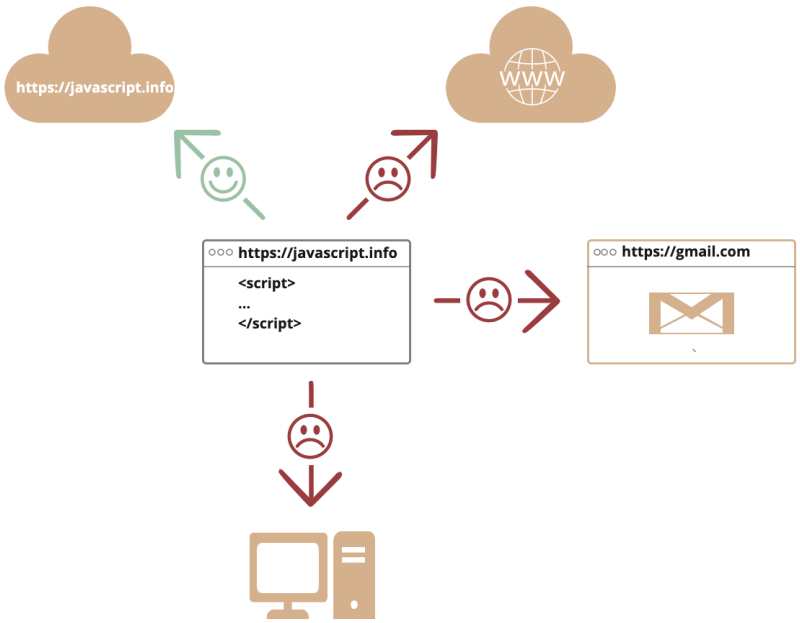
Praktikum 7
Web Service Rest API

A. Tujuan

1. Mampu memahami data JSON
2. Mampu memahami dan mempraktekkan Javascript
3. Mampu memahami konsep dan mempraktekkan Rest API

B. Dasar Teori

1. Java Script




```
1.  console.log("Hello, World!");
```

JavaScript adalah bahasa pemrograman tingkat tinggi yang pada awalnya dikembangkan untuk membuat website menjadi lebih "hidup". Bersama dengan HTML dan CSS, JavaScript menjadi bahasa pemrograman paling populer untuk mengembangkan aplikasi berbasis web. Bahasa ini mampu memberikan *logic* ke dalam website, sehingga website tersebut memiliki fungsionalitas tambahan dan lebih interaktif.

Awalnya JavaScript dibuat supaya dapat berjalan di lingkungan browser dan membuat website menjadi lebih interaktif. Namun, saat ini Anda sebagai developer dapat menggunakan bahasa pemrograman JavaScript untuk mengembangkan berbagai macam platform. Sehingga, tidak hanya sebatas browser/client, tetapi JavaScript juga bisa berjalan di luar browser menggunakan Node.js.

JavaScript termasuk ke dalam kategori *scripting language*. Apa maksudnya? Salah satu ciri-ciri utama dari bahasa *scripting* adalah kode tidak perlu dikompilasi agar bisa dijalankan.

Scripting language menggunakan *interpreter* untuk menerjemahkan kode atau perintah yang kita tulis supaya dimengerti oleh mesin.

Alernatif Online Editor dan Compiler :

1. glot.io (Node.js)
2. CodeSandbox (Node.js dan Browser)
3. Replit (Node.js dan Browser)
4. OwnCode (Browser)

Pada JavaScript setidaknya ada tiga cara untuk mendeklarasikan sebuah variabel, yaitu menggunakan keyword var, let, dan const.

```
let lastName;

lastName = "Skywalker";


console.log(lastName);
```

2. Json

   JavaScript object notation atau JSON adalah format yang digunakan untuk menyimpan dan mentransfer data. JSON memiliki struktur data yang sederhana dan mudah.
   JSON sendiri terdiri dari dua struktur, yaitu:

   a. Kumpulan *value* yang saling berpasangan. Dalam JSON, contohnya adalah object.
   b. Daftar *value* yang berurutan, seperti array.

```
{
  "first_name" : "Sammy",
  "last_name" : "Shark",
  "location" : "Ocean",
  "online" : true,
  "followers" : 987
}
```

3. Rest API

   REST atau **RE**presentational **S**tate **T**ransfer adalah salah satu gaya arsitektur yang dapat diadaptasi ketika membangun web service. Arsitektur ini sangat populer digunakan karena pengembangannya yang relatif mudah. REST menggunakan pola request-response dalam berinteraksi dan memanfaatkan protokol HTTP

   RESTful merupakan sebutan untuk web services yang menerapkan arsitektur REST. REST juga merupakan API (*application program interface*) karena ia digunakan untuk menjembatani antara sistem yang berbeda (client dan server).

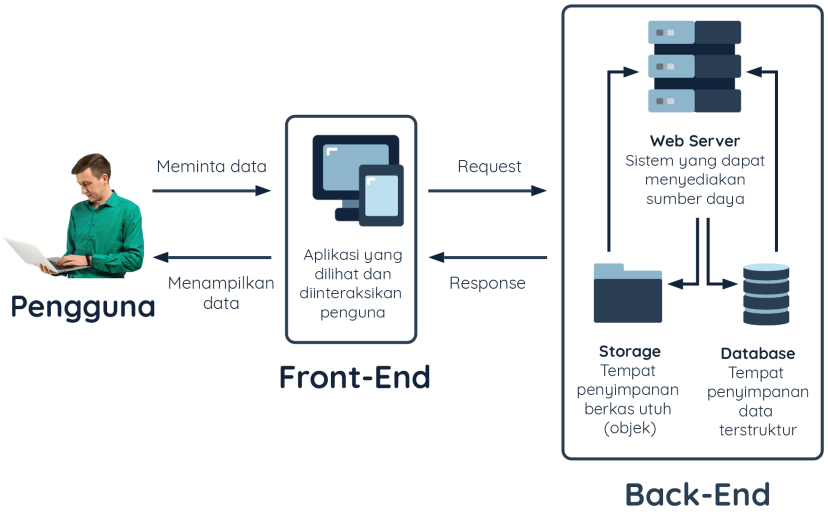
   API atau Application Program Interface merupakan antarmuka yang menjadi perantara antara sistem aplikasi yang berbeda. API tak hanya dalam bentuk Web Service, bisa saja berupa SDK (Software Development Kit) ataupun lainnya.

ketika membangun REST API kita harus memperhatikan empat poin berikut:

- Format request dan response.
- HTTP Verbs/Methods.
- HTTP Response code.
- URL Design

Back-End merupakan bagian dari aplikasi yang bertanggung jawab untuk menyediakan kebutuhan yang tak terlihat oleh pengguna (tidak berinteraksi langsung dengan pengguna), seperti bagaimana data disimpan, diolah, serta ditransaksikan secara aman. Itu semua bertujuan untuk mendukung aplikasi Front-End bekerja sesuai dengan fungsinya. Sosok yang menggeluti bidang ini disebut *Back-End Developer.*

Server merupakan sebuah sistem yang dapat menyediakan sumber daya berupa data, layanan, atau program untuk disajikan ke komputer lain. Ingat! Pengertian dari server bukanlah sebuah perangkat keras ataupun komputer, namun server sendiri lebih merujuk kepada sistem yang dapat membuat perangkat (termasuk komputer) dapat melayani sebuah permintaan dari perangkat lain. Jika diterjemahkan ke dalam Bahasa Indonesia, server memang berarti penyaji, atau pelayan. Server bertugas untuk melayani sebuah layanan (services) atau jasa. Dalam dunia komputer ada banyak service yang dapat dilayani oleh server.



Kita perlu membuat dan menjalankan program (mirip seperti SOP) agar dapat menentukan logika bisnis sesuai dengan kebutuhan. Program tersebut perlu disimpan di server dan dapat diakses secara remote melalui internet atau intranet agar aplikasi Front-End dan Back-End dapat saling terhubung. Jadi untuk membuat sistem aplikasi setidaknya membutuhkan:

- **Web Server** : Server yang dapat menjalankan program dan dapat diakses melalui internet atau intranet.
- **Web Service** : Program yang dijalankan di web server agar kebutuhan bisnis terpenuhi.

Web service berjalan di dalam web server sehingga ia dapat diakses melalui internet. Melalui web service inilah aplikasi Front-End (*client*) dan Back-End dapat bertransaksi.

C. Praktikum

Environment yang digunakan dalam praktek :

1. Javascript
2. Instal node.js : https://nodejs.org/en/
3. Expressjs : https://expressjs.com/en/starter/installing.html
4. Instal Insomnia : https://insomnia.rest/download dan https://docs.insomnia.rest/ (digunakan untuk pengujian terhadap API yang sudah dibuat)
5. Instal Vscode : sebagai editor code
6. JSON : format data yang digunakan untuk disimpan dan ditransfer
7. HTTP : protokol yang digunakan

I. JavaScript

1. Percobaan Percabangan

```
let language = "French";
let greeting = "Selamat Pagi"

if (language === "English") {
        greeting = "Good Morning!";
} else if (language === "French") {
        greeting = "Bonjour!"
} else if (language === "Japanese") {
        greeting = "Ohayou Gozaimasu!"
}

console.log(greeting);
```

2. Percobaan Perulangan

```
let myArray = ["Luke", "Han", "Chewbacca",
"Leia"];

for(const arrayItem of myArray) {
  console.log(arrayItem)
}
```

3. Pembuatan Objek

   Object mampu menyimpan nilai dari beragam tipe data dan membentuk data yang lebih kompleks. Object berisi pasangan *key* dan *value* yang juga dikenal dengan *property*. Key berperan mirip seperti nama variabel yang menyimpan sebuah nilai. Sementara, value berisi nilai dengan tipe data apa pun termasuk objek lain.

```
const user = {
  firstName: "TRI",
  lastName: "PENS",
```

```
  age: 2,
  isJedi: true,
};
console.log(`Halo, nama saya ${user.firstName} ${user.lastName}`);
console.log(`Umur saya ${user.age} tahun`);
```

4. Fungsi / Metode pada Javascript

```
const user = {
  id: 24,
  displayName: 'TRI',
  fullName: 'Teknologi Rekayasa Internet',
};

function introduce({displayName, fullName}) {
  console.log(`${displayName} adalah ${fullName}`);
}

introduce(user);
```

5. Arrow Function

   Arrow function mirip seperti regular function secara perilaku, tetapi berbeda dalam penulisannya. Sesuai namanya, fungsi didefinisikan menggunakan tanda panah atau fat arrow (`=>`).

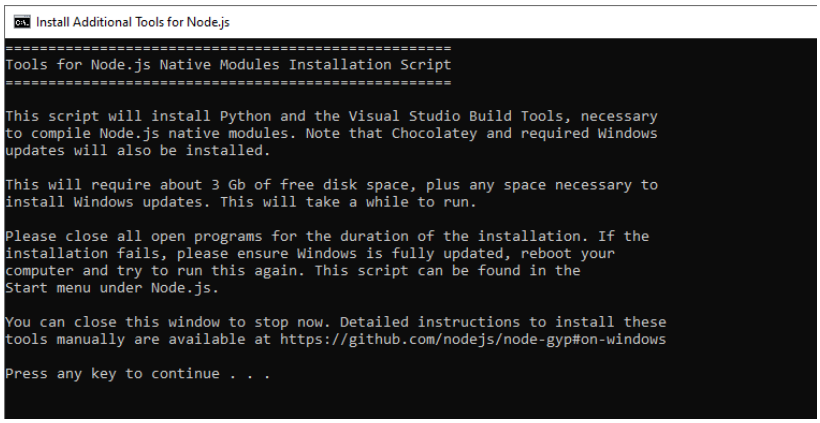
```
const sayName = (name) => {
  console.log(`Nama saya ${name}`)
  console.log("Politeknik Elektronika Negeri Surabaya");
}

sayName("TRI");
```

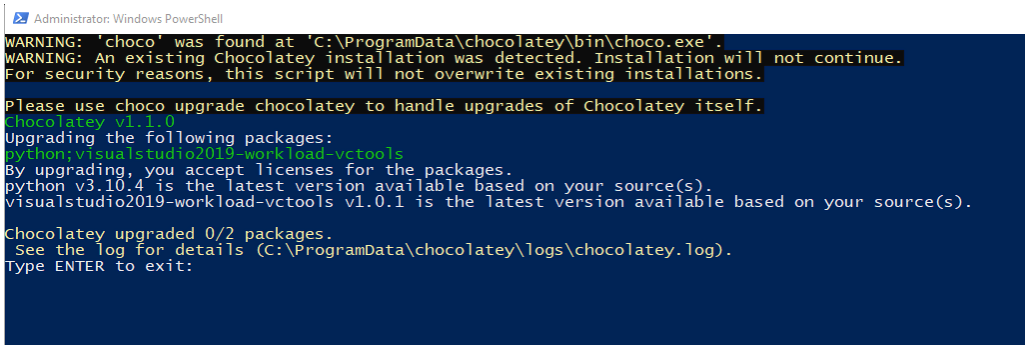
II. Dasar-dasar Node Js

Node.js adalah javascript runtime Environment

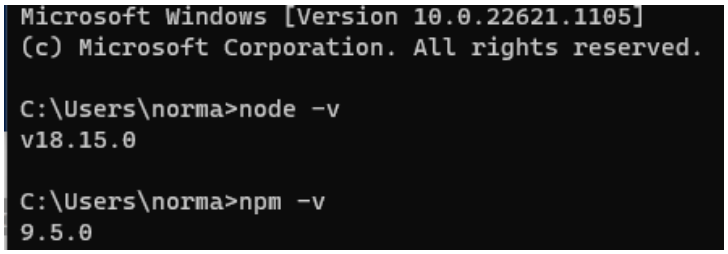
1. Download dan Instal node js : https://nodejs.dev/en/download/
2. Setelah penginstalan default selesai, Command Prompt akan terbuka dan menampilkan setting tambahan. Tekan dua kali tombol apa pun untuk melanjutkan.



3. Setelah proses selesai, Command Prompt akan meminta Anda menekan Enter untuk menutup jendela.



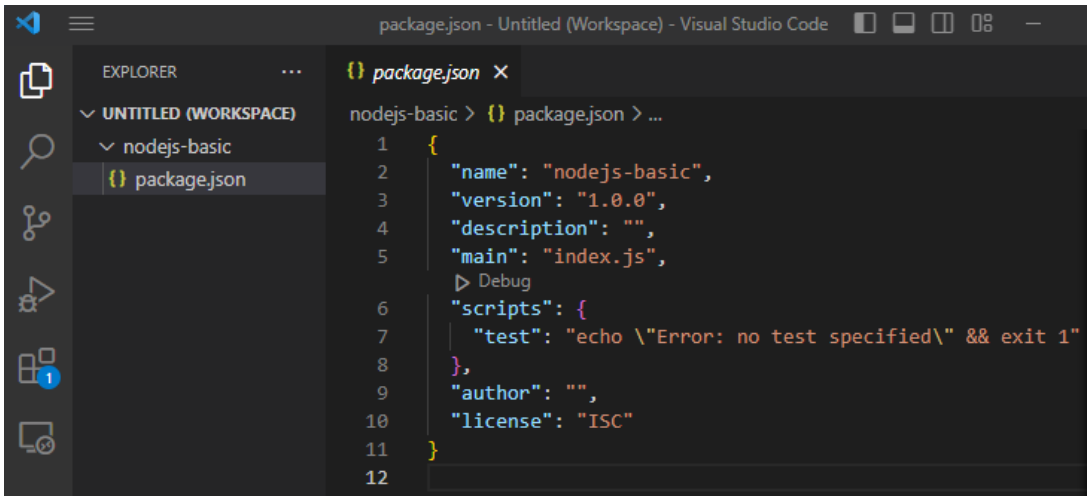
4. Untuk memverifikasi versi Node.js, buka **Command Prompt** dan jalankan command berikut:



5. membuat proyek node js dengan buatlah folder baru terlebih dahulu misal pada *C -> javascript-projects -> nodejs-basic*
6. buka folder *nodejs-basic* menggunakan VSCode. Caranya, pada Visual Studio Code pilih menu *File -> Open Folder -> [pilih foldernya]*.
7. Untuk membuat proyek JavaScript, silakan buka Terminal pada VSCode. Pilih menu *Terminal -> New Terminal*, kemudian tuliskan perintah: npm init
   NPM alias Node Package Manager merupakan JavaScript Package Manager bawaan dari Node.js. Melalui NPM ini kita dapat membuat Node.js package (proyek) dan mengelola penggunaan package eksternal yang digunakan.
8. Setelah menuliskan perintah di atas, Anda akan diberikan beberapa pertanyaan untuk mengisi nilai *package name, version, description*. dapat menggunakan nilainya dengan langsung menekan tombol *Enter*.

```
Press ^C at any time to quit.
package name: (nodejs-basic)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to C:\javascript-project\nodejs-basic\package.json:
```

9. Setelah mengisi seluruh pertanyaan yang diberikan, Anda akan diberitahu untuk melihat hasil akhir yang dibuat pada berkas package.json.



10. buat file baru dengan nama index.js
11. tuliskan kode berikut

```
const message = (name) => {
    console.log(`Hello ${name}`);
}

message('JavaScript');
```

12. eksekusi perintah diatas pada terminal dengan : node index.js
13. Modularization

    Modularisasi dalam pemrograman merupakan teknik pemisahan kode menjadi modul-modul yang bersifat independen namun bisa saling digunakan untuk membentuk suatu program yang kompleks. Pada Node.js, setiap berkas JavaScript adalah modul. Dimana dapat membagikan nilai variabel, objek, class, atau apa pun itu antar modul. Untuk melakukannya, perlu untuk mengekspor nilai pada module tersebut. Untuk mengekspornya, simpanlah nilai tersebut pada properti module.exports.

coffee.js

```
1.  const coffee = {
2.      name: 'Tubruk',
3.      price: 15000,
4.  }
5.
6.
7.  module.exports = coffee;
```

app.js

```
1.  const coffee = require('./coffee');
2.
3.  console.log(coffee);
```

Setelah itu nilai coffee dapat digunakan pada berkas JavaScript lain dengan cara mengimpor nilainya melalui fungsi global require(). Perhatikan nilai parameter yang diberikan pada require(). Parameter merupakan lokasi dari module target impor. Ingat! Jika Anda hendak mengimpor modul lokal (*local module*), selalu gunakan tanda ./ di awal alamatnya ya.

III. Membuat REST API pada framework expressjs yang dijalankan pada node.js

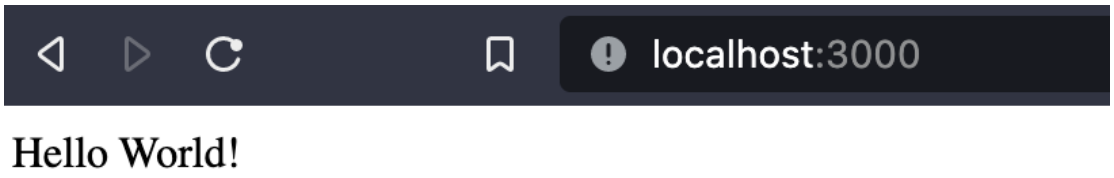
1. Instal Express di directory yang sedang aktif
   `$ npm install express`
2. pada vscode ketikkan perintah berikut :

```
const express = require('express')
const app = express()
const port = 3000

app.get('/', (req, res) => {
  res.send('Hello World!')
})

app.listen(port, () => {
  console.log(`Example app listening on port ${port}`)
})
```

1. pada directory myapp simpan code diatas dengan nama file yaitu app.js
2. Run aplikasi dengan perintah berikut :
   `$ node app.js`
3. Kemudian load pada browser dengan hostname berikut
   load http://localhost:3000/
   akan muncul tampilan berikut



Ket :
Aplikasi ini memulai server dan mendengarkan pada port 3000 untuk koneksi. Aplikasi merespons dengan "Halo World!" untuk permintaan ke URL root (/) atau route. Untuk setiap jalur lainnya, itu akan merespons dengan 404 Not Found.

IV. Membuat endpoint baru untuk case perpustakaan

1. buat rancangan 2 end point baru, misal
   hostname/buku/{idbuku} => mendapatkan data
   hostname/buku/tambah => membuat data
2. Membuat daftar buku dengan Array
   a. Ketikkan Perintah berikut pada vscode dan simpan dengan nama dataBuku.js

```
const express = require('express')
const app = express()
const port = 3000

let daftarBuku = []

app.use(express.json())

app.get('/', (req, res) => {
  res.send('Hello World!')
})

// mendapatkan data buku
app.get('/buku/:id_buku', (req, res) => {
        const bukuYangDicari = req.params.id_buku
        const indexHasilPencarian = daftarBuku.findIndex((buku) => buku ==
bukuYangDicari)
        if (indexHasilPencarian >= 0) {
        res.json({
        "ok": true,
        "message": "Buku " + bukuYangDicari + " ditemukan"
        });
        } else {
        res.json({
        "ok": false,
        "message": "Buku " + bukuYangDicari + " tidak ditemukan dalam database"
        });
        }
})
```

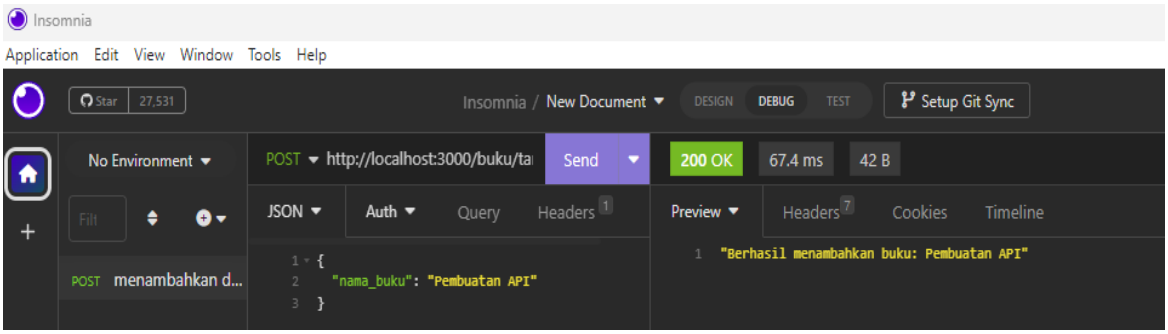
```
// menambahkan data buku
app.post('/buku/tambah', (req, res) =>{
        const namaBukuBaru = req.body.nama_buku
        daftarBuku.push(namaBukuBaru)

        res.json("Berhasil menambahkan buku: " + namaBukuBaru)
})
app.listen(port, () => {
  console.log(`Example app listening on port ${port}`)
})
```

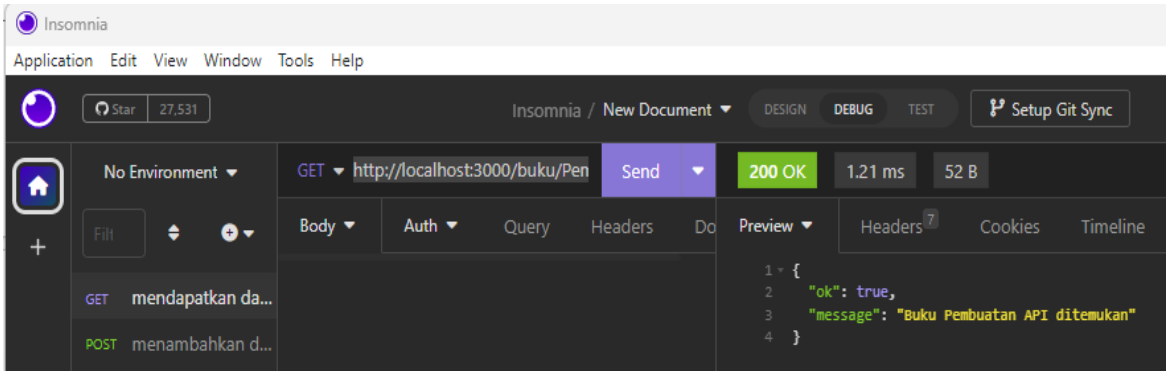
b. running program pada terminal : node dataBuku.js

3. buat simulasi client pada insomnia

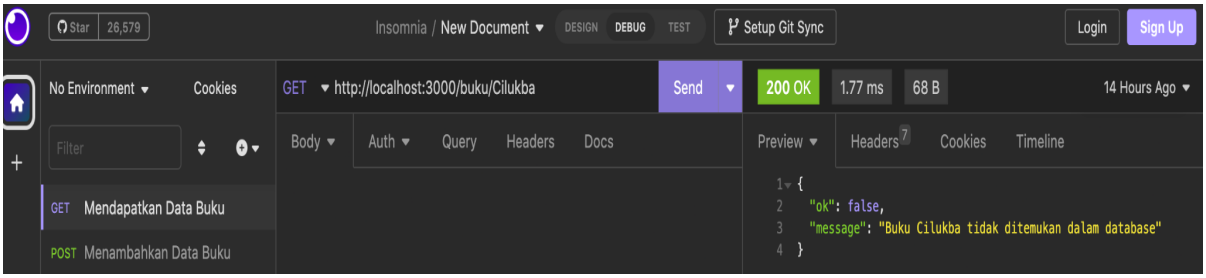
http://localhost:3000/buku/tambah (metode POST)



http://localhost:3000/buku/Pembuatan API (metode GET)



jika menginputkan data yang salah



D. Tugas

Buatlah Rest API untuk menambahkan dan mendapatkan mata kuliah yang ada di Program Studi Teknologi Rekayasa Internet PENS (mengguankan metode POST dan GET).

E. Laporan Resmi

1. Analisalah semua program diatas dan buat kesimpulan