Praktikum 8 - 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 tunjukan hasil akhir dari men-share repository github yang telah dibuat.

A. Membuat Server API dengen Expess.js

1. Buat sebuah folder proyek API dengan nama **APIproject8**
2. Lakukan seperti pada praktikum 3

   Ketik: npm init -y , setelah itu npm install express 3.

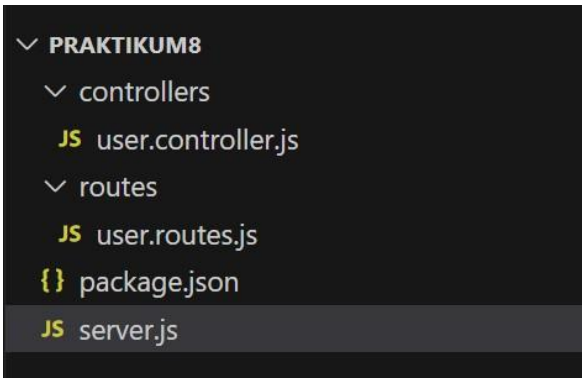
Buat file server.js

```
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
```

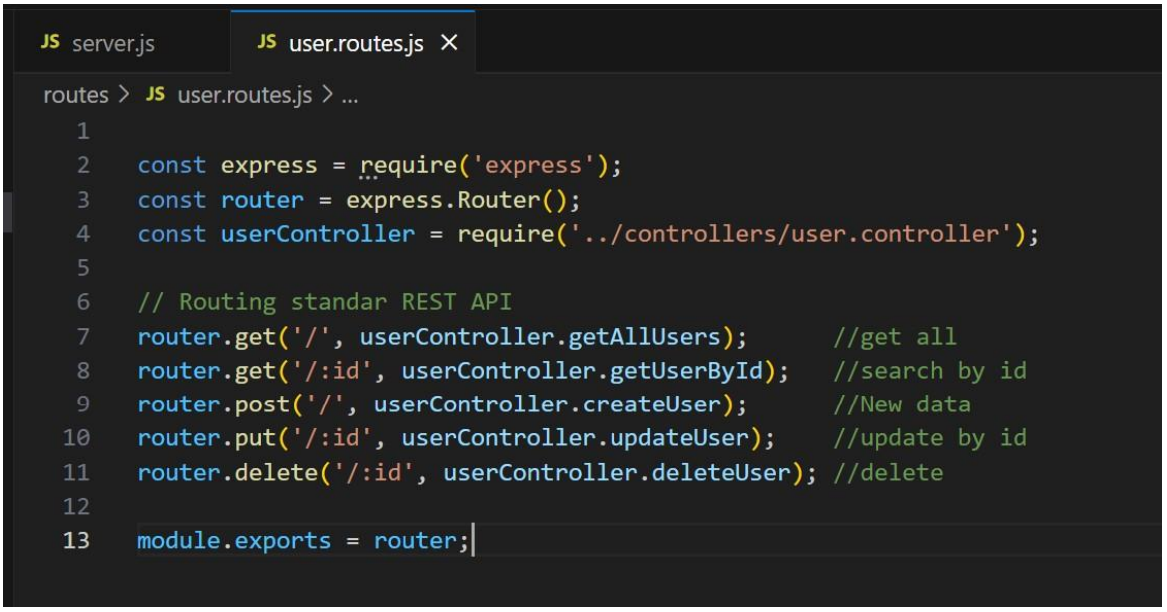
4. Jalankan server.js dengan mengetik Ketik:
   node server.js

B. Membuat Struktur MVC (Routes-Controller) 1. Buat folder **routes, controllers** dan **models**

2. Kemudian didalam folder routes buat sebuah file dengan nama user.routes.js



3. Tulis kode program di file user.routes.js seperti pada gambar dibawah ini



4. Buat file di dalam folder controllers dengan nama user.controller.js
5. Tulis kode program di dalam file user.controller.js seperti pada gambar dibawah ini

Karena pada controller user tersebut require model bernama User, maka kita siapkan Model user, yang berkaitan dengan database.

6. Update file server.js dengan menambahkan kode berikut

```
8   // Routes
9   const userRoutes = require('./routes/user.routes');
10  app.use('/api/users', userRoutes);
```

Kode diatas pada file server.js untuk memberitahu ada routes bernama userRoutes dengan lokasi file di routes/user.routes (tidak perlu ditulis .js)

C. Membuat koneksi Database dengan Models

1. Nyalakan mysql service dan buatlah sebuah database dengan nama dbpraktikum8

```
CREATE DATABASE IF NOT EXISTS dbpraktikum8; CREATE TABLE IF NOT EXISTS users (
id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(100) NOT NULL, email
VARCHAR(100) NOT NULL UNIQUE, password VARCHAR(255) DEFAULT NULL,
created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP, updated_at TIMESTAMP
DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP);
```

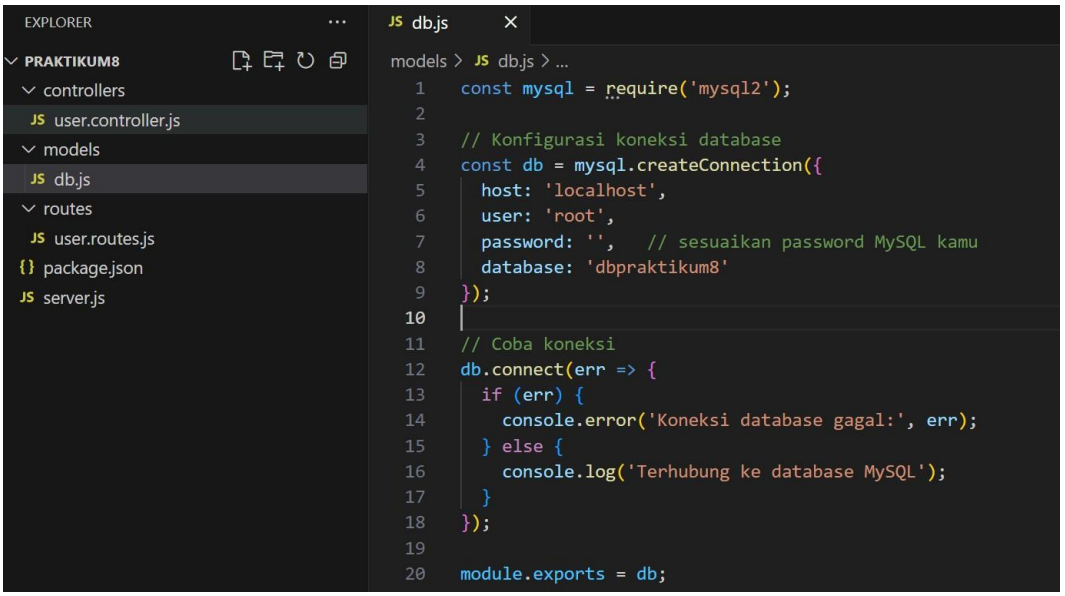
2. Lalu masukan data dummy ke dalamnya

```
INSERT INTO users (name, email, password) VALUES
('Riska Safitri', 'riska@mail.com', '123456'),
('Josephine', 'josep@mail.com', 'abcdef'),
('Moh. Ilham', 'ilham@mail.com', 'qwerty');
```

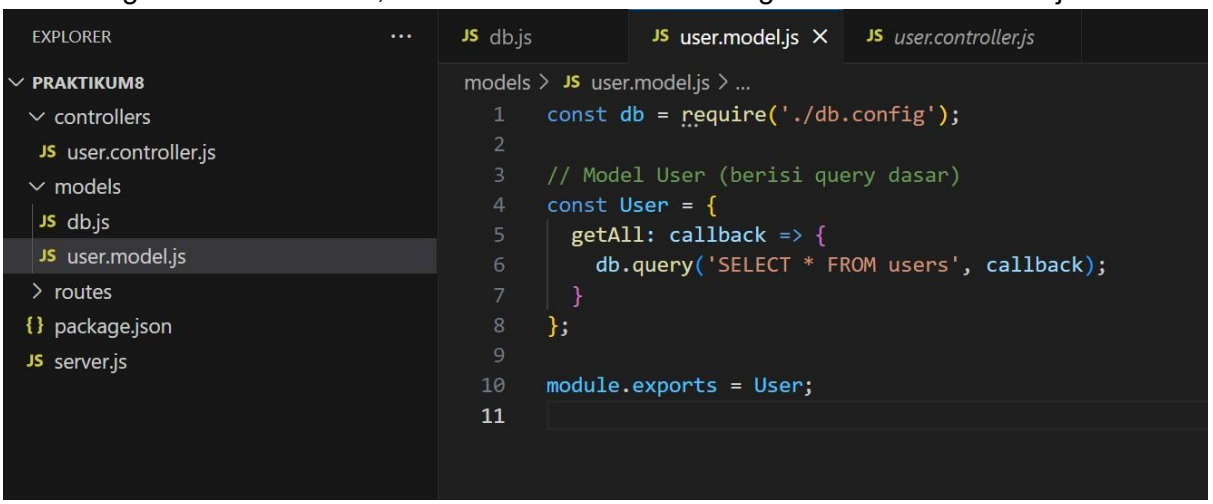
3. Jika database sudah terisi data di tabel users, lalu kita persiapkan kembali di express.js

4. Install Module mysql2 dengan menggunakan node. Masih di folder project ketik perintah berikut: npm install express mysql2

5. Kemudian buat sebuah file di dalam folder models, dengan nama db.config.js dan ketikan seperti berikut



6. File db.config.js adalah sebagai class connector antara express dan database
7. Buat file lagi untuk model user, di dalam folder models. Dengan nama user.model.js



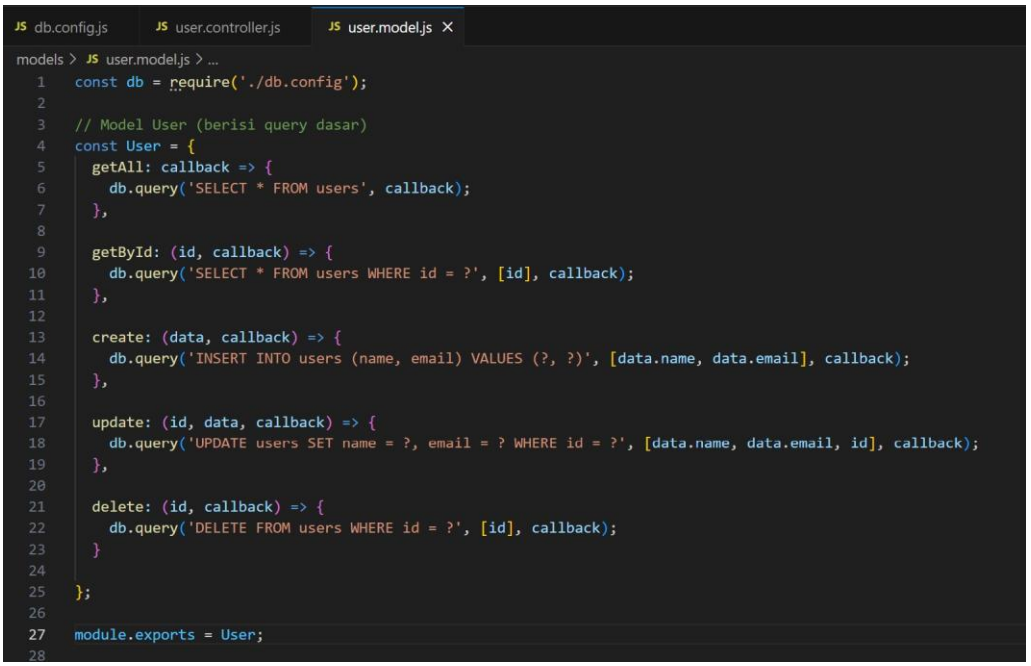
8. Jalankan atau restart ulang node server.js
   (Pastikan mysql sudah running, user password mysql sudah benar)

C. Melakukan Test API

Gunakan browser/postman untuk mendapatkan data getAll users dengan mengunjungi endpoints /api/users/

D. Lengkapi Controllers dan Model

1. Tambahkan class untuk model baru, agar terhubung dengan controller. Ubah pada file user.model.js



2. Tambahkan class baru untuk routes yang sudah dipersiapkan lainnya, bisa dilihat pada kode program dibawah ini

File: user.controller.js

```
// GET user by ID
exports.getUserById = (req, res) => {
  const { id } = req.params;
  User.getById(id, (err, results) => {
    if (err) return res.status(500).json({ error: err.message });
    if (results.length === 0) return res.status(404).json({ message: 'User tidak ditemukan' });
    res.json(results[0]);
  });
};

// POST user baru
exports.createUser = (req, res) => {
  const data = req.body;
  User.create(data, (err, result) => {
    if (err) return res.status(500).json({ error: err.message });
    res.status(201).json({ id: result.insertId, ...data });
  });
};

// PUT update user
exports.updateUser = (req, res) => {
  const { id } = req.params;
  const data = req.body;
  User.update(id, data, (err, result) => {
    if (err) return res.status(500).json({ error: err.message });
    if (result.affectedRows === 0) return res.status(404).json({ message: 'User tidak ditemukan' });
    res.json({ message: 'User berhasil diupdate' });
  });
};

// DELETE user
exports.deleteUser = (req, res) => {
  const { id } = req.params;
  User.delete(id, (err, result) => {
    if (err) return res.status(500).json({ error: err.message });
    if (result.affectedRows === 0) return res.status(404).json({ message: 'User tidak ditemukan' });
    res.json({ message: 'User berhasil dihapus' });
  });
};
```

E. Melakukan Test API secara Lengkap

Dengan menggunakan POSTMAN, lakukan pengujian berikut:

1. Menguji endpoint /
2. Menguji endpoint /api/users (Method: GET)
3. Menguji endpoint /api/users/1 (Method: GET)
4. Menguji endpoint /api/users (Method: POST)
   Tambah body -> raw -> JSON
   {
   "name": "Budi Santoso",
   "email": "budi@example.com"
   }

5. Menguji /api/users/2 (Method: PUT)
   Masukan Body -> raw -> JSON

```
{
  "name": "Joe Taslim",
  "email": "jojo@example.com"
}
```

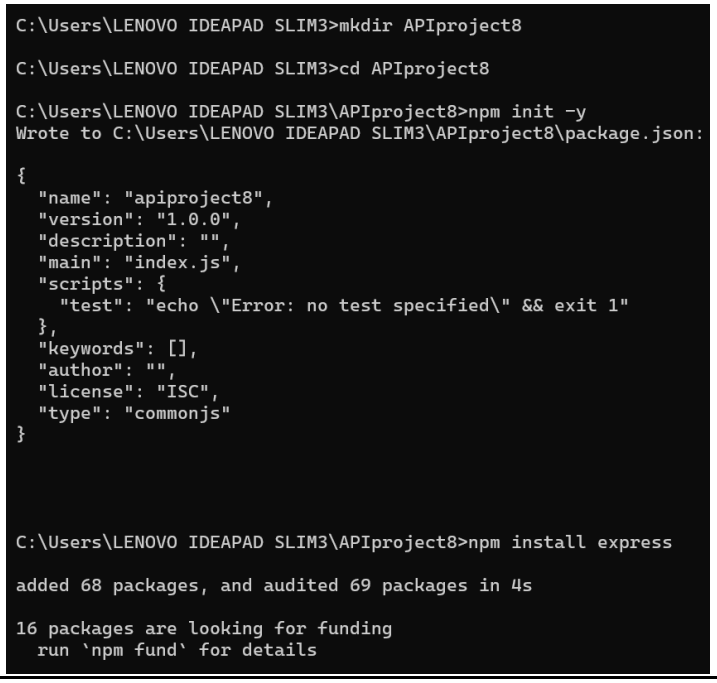
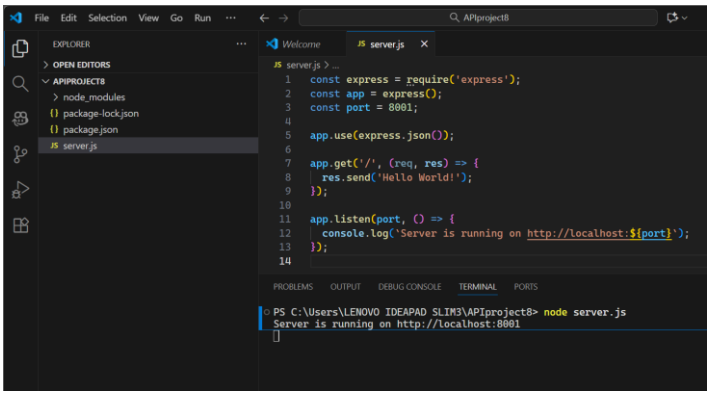
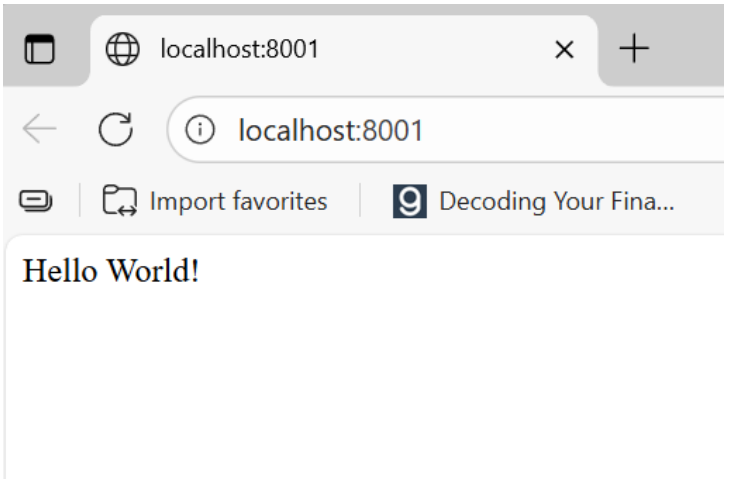
6. Menguji /api/users/3 (Method: DELETE)

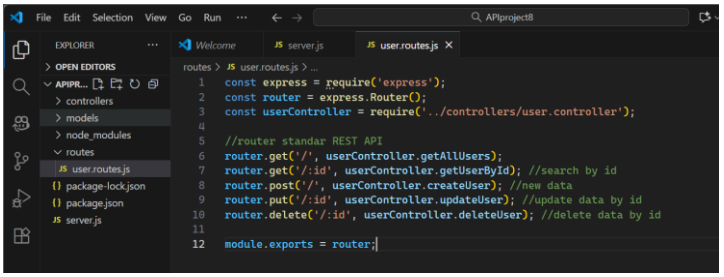
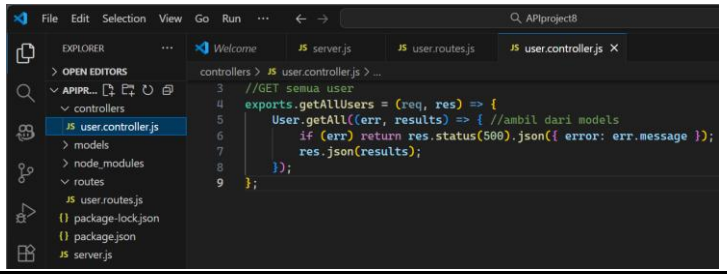
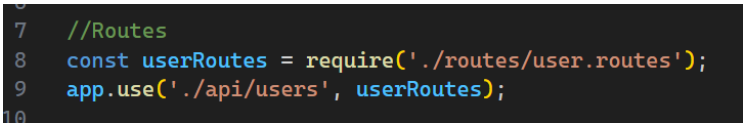
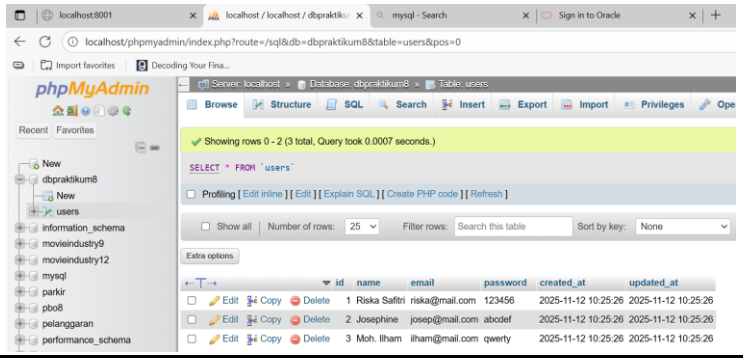
F. Github + Visual Code

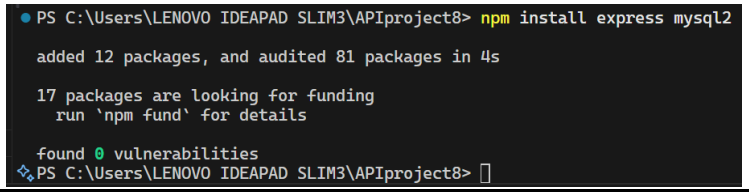
1. Buat proyek di Github dengan nama **Latihan8**

```
git init
git add
.
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/agunghakase/Latihan8.git
git push -u origin main
```

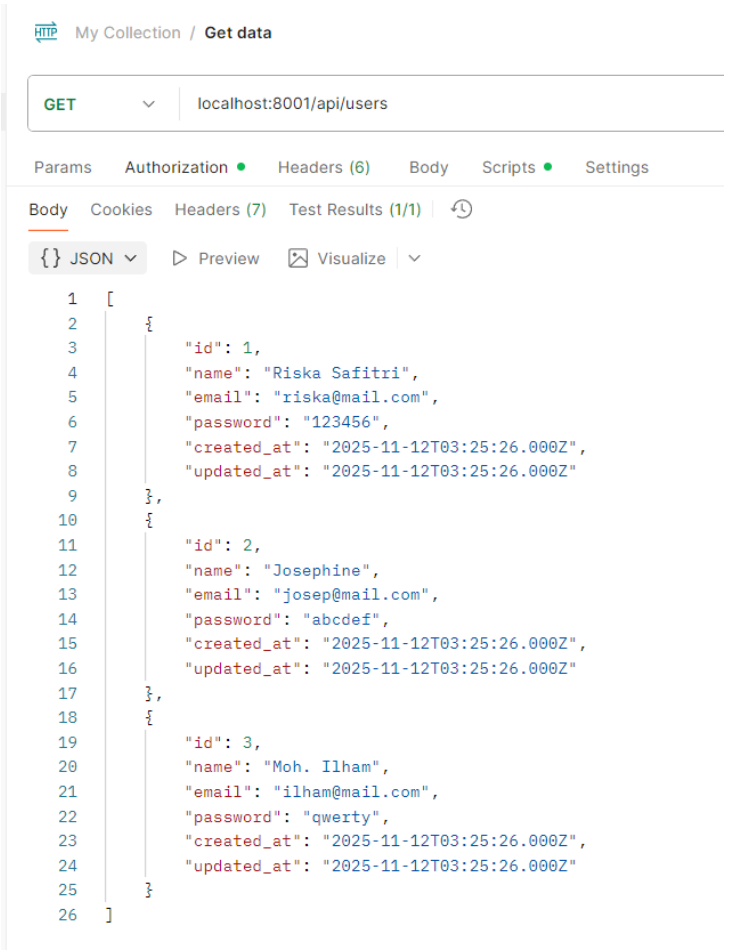
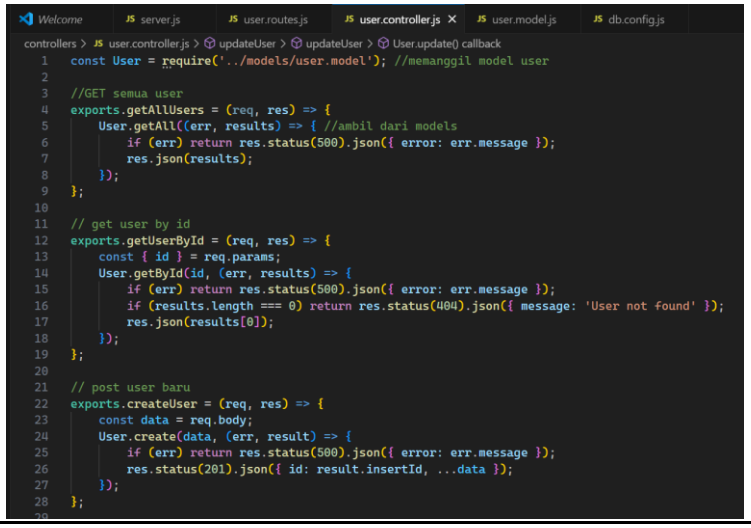
Hasil Pengerjaan

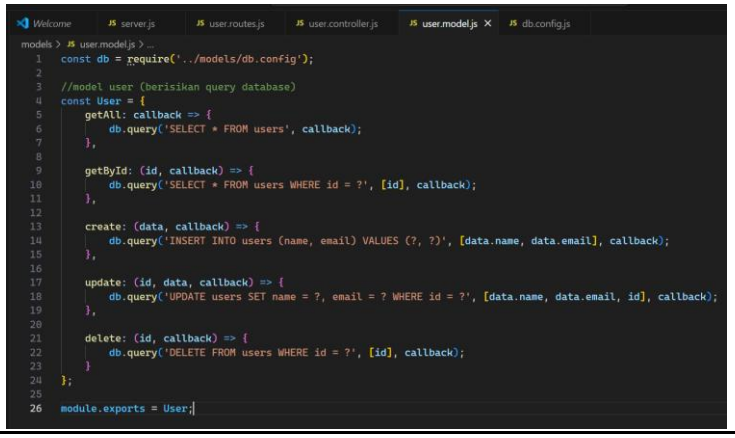
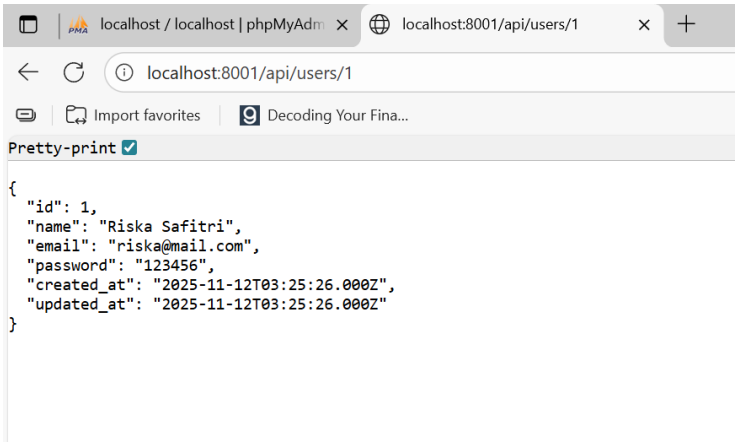
No.	Instruksi	Screenshot	Kendala/Saran
A.	Membuat server api dengan express js		
1.	Membuat sebuah folder proyek dengan nama APIproject8, ketik npm init -y dan npm install express		
2.	Buat file server.js serta isi kode sesuai dengan perintah dan Jalankan server.js dengan mengetik node server.js		
3.	Output dari node server.js		

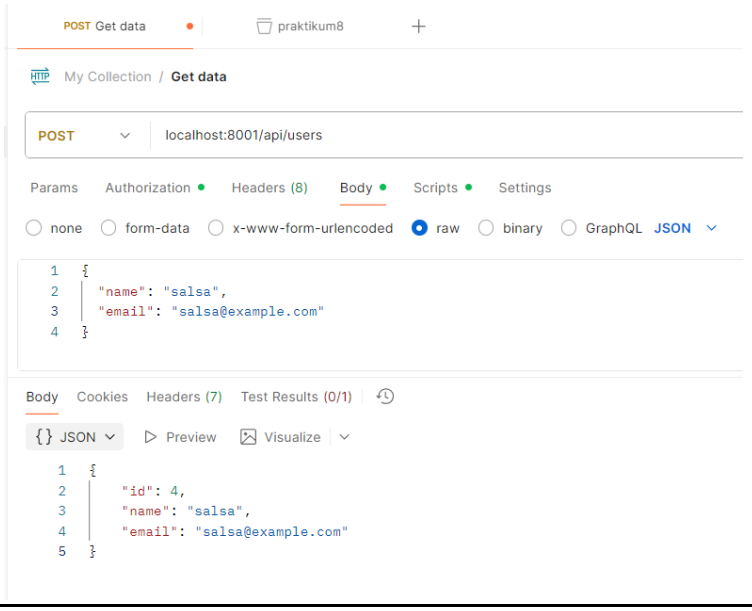
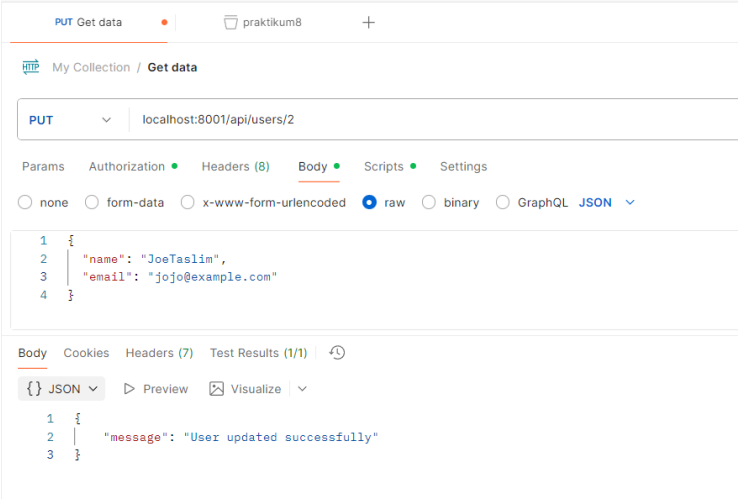
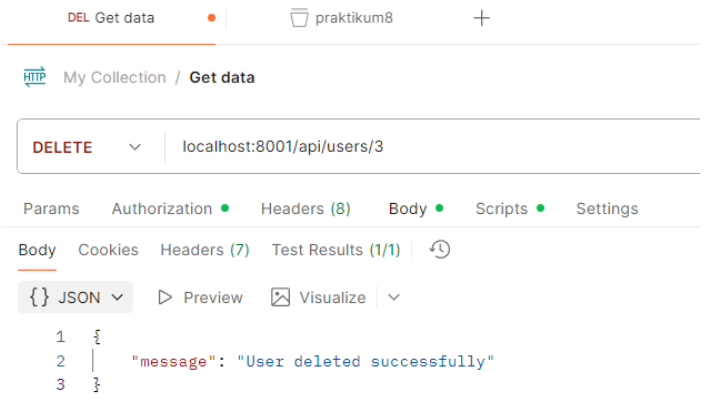
B.	Membuat struktur mvc (routes-controller)		
1.	Buat folder routes, controllers dan models kemudian di dalam folder routes buat sebuah file dengan nama user.routes.js dan tuliskan kode program di file user.routes.js		
2.	Buat file didalam folder controllers dengan nama user.controllers.js dan tuliskan kode programnya		
3.	Update file server.js dengan menambahkan kode berikut. Kode tersebut untuk memberitahu ada routes Bernama userRoutes dengan Lokasi file di routes/user.routes		
C.	Membuat koneksi database dengan models		
1.	Nyalakan mysql service dan buatlah database dengan nama dbpraktikum8 dan masukkan data dummy ke dalamnya		

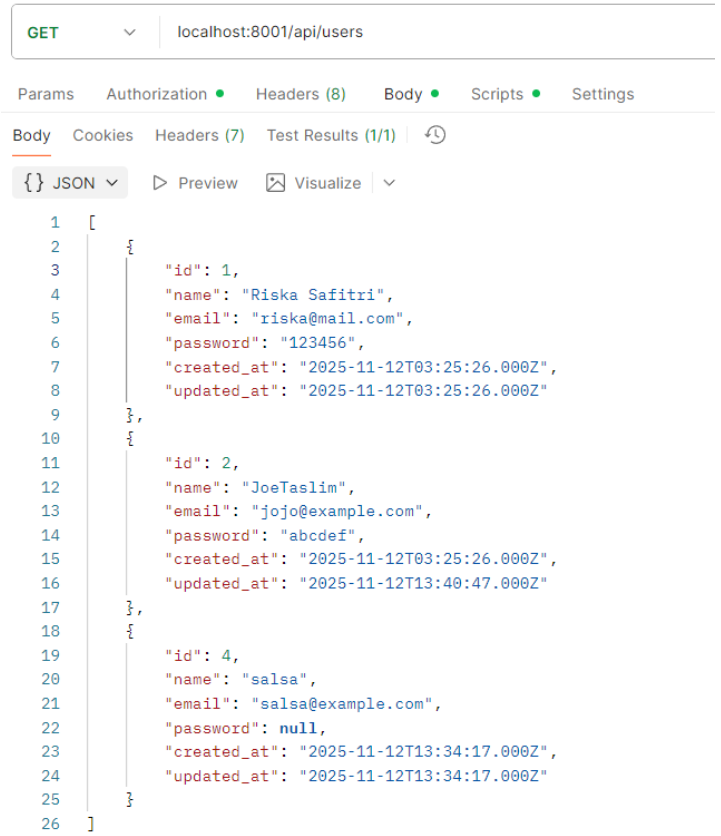
2.	Database sudah terisi, selanjutnya npm install express mysql2		
3.	Buat file db.config.js di folder model dan isikan kode seperti perintah		
4.	Buat file user.model.js di folder model. Jalankan dan restart ulang node		

5.	Setelah di jalankan ulang		
D.	Melakukan test API		

1.	Gunakan browser/postman untuk mendapatkan data getAll users dengan mengunjungi endpoints /api/users/		
E.	Lengkapi controllers dan model		
1.	Lengkapi kode yg ada di controllers		

		 	
2.	Lengkapi kode y gada di model	 	
3.	Output get by id	 	

4.	Output tambah data baru		
5.	Output update data		
6.	Output hapus data		

7.	Output akhir		
F.	Commit github		
1.	Git init Git add . Git commit -m “first commit” Git branch -M main Git remote Git add origin Git push -u origin main