Praktikum 10 - 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 JSON Web Token (Dynamic Bearer Token)

1. Lanjutkan Project Praktikum 8-9, dengan menggunakan file yang sama (copy)
2. Install library JWT **npm install jsonwebtoken bcryptjs**

3. Tambahkan file auth.controller.js, auth.middleware.js, dan auth.routes.js
4. Buat file .env disamping server.js (root folder) Isi file .env dengan variable sebagai berikut:
   JWT_SECRET=”KUNCI-RAHASIA”
   JWT_EXPIRE=1d

5. Tambahkan script berikut di server.js `require('dotenv').config();`

6. Revisi model sebelumnya pada user.model.js dengan menambahkan fungsi baru seperti berikut, tambahkan findByEmail

```
    delete: (id, callback) => {
        db.query('DELETE FROM users WHERE id = ?', [id], callback);
    },

    // Get user by Email (untuk login)
    findByEmail: (email, callback) => {
        db.query('SELECT * FROM users WHERE email = ?', [email], callback);
    },

};
```

7. Masukan script berikut pada auth.controller.js yang telah dibuat

```
auth.controller.js U
controllers > auth.controller.js > login > login > User.findByEmail() callback
const User = require('../models/user.model');
const bcrypt = require('bcryptjs');
const jwt = require('jsonwebtoken');

exports.login = (req, res) => {
    const { email, password } = req.body;

    User.findByEmail(email, (err, results) => {
        if (err) return res.status(500).json({ message: err.message });
        if (results.length === 0) return res.status(404).json({ message: "User not found" });

        const user = results[0];

        const match = bcrypt.compareSync(password, user.password);
        if (!match) return res.status(400).json({ message: "Wrong password" });

        const token = jwt.sign(
            { id: user.id, email: user.email },
            process.env.JWT_SECRET,
            { expiresIn: "7d" }
        );

        res.json({
            message: "Login success",
            token,
            user: { id: user.id, name: user.name, email: user.email }
        });
    });
};
```

8. Ubah auth.middleware.js yang sebelumnya menggunakan token biasa, menjadi json web token seperti gambar dibawah ini

```
h.controller.js U | user.model.js M | auth.middlewares.js M
ewares > auth.middlewares.js > ...
const jwt = require("jsonwebtoken");
const User = require("../models/user.model");

module.exports = (req, res, next) => {
    const header = req.headers.authorization;

    if (!header || !header.startsWith("Bearer ")) {
        return res.status(401).json({ message: "Unauthorized" });
    }

    const token = header.split(" ")[1];

    try {
        const decoded = jwt.verify(token, process.env.JWT_SECRET);

        // Optional: cek user masih ada
        User.getById(decoded.id, (err, results) => {
            if (err) return res.status(500).json({ message: err.message });
            if (results.length === 0) {
                return res.status(401).json({ message: "Invalid token user" });
            }

            req.user = results[0];
            next();
        });

    } catch (err) {
        return res.status(401).json({ message: "Invalid token" });
    }
};
```

9. Tambahkan Routes untuk mengakses login pada auth.routes.js

10. Pada server.js tambahkan kode berikut untuk menambahkan routes

```
const authRoutes = require("./routes/auth.routes");
app.use('/api/auth', authRoutes);
```

B. SETUP DATABASE

1. Pada tabel user seharusnya password di enkripsi dengan bcryptjs. Oleh karena itu kita buat hasil enkripsinya dengan membuat tools node js seperti berikut
2. Buat file createpwd.js dan tulis kode seperti dibawah ini

```
const bcrypt = require('bcryptjs');
const hash = bcrypt.hashSync('passwordkamu', 10);
console.log(hash);
```

3. Ganti **passwordkamu** dengan password yang diinginkan, lalu ketik perintah node

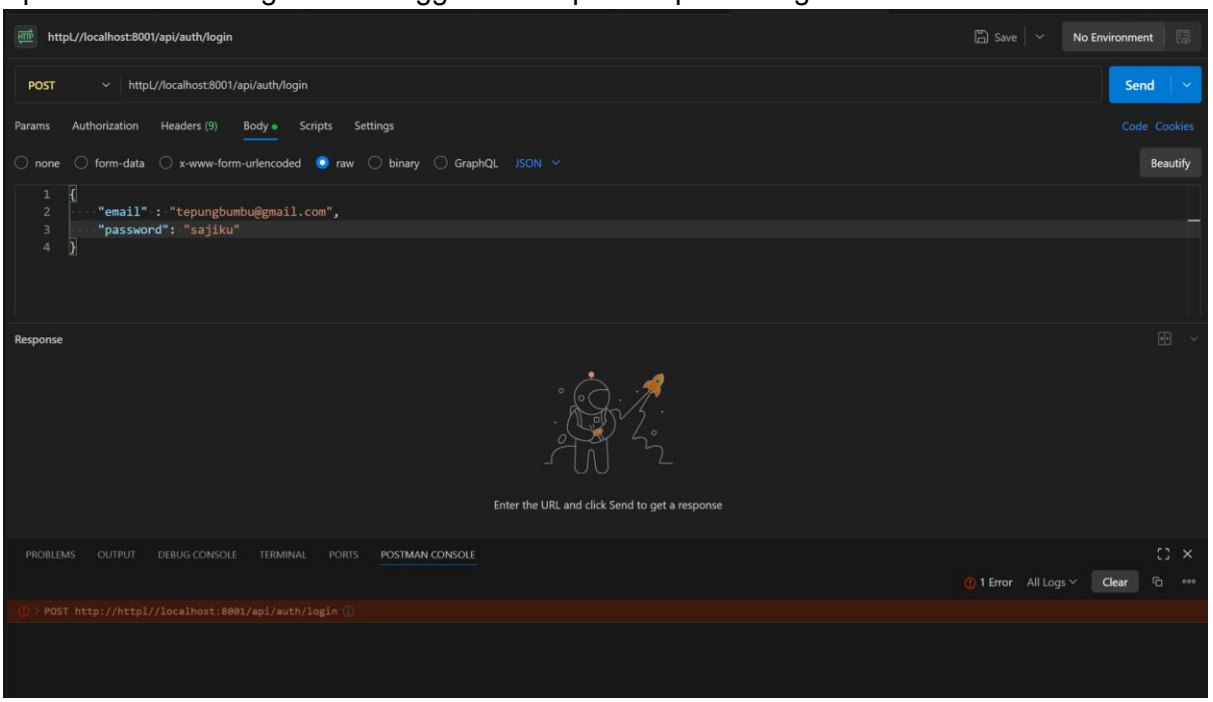
```
PS D:\Node\Praktikum8> node .\createpwd.js
$2b$10$UtTsJEfWY0JfqNOHTI859uf2QADWr9ry4g8w/t3/1fyZNZvwjOnC2
```

4. Ganti isi field password dengan hasil dari hash menggunakan bcryptjs.

   Note: Ini adalah simulasi, kedepannya untuk create user dan password seharusnya menggunakan bcryptsjs agar lebih aman.

B. Gunakan POSTMAN dapatkan Token BEARER

1. Install postman di visual code, dan lakukan login berdasarkan email dan password yang terdaftar di database
2. Dapatkan bearer dengan memanggil API endpoints api/auth/login

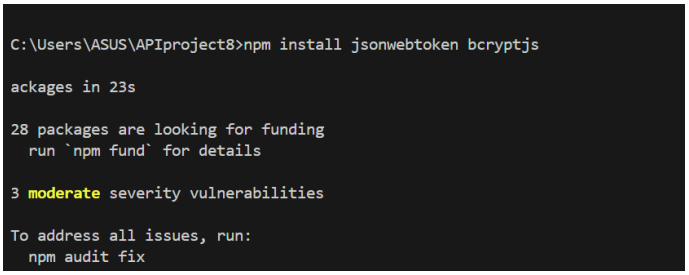
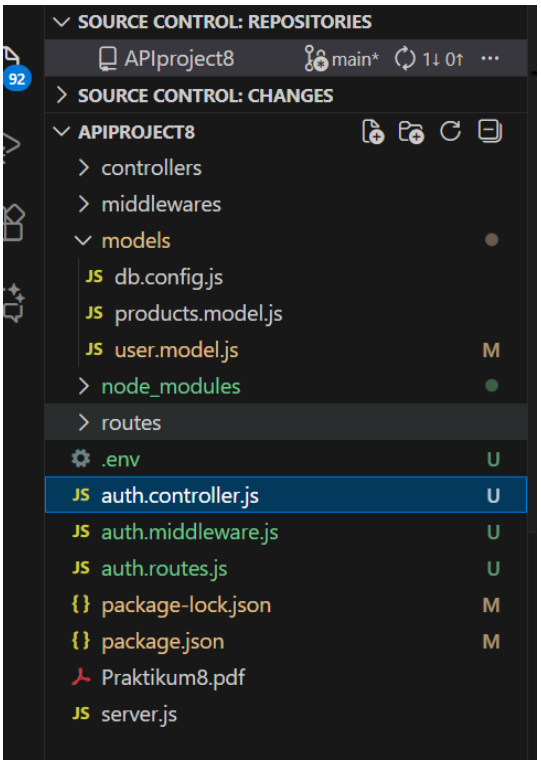
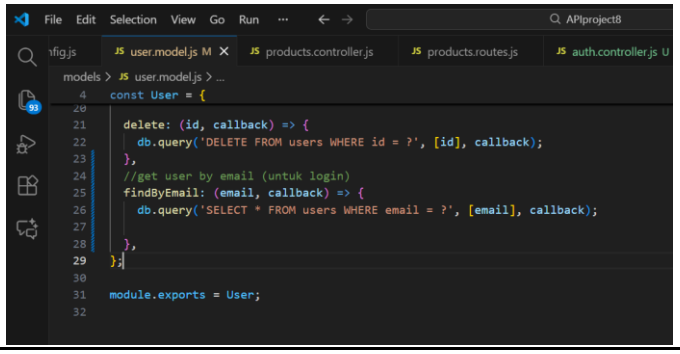


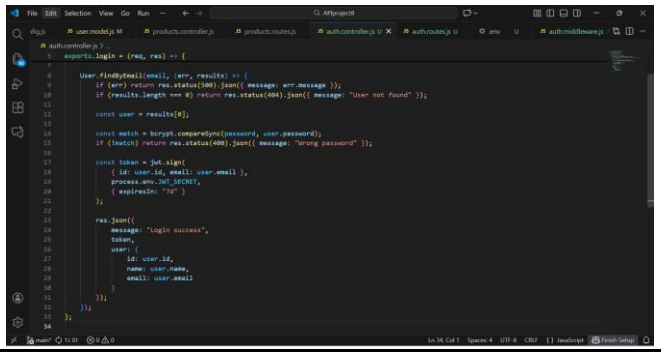
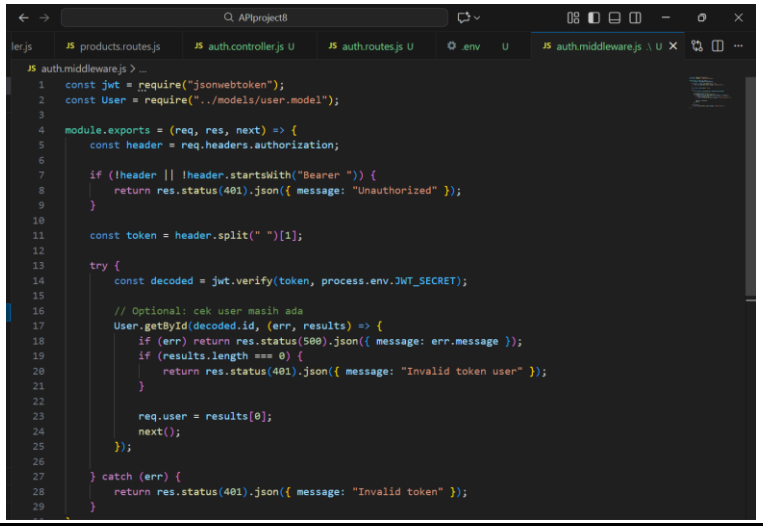
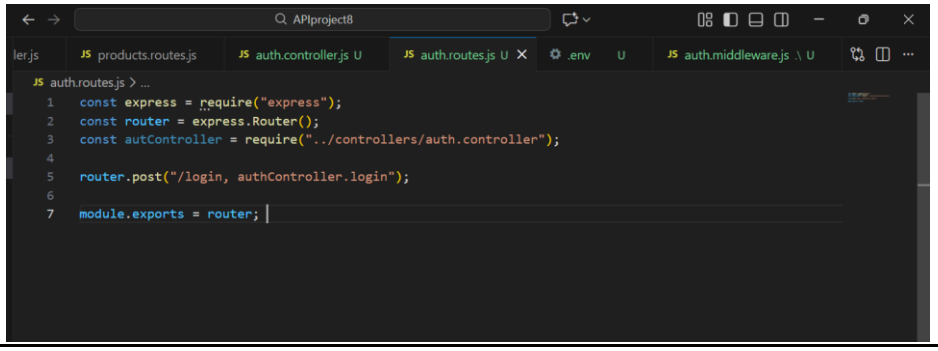
3. Catat bearer yang di dapatkan, lalu gunakan bearer tersebut untuk memanggil endpoints lainya yang pada praktikum 9 telah di proteksi.
4. Token yang di dapat dari login, bisa digunakan untuk mengakses semua API yang diproteksi dalam sistem.

F. Github + Visual Code

1. Buat proyek di Github dengan nama **Latihan10**
   git init
   git add
   .
   git commit -m "first commit" git branch -M main git remote add origin https://github.com/**agunghakase**/**Latihan9.git** git push -u origin main

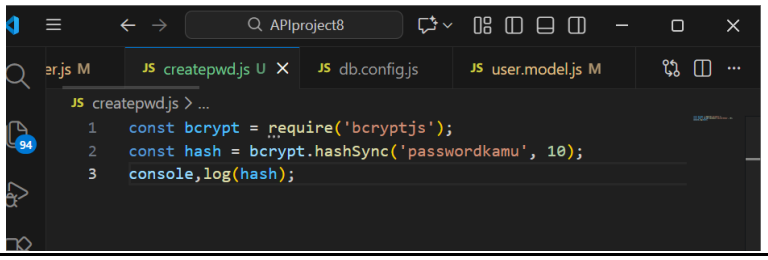
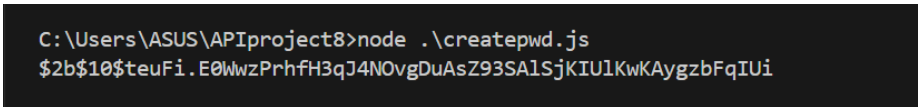
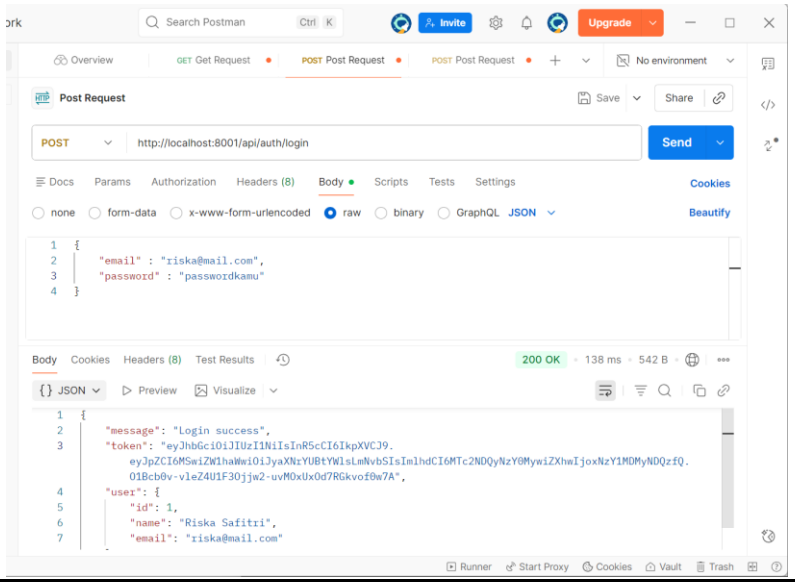
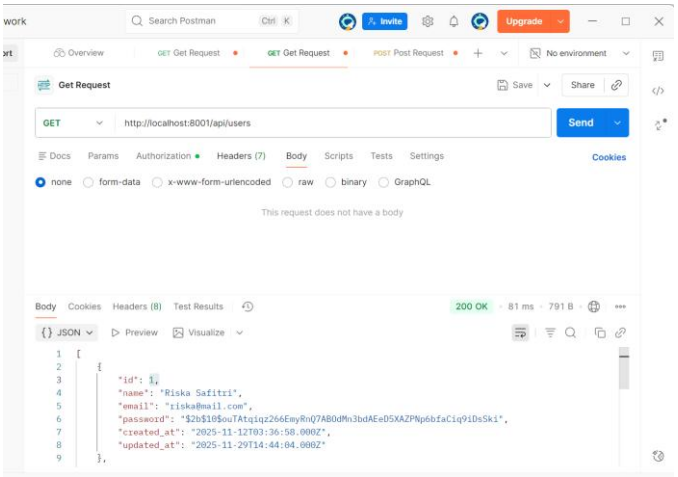
Hasil Pengerjaan

No.	Instruksi	Screenshot
A.	Installasi dan Konfigurasi	
1.	Install library JWT npm install jsonwebtoken bcryptjs	
2.	Tambahkan file auth.controller.js, auth.middleware.js, dan auth.routes.js	
3.	mengisi user.model.js	

	Mengisi auth.controller.js	
	Mengisi auth.middleware.js	
	Mengisi auth.routes.js	
	Menambahkan require dotenv ke server.js	
B.	Github dan Viscode	

```
require('dotenv').config();

const authRoutes = require("./routes/auth.routes");
app.use('/api/auth', authRoutes);
```

1.	Membuat file createpwd.js dan mengisinya	
2.	Mengganti passwordkamu dengan hashpassword	
3.	Memanggil API dengan endpoints api/auth/login	
4.	Memanggil endpoint laiin	
5.	Link github	https://github.com/ftrnaa/latihan-8.git