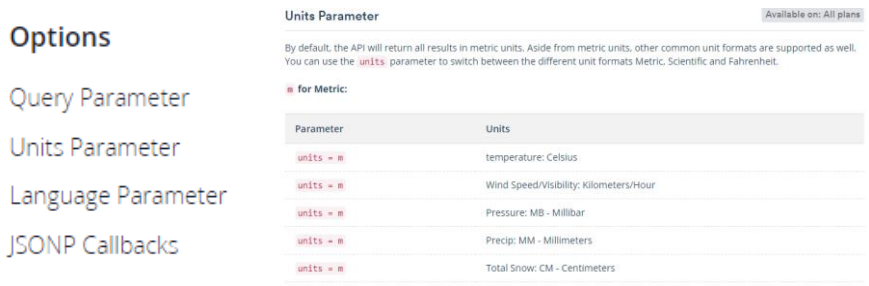
12. Terlihat bahwa data tersebut memiliki hirarki yaitu `request`, `location`, `current` dimana didalamnya memiliki sub-hirarki atau data lagi. Kode pada no.4 baris ke-8 menunjukkan bahwa anda telah mengakses `data -> current -> temperature`. Hal ini berarti anda juga dapat mengakses data lainnya. Sementara itu **`response.body`** menunjukkan bahwa anda mengakses keseluruhan data yang ditampilkan.
13. Buatlah file baru dengan nama **cekCuaca.js** dan ketikan kode berikut ini

```
const request = require('postman-request')
const urlCuaca =
'http://api.weatherstack.com/current?access_key=*****&query=**
***,*****'
request({ url: urlCuaca, json: true }, (error, response) => {
    console.log('Saat ini suhu diluar mencapai ' +
response.body.current.temperature +
        ' derajat celcius. Kemungkinan terjadinya hujan adalah
' + response.body.current.precip
        + '%')
})
```

14. **PENTING!** Jangan lupa mengganti tanda bintang dengan data anda masing-masing. Jalankan program tersebut dan pahami apa yang ditampilkan

15. Bukalah dokumentasi Weatherstack melalui link berikut ini https://weatherstack.com/documentation dan perhatikan bahwa Weatherstack memiliki beberapa opsi. Silakan klik **unit parameter**. Anda dapat melihat bahwa akan muncul units yang terkait dengan beberapa parameter suhu Celcius, Fahrenheit dan lainnya. Anda dapat menambahkan unit parameter ini keakhir pemanggilan API anda seperti berikut

```
const urlCuaca =
'http://api.weatherstack.com/current?access_key=*****&query=**
***,*****&units=m'
```



16. Gantilah huruf **m** dengan parameter lainnya yang tersedia pada dokumentasi weatherstack. Silakan jelajahi dokumentasi tersebut anda akan menemukan juga parameter *language* (bahasa) dan lainnya.

c. Latihan 1 – API Access Weatherstack

1. Pada kode yang ada pada Langkah b no.13, tambahkan kode agar anda bisa mengakses weather_descriptions (deskripsi cuaca)
2. Tampilkan teks berikut sesuai dengan keinginan anda.
3. Perlu diperhatikan bahwa data weather_descriptions menggunakan format array sehingga anda perlu menambah [] di akhir pemanggilan data. Misal: seandainya temperature dalam format array maka akan menjadi
   `response.body.current.temperature[0]`

d. Latihan 2 – API Mapbox

1. Silakan buat akun https://www.mapbox.com/**.** Setelah itu login dan cek menu **tokens.**
2. Bukalah dokumentasi API mapbox melalui link berikut ini https://docs.mapbox.com/api/search/geocoding/ Carilah **Forward Geocoding** (menu dibagian kanan)
3. Pahami dokumentasinya dan lihat bagaimana cara pemanggilan API nya. Perhatikan data yang ada pada tabel dan juga contoh pada **example request**
4. Berikut adalah salah satu example request yang mereka sediakan
   `https://api.mapbox.com/geocoding/v5/mapbox.places/Washington.json?limit=2&access_token=pk.eyJ1IjoicHJvc2thOTkiLCJhIjoiY2xsbmdkYWppMDBlYzNybXcwNmk2M3p6OSJ9.9S7iRjDXb1JqUbpKK949ew`

5. Gantilah teks warna hijau dengan kota atau tempat yang ingin anda cari, misal: **Padang** atau **Universitas Negeri Padang**. Lalu, ganti teks warna **merah** dengan token anda.
6. Cobalah akses di browser dan lihat data yang ditampilkan
7. Lalu cobalah lakukan **request data** dengan menambahkan kode berikut ke file **app.js**

```
const geocodeURL =
'https://api.mapbox.com/geocoding/v5/mapbox.places/*****
.json?access_token=*****&limit=1'

request({ url: geocodeURL, json: true }, (error,
response) => {
    const latitude = response.body.features[0].center[1]
    const longitude =
response.body.features[0].center[0]
    console.log(latitude, longitude)
})
```

8. Gantilah teks warna **hijau** dengan lokasi atau tempat yang ingin anda cari dan teks warna **merah** dengan token API Mapbox anda
9. `Limit=1` digunakan untuk membatasi pencarian data agar hanya menampilkan 1 pencarian. Cobalah ganti angka 1 menjadi 2 atau 3, lalu gantilah array `[0]` pada `features` dengan angka 1 atau 2. Lakukan untuk kedua variable `latitude` dan `longitude`
10. Perhatikanlah bahwa tampilan latitude dan longitude pada terminal telah berganti

```
-0.924759 100.363256
Saat ini suhu diluar mencapai 26 derajat celcius. Kemungkinan terjadinya hujan adalah 0%
```

```
-0.945967 100.358669
Saat ini suhu diluar mencapai 26 derajat celcius. Kemungkinan terjadinya hujan adalah 0%
```

Penting: output yang tampil tergantung pada lokasi yang anda acari. Meski limit anda tingkatkan menjadi lebih dari 1, data yang didapatkan tidak selalu lebih dari 1 karena bergantung kepada jumlah data yang disediakan oleh Mapbox API. Cobalah mencari lokasi yang popular seperti Jakarta atau Bali.

e. Latihan 3 – Memanggil data API

1. Lanjutkanlah program pada **bagian d.** Lakukanlah pemanggilan data sebagaimana yang telah anda pelajari pada saat mengakses API weatherstack untuk menampilkan query, place_name dan place_type. Contoh tampilannya adalah sebagai berikut

```
data yang anda cari adalah: Padang Barat
data yang ditemukan adalah: Padang Barat, Padang, West Sumatra, Indonesia
tipe lokasi adalah: locality
```

2. Lakukanlah pemanggilan data dari API weatherstack dan mapbox API hingga menampilkan output seperti berikut. **Penting**! Data lokasi bisa berbeda tergantung lokasi yang anda cari.Pastikan data lokasi yang anda cari dengan API mapbox sesuai dengan koordinat yang anda berikan pada API weatherstack

```
Koordinat lokasi anda adalah -0.924759, 100.363256
Data yang anda cari adalah: padang
Data yang ditemukan adalah: Padang Barat, Padang, West Sumatra, Indonesia
Tipe lokasi adalah: locality
Saat ini suhu di Padang mencapai 26 derajat celcius.
Kemungkinan terjadinya hujan adalah 0%
```

REFERENCES

1. Casciaro. (2014, December). Node.js Design Patterns (3rd ed.). Packt Publishing.
2. Pasquali, S., & Faaborg, K. (2017, December 29). Mastering Node.js: Build Robust and Scalable Real-Time Server-Side Web Applications Efficiently.
3. Guides | Node.js. (n.d.). Node.js. https://nodejs.org/en/docs/guides
4. *npm Docs*. (n.d.). Npm Docs. https://docs.npmjs.com/
5. Leka, M. (2022, June 8). *Exploring the JavaScript Ecosystem: Popular Tools, Frameworks, and Libraries*. Medium. https://mirzaleka.medium.com/exploring-javascript-ecosystem-popular-tools-frameworks-libraries-7901703ec88f