

NIM : 3312411077

Nama : Anisa Frity Amelia

Kelas : IF3A Web Pagi

Link Github : <https://github.com/anisafrityamelia/praktikum3.git>

Praktikum 3 - Matakuliah Pilihan 1 (Web)

Program Studi: Teknik Informatika

Lakukan praktikum dibawah ini, dan buat screenshot untuk pembuktian mengerjakan setiap poin dengan mengisi tabel dibawah, kemudian tunjukan hasil akhir dari men-share repository github yang telah dibuat.

A. Membuat Server API dengan Express.js

1. Buat sebuah folder proyek API dengan nama **APIproject1**
2. Dengan command line masuk kedalam folder tersebut dan melakukan inisialisasi nodejs dengan mengetik perintah seperti pada gambar dibawah ini. (npm init -y)

```
PS D:\expressjs> cd .\APIproject1\  
PS D:\expressjs\APIproject1> npm init -y  
Wrote to D:\expressjs\APIproject1\package.json:  
  
{  
  "name": "apiproject1",  
  "version": "1.0.0",  
  "description": "",  
  "main": "index.js",  
  "scripts": {  
    "test": "echo \"Error: no test specified\" && exit 1"  
  },  
  "keywords": [],  
  "author": "",  
  "license": "ISC"  
}
```

3. Install [express.js](#) di folder tersebut dengan mengetik perintah (npm install express) seperti dibawah ini

```
PS D:\expressjs\APIproject1> npm install express  
  
added 68 packages, and audited 69 packages in 2s  
  
16 packages are looking for funding  
  run `npm fund` for details  
  
found 0 vulnerabilities
```

4. Pada folder tersebut buatlah file bernama [server.js](#)

B. Membuat API pertama

1. Pada file [server.js](#) ketik seperti pada kode program dibawah ini

```
JS server.js > ...
1  const express = require('express');
2  const app = express();
3  const PORT = 8001;
4
5  app.use(express.json());
6
7  app.get('/', (req, res) => {
8    res.send('Hello, World');
9  });
10
11 app.listen(PORT, () => {
12   console.log(`Server berjalan di http://localhost:${PORT}`);
13 });
14
```

2. Kemudian jalankan aplikasi dengan mengetik pada console, n Node server.js
3. Gunakan browser untuk mengecek hasilnya menuju ke localhost:8001
4. Port bisa diubah apabila saat dijalankan bentrok dengan port lain

C. Menambahkan Routes (Api Endpoints)

1. Buat route baru di file [server.js](#) dengan mengetik kode program seperti dibawah ini

```
app.get('/api/about', (req, res) => {
  res.json([
    { id: 1, name: 'Andi', job: 'Senior Programmer' },
    { id: 2, name: 'Budi', job: 'Technical Report' },
    { id: 3, name: 'Cindy', job: 'Front-end Programmer' },
    { id: 4, name: 'Deli', job: 'UI/UX Designer' },
    { id: 5, name: 'Erlang', job: 'Marketing' }
  ]);
});
```

2. Setelah dibuat, restart server API, dengan menutup server yang berjalan di command line, kemudian jalankan kembali node [server.js](#)
3. Gunakan browser untuk mengecek routes yang baru dibuat dengan menuju ke alamat /api/about
4. Praktikum C ini membuat **API Endpoints** baru dengan lokasi **/api/about**
5. Screenshot hasil API endpoints dari browser

D. Menambahkan Bearer token pada API (Keamanan level 1)

1. Pada bagian atas kode program buat konstanta baru untuk **TOKEN** dan fungsi baru **authBearer** seperti pada gambar dibawah ini.

```
APIproject1 > JS server.js > authBearer
1  const express = require('express');
2  const app = express();
3  const PORT = 8001;
4
5  // Token rahasia (hardcode, bisa juga dari .env)
6  const TOKEN = "mysecrettoken";
7
8  // Middleware sederhana untuk cek Bearer Token
9  function authBearer(req, res, next) {
10     const authHeader = req.headers['authorization'];
11     const token = authHeader && authHeader.split(' ')[1];
12
13     if (!token) {
14         return res.status(401).json({ error: 'Token tidak ditemukan' });
15     }
16
17     if (token !== TOKEN) {
18         return res.status(403).json({ error: 'Token salah atau tidak valid' });
19     }
20
21     next();
22 }
23
24
25 app.use(express.json());
26 app.listen(PORT, () => {
27     console.log(`Server berjalan di http://localhost:${PORT}`);
28 });
```

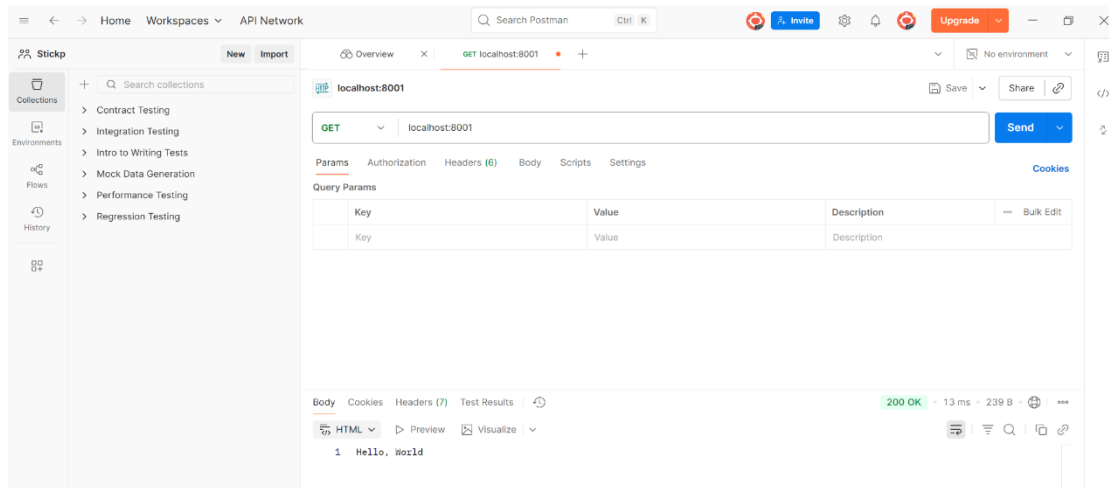
2. Buat **API Endpoints** baru dengan menggunakan authBearer seperti pada kode program dibawah ini

```
app.get('/api/payments', authBearer, (req, res) => {
    res.json([
        { id: 101, user: 'Andi', amount: 150000, status: 'Success', method: 'Bank Transfer' },
        { id: 102, user: 'Budi', amount: 250000, status: 'Pending', method: 'Credit Card' },
        { id: 103, user: 'Cici', amount: 50000, status: 'Failed', method: 'E-Wallet' }
    ]);
});
```

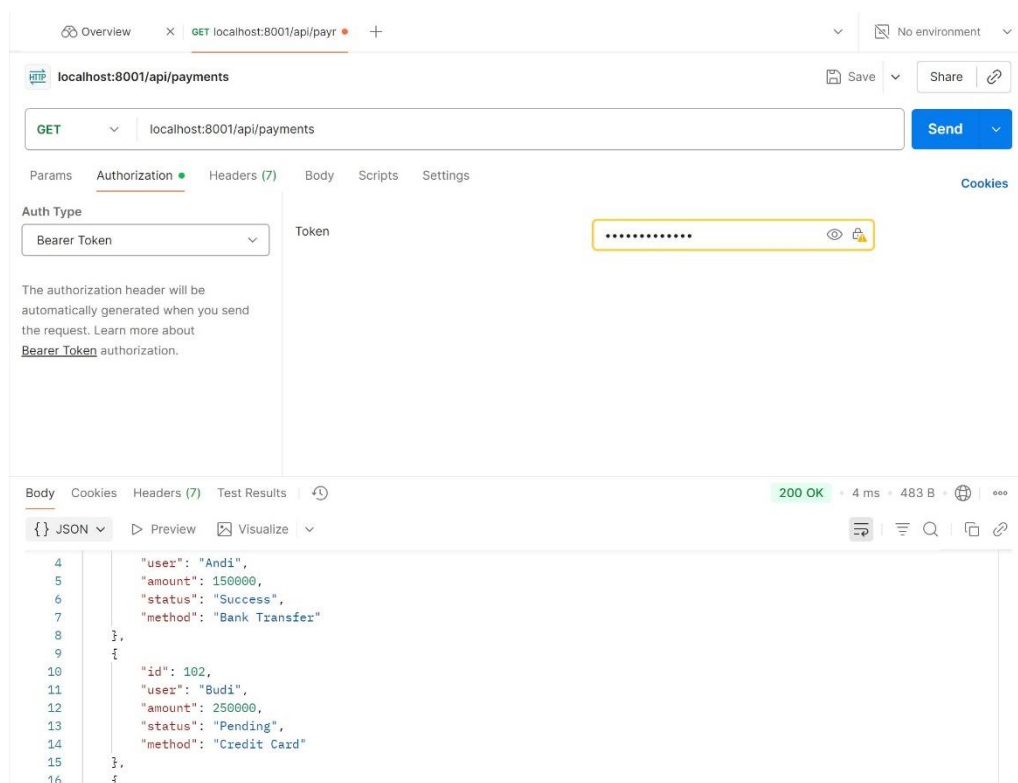
3. Restart node [server.js](#)

E. Membuka API dengan POSTMAN

1. Download aplikasi postman di <https://www.postman.com/downloads/>
2. Kemudian lakukan instalasi postman, dan login dengan google anda
3. Buka tab baru, lalu menuju GET localhost:8001 lalu tekan tombol **Send**



4. Lakukan hal serupa untuk membuka endpoints `/api/aobut` dan `/api/payments`
5. Apa yang terjadi?
6. Untuk membuka endpoints `api/payments` maka masukan bearer token sesuai dengan kode program TOKEN kalian.
7. Pada tab Authorization pilih Auth Type menjadi bearer token, lalu isi token dengan **“mysecrettoken”**



F. Github + Visual Code

1. Buat proyek di Github dengan nama **Latihan2**

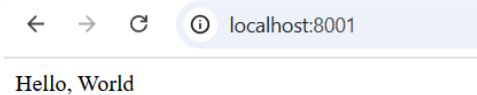
git init



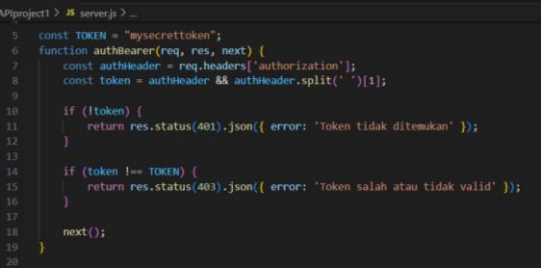
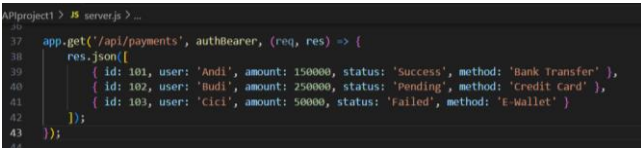
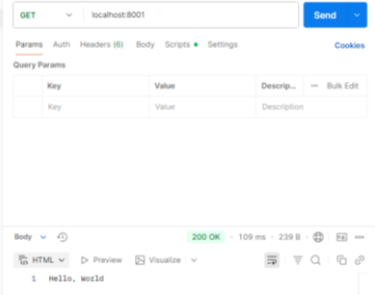
git add .

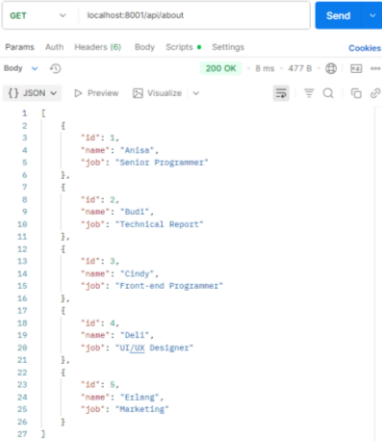
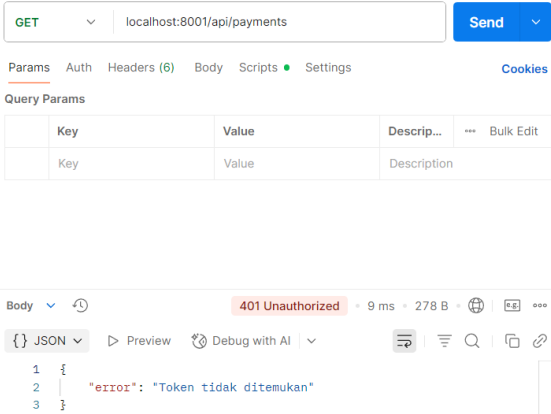
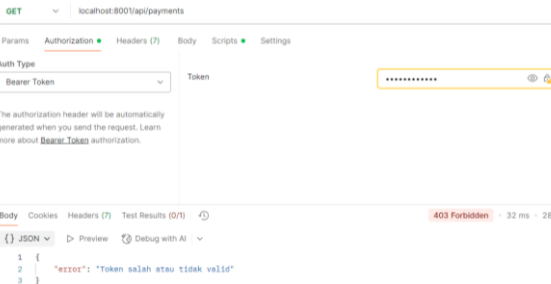
git commit -m "first commit"

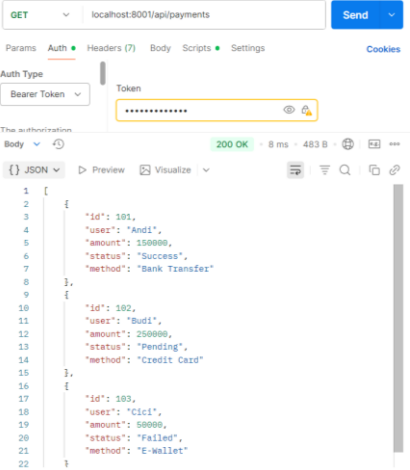
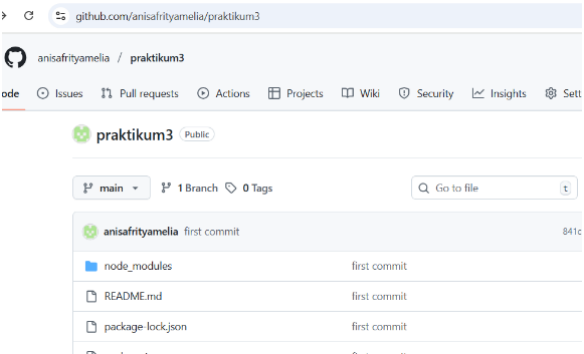
git branch -M main
git remote add origin <https://github.com/agunghakase/Latihan3.git>
git push -u origin main

Hasil Pengerjaan

No	Instruksi	Screenshot	Kendala/ Saran
A	Membuat Server API dengan Express.js		
1.	Membuat folder proyek API dengan nama APIproject1, kemudian dengan command line masuk ke folder tersebut dan melakukan inisialisasi nodejs dengan perintah npm init -y		
2.	Menginstall express.js di folder APIproject1 dengan perintah npm install express		
B	Membuat API pertama		
1.	Pada folder APIproject1 buat file bernama server.js dan ketik kode program seperti gambar disamping		
2.	Jalankan aplikasi dengan mengetik node server.js pada console dan gunakan browser untuk mengecek hasilnya		
C	Menambahkan Routes (Api Endpoints)		

1.	Buat route baru di file server.js dengan mengetik kode program seperti pada gambar	 <pre> APIproject1 > JS server.js > ... 27 app.get('/api/about', (req, res) => { 28 res.json([29 {id: 1, name: 'Anisa', job: 'Senior Programmer'}, 30 {id: 2, name: 'Budi', job: 'Technical Report'}, 31 {id: 3, name: 'Cindy', job: 'Front-end Programmer'}, 32 {id: 4, name: 'Deli', job: 'UI/UX Designer'}, 33 {id: 5, name: 'Erlang', job: 'Marketing'}, 34]); 35 }); </pre>	
2.	Jalankan kembali node server.js dan gunakan browser untuk mengecek routes baru menuju ke alamat /api/about	 <pre> localhost:8001/api/about Pretty-print [{ "id": 1, "name": "Anisa", "job": "Senior Programmer" }, { "id": 2, "name": "Budi", "job": "Technical Report" }, { "id": 3, "name": "Cindy", "job": "Front-end Programmer" }, { "id": 4, "name": "Deli", "job": "UI/UX Designer" }, { "id": 5, "name": "Erlang", "job": "Marketing" }] </pre>	
D	Menambahkan Bearer token pada API (Keamanan level 1)		
1.	Pada bagian atas program tambahkan konstanta baru untuk token dan fungsi baru authBearer seperti gambar disamping	 <pre> APIproject1 > JS server.js > ... 5 const TOKEN = "mysecrettoken"; 6 function authBearer(req, res, next) { 7 const authHeader = req.headers['authorization']; 8 const token = authHeader && authHeader.split(' ')[1]; 9 10 if (!token) { 11 return res.status(401).json({ error: 'Token tidak ditemukan' }); 12 } 13 14 if (token !== TOKEN) { 15 return res.status(403).json({ error: 'Token salah atau tidak valid' }); 16 } 17 next(); 18 } </pre>	
2.	Buat API Endpoints baru dengan menggunakan authBearer seperti pada gambar disamping	 <pre> APIproject1 > JS server.js > ... 37 app.get('/api/payments', authBearer, (req, res) => { 38 res.json([39 { id: 101, user: 'Andi', amount: 150000, status: 'Success', method: 'Bank Transfer' }, 40 { id: 102, user: 'Budi', amount: 250000, status: 'Pending', method: 'Credit Card' }, 41 { id: 103, user: 'Cici', amount: 50000, status: 'Failed', method: 'E-Wallet' }, 42]); 43 }); </pre>	
E	Membuka API dengan POSTMAN		
1.	Buka aplikasi postman, lalu menuju GET localhost:8001, maka akan menampilkan hasil seperti gambar disamping		

2.	Buka aplikasi postman, lalu menuju GET localhost:8001/api/about, maka akan menampilkan hasil seperti gambar disamping		
3.	Buka aplikasi postman, lalu menuju GET localhost:8001/api/payment, maka akan menampilkan hasil seperti gambar disamping	<p>a) API menolak akses, muncul pesan error 401 (token tidak ditemukan) karena token tidak diberikan.</p>  <p>b) API mengembalikan pesan 403 (token salah atau tidak valid) karena token salah.</p>  <p>c) Agar bisa mengakses /api/payment masukkan bearer token yang benar yaitu mysecrettoken pada authorization, sehingga /api/payment dapat diakses.</p>	

		 <pre> 1 [2 { 3 "id": 101, 4 "user": "Andi", 5 "amount": 150000, 6 "status": "Success", 7 "method": "Bank Transfer" 8 }, 9 { 10 "id": 102, 11 "user": "Budi", 12 "amount": 250000, 13 "status": "Pending", 14 "method": "Credit Card" 15 }, 16 { 17 "id": 103, 18 "user": "Cici", 19 "amount": 50000, 20 "status": "Failed", 21 "method": "E-wallet" 22 } 23] </pre>	
F	Github + Visual Code		
1.	Membuat repository baru di Github dan commit proyek ke dalam Github		
2.	Link Github	https://github.com/anisafrityamelia/praktikum3.git	