

Praktikum 3 - Matakuliah Pilihan 1 (Web)

Program Studi: Teknik Informatika

Nama: Salsa Putri Ajriyanti

Nim: 3312401043

Kelas: IF 3A Pagi

Lakukan praktikum dibawah ini, dan buat screenshot untuk pembuktian mengerjakan setiap poin dengan mengisi tabel dibawah, kemudian tunjukkan 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 });
```

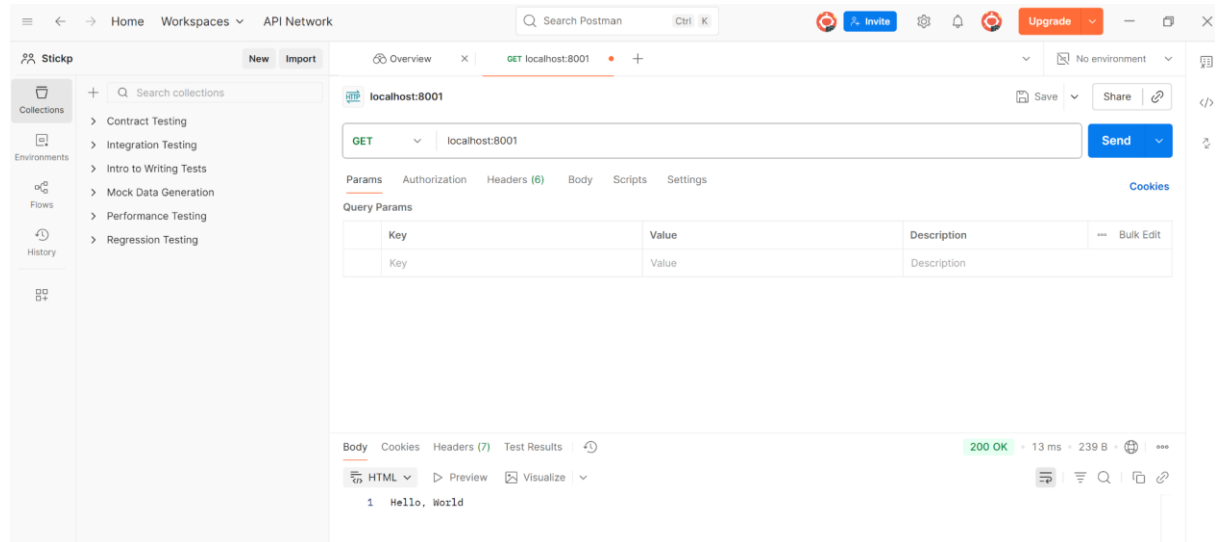
2. Buat **API Endpoints** baru dengan menggunakan authBearer seperti pada kode program dibawah ini

```
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' }
    ]);
});
```

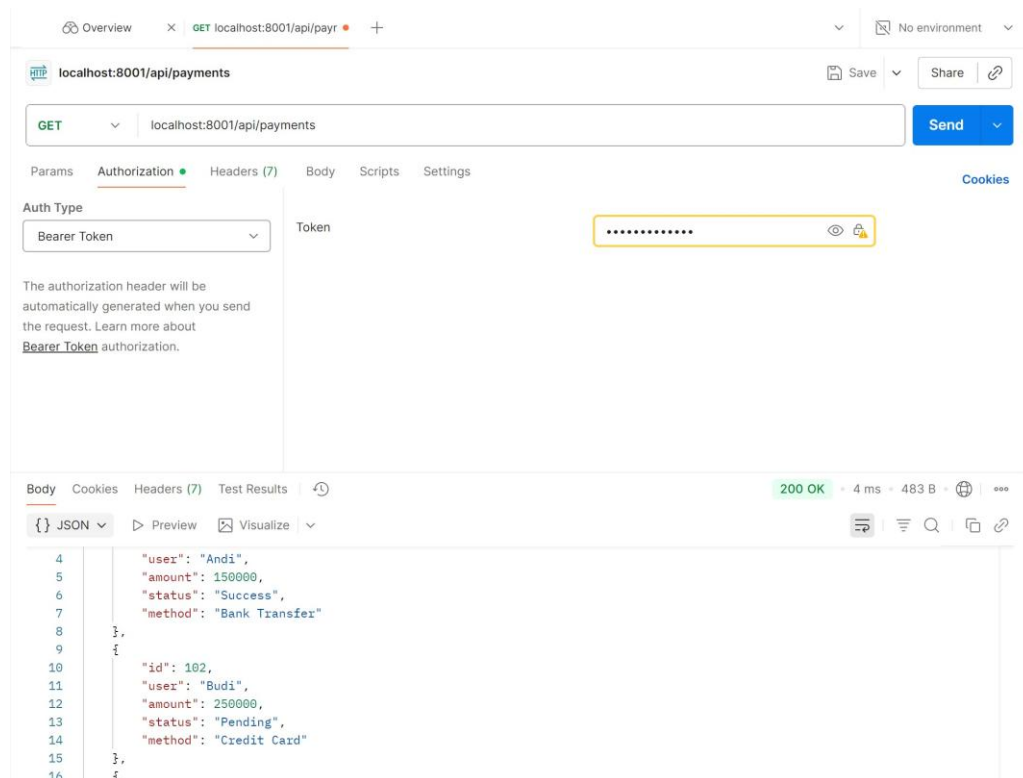
3. Restart node [server.js](#)

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”**



E. 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`

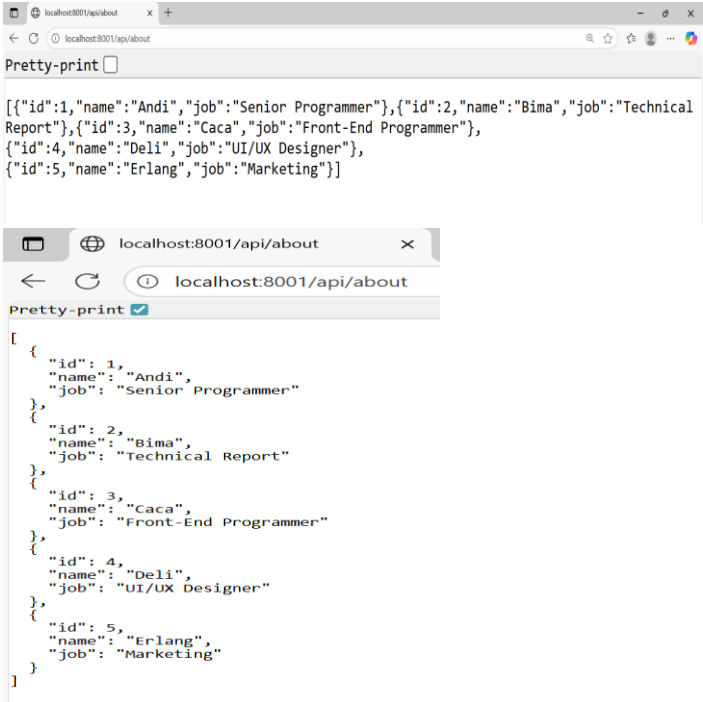
`origin https://github.com/agunghakase/Latihan3.git git push -u`

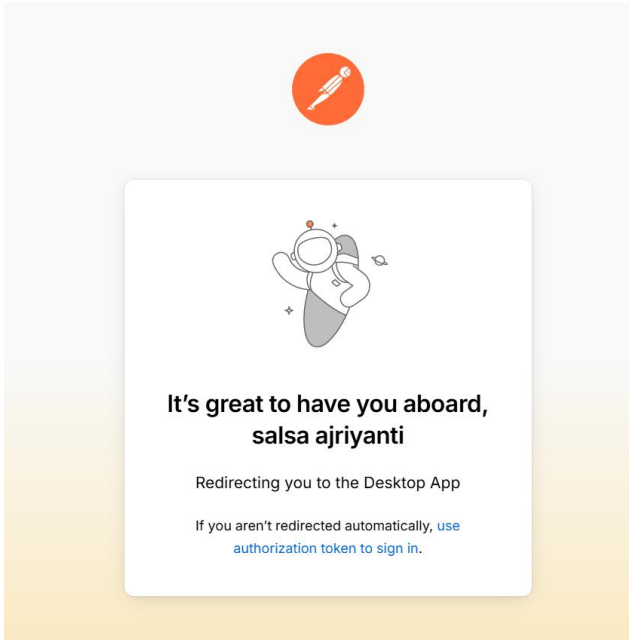
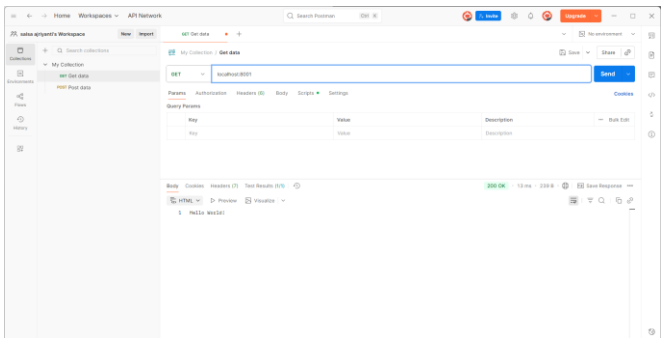
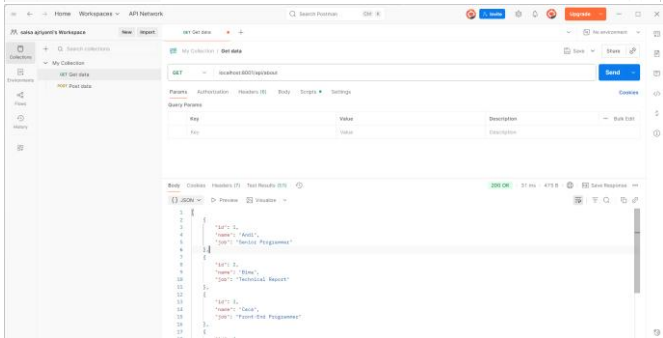
`origin main`

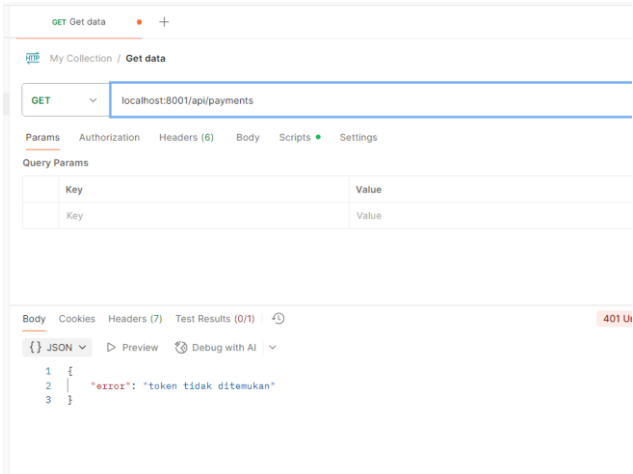
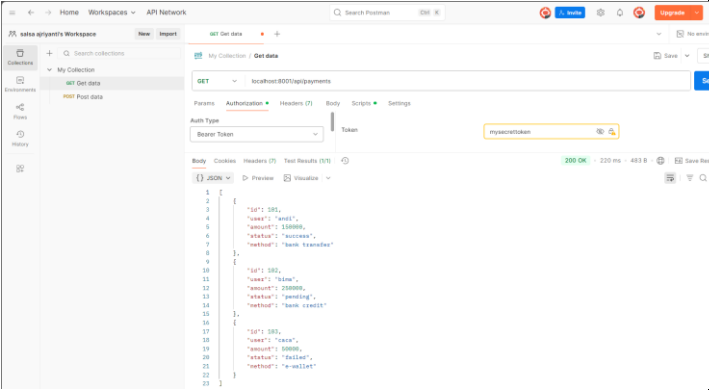
Hasil Pengerjaan

No.	Instruksi	Screenshot	Kendala/Saran
A.	Membuat Server API dengan Express.js		

1.	Membuat folder APIproject1 dan melakukan inialisasi node js dengan mengetik perintah npm init -y	<pre> C:\Users\LENOVO IDEAPAD SLIM3\KODING>cd APIproject1 C:\Users\LENOVO IDEAPAD SLIM3\KODING\APIproject1>npm init -y Wrote to C:\Users\LENOVO IDEAPAD SLIM3\KODING\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", "type": "commonjs" } </pre>	
2.	Menginstall express.js	<pre> C:\Users\LENOVO IDEAPAD SLIM3\KODING\APIproject1>npm install express added 68 packages, and audited 69 packages in 3s 16 packages are looking for funding run `npm fund` for details found 0 vulnerabilities C:\Users\LENOVO IDEAPAD SLIM3\KODING\APIproject1>. code '.' is not recognized as an internal or external command, operable program or batch file. </pre>	
B.	Membuat API pertama		
1.	Membuat file server.js dan isi kode sesuai instruksi di atas serta menjalankan aplikasi lewat browser	 <pre> 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 is running on http://localhost:\${PORT}`); 13 }); </pre> <pre> PS C:\Users\LENOVO IDEAPAD SLIM3\KODING\APIproject1> node server.js Server is running on http://localhost:8001 </pre>	
2.	Hasil dari kode server.js		

C.	Menambahkan Routes (Api Endpoints)		
1.	Buat route baru di file server.js dengan menetik kode program	<pre> 11 app.listen(PORT, () => { 12 console.log('Server is running on http://localhost:\${PORT}'); 13 }); 14 15 app.get('/api/about', (req, res) => { 16 res.json([17 { id:1, name: 'Andi', job: 'Senior Programmer'}, 18 { id:2, name: 'Bima', job: 'Technical Report'}, 19 { id:3, name: 'Caca', job: 'Front-End Programmer'}, 20 { id:4, name: 'Deli', job: 'UI/UX Designer'}, 21 { id:5, name: 'Erlang', job: 'Marketing'} 22]); 23 }); </pre>	
2.	Hasil dari routes yang baru dibuat dengan Alamat /api/about		
D.	Menambahkan Bearer token pada API (Keamanan level 1)		
1.	buat konstanta baru untuk TOKEN dan fungsi baru authBearer	<pre> JS server.js > app.get('/api/payments') callback 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 </pre>	

2.	Buat API Endpoints baru dengan menggunakan authBearer	<pre>46 app.get('/api/payments', authBearer, (req, res) => { 47 res.json([48 { id: 101, user: 'andi', amount: 150000, status: 'success', method: 'bank transfer'}, 49 { id: 102, user: 'bima', amount: 250000, status: 'pending', method: 'bank credit'}, 50 { id: 103, user: 'caca', amount: 50000, status: 'failed', method: 'e-wallet'} 51]); 52 });</pre>	
D.	Membuka API dengan POSTMAN		
1.	Download aplikasi postman dan masuk dengan akun google	 The image shows the Postman welcome screen. At the top is the Postman logo (an orange circle with a white rocket). Below it is a white card with a line-art illustration of an astronaut. The text on the card reads: "It's great to have you aboard, salsa ajriyanti", "Redirecting you to the Desktop App", and "If you aren't redirected automatically, use authorization token to sign in ."	
2.	Buka tab baru, lalu menuju GET localhost:8001 lalu tekan tombol Send , begitu juga dengan endpoints /api/about dan /api/payments	 The image shows the Postman interface with a new tab. The URL bar is set to "GET localhost:8001". The "Send" button is highlighted. Below the URL bar, the "Query Params" section is empty. The "Body" tab is selected, showing a "Raw" JSON response: {"hello": "world!"}.  The image shows the Postman interface with a new tab. The URL bar is set to "GET localhost:8001/api/about". The "Send" button is highlighted. Below the URL bar, the "Query Params" section is empty. The "Body" tab is selected, showing a "Raw" JSON response: {"name": "Salsa Ajriyanti", "email": "salsa@ajriyanti.com", "phone": "08123456789", "address": "Jl. Raya No. 123, Jakarta", "bio": "Software Engineer", "skills": ["JavaScript", "React", "Node.js", "MongoDB"], "social": {"github": "sajriyanti", "linkedin": "sajriyanti"}, "timestamp": "2023-10-27T10:30:00Z"}.	

			
3.	yg terjadi Adalah sama seperti saat kita membuka routes localhost di browser		
4.	<p>Untuk membuka endpoints api/payments maka masukan bearer token sesuai dengan kode program TOKEN kalian.</p> <p>Pada tab Authorization pilih Auth Type menjadi bearer token, lalu isi token dengan “mysecrettoken”</p>		<p>jadi intinya disuruh akses API api/payments dengan cara menambahkan bearer token yang sesuai sama kode TOKEN di program</p>
E.	Github + Visual Code		

1. Buat proyek di Github

