Nama : Muhammad Rizaldy Akbar Kelas : SE063

NIM : 2211104065

Server.js:

const express = require('express'); const bcrypt = require('bcryptjs');

const bodyParser = require('body-parser'); const fs = require('fs');

const app = express(); app.use(bodyParser.json());

// Fungsi untuk validasi input

function validateInput(username, password) {

// Validasi panjang username dan password

if (username.length < 5 || username.length > 20) { return 'Username harus antara 5 dan 20 karakter';

}

if (password.length < 8 || password.length > 20) { return 'Password harus antara 8 dan 20 karakter';

}

// Validasi password tidak mengandung username if (password.includes(username)) {

return 'Password tidak boleh mengandung username';

}

return null; // Valid

}

// Fungsi untuk mengecek apakah password memenuhi aturan function passwordStrength(password) {

const regex = /[!@#$%^&\*(),.?":{}|<>]/; // Karakter unik return regex.test(password);

}

// Registrasi user

app.post('/register', (req, res) => {

const { username, password } = req.body;

const error = validateInput(username, password);

if (error) {

return res.status(400).send(error);

}

if (!passwordStrength(password)) {

return res.status(400).send('Password harus mengandung minimal 1 karakter unik');

}

// Hash password menggunakan bcrypt bcrypt.hash(password, 10, (err, hashedPassword) => {

if (err) return res.status(500).send('Error hashing password'); const user = { username, password: hashedPassword };

// Simpan user ke file JSON

fs.readFile('./data/users.json', (err, data) => {

if (err) return res.status(500).send('Error reading users data'); const users = JSON.parse(data);

users.push(user);

fs.writeFile('./data/users.json', JSON.stringify(users, null, 2), (err) => { if (err) return res.status(500).send('Error saving user data'); res.status(201).send('User registered successfully');

});

});

});

});

// Login user

app.post('/login', (req, res) => {

const { username, password } = req.body;

fs.readFile('./data/users.json', (err, data) => {

if (err) return res.status(500).send('Error reading users data'); const users = JSON.parse(data);

const user = users.find(user => user.username === username); if (!user) {

return res.status(400).send('User not found');

}

// Verifikasi password

bcrypt.compare(password, user.password, (err, result) => {

if (err) return res.status(500).send('Error comparing password'); if (!result) {

return res.status(400).send('Incorrect password');

}

res.status(200).send('Login successful');

});

});

});

// Menjalankan server const port = 3000; app.listen(port, () => {

console.log(`Server running on [http://localhost:${](http://localhost/)port}`);

});

Penjelasan kode:

Kode di atas adalah implementasi sederhana server autentikasi menggunakan Node.js dan Express. Aplikasi ini menyediakan dua endpoint utama: /register untuk registrasi pengguna dan

/login untuk proses login. Saat pengguna mendaftar, input username dan password akan divalidasi terlebih dahulu, seperti memastikan panjang karakter yang sesuai, memastikan

password tidak mengandung username, serta wajib mengandung minimal satu karakter unik.

Jika valid, password kemudian di-*hash* menggunakan bcrypt dan disimpan dalam file users.json.

Saat login, server akan membaca data dari users.json, mencari username yang cocok, lalu

membandingkan password yang dimasukkan dengan password yang telah di-*hash*. Jika cocok, pengguna berhasil login. Sistem ini juga menangani berbagai kemungkinan error seperti kesalahan dalam pembacaan file atau hashing password. Secara keseluruhan, aplikasi ini menunjukkan bagaimana membuat autentikasi dasar dengan validasi input, hashing password, dan penyimpanan data dalam file JSON.

data/users.json

[

{

"username": "john\_doe", "password":

"$2b$10$NdCthFTpoMQAunDpHT2lHuuJUUz2wnGeR/7hR62MSl.uB.WyPe06W"

}

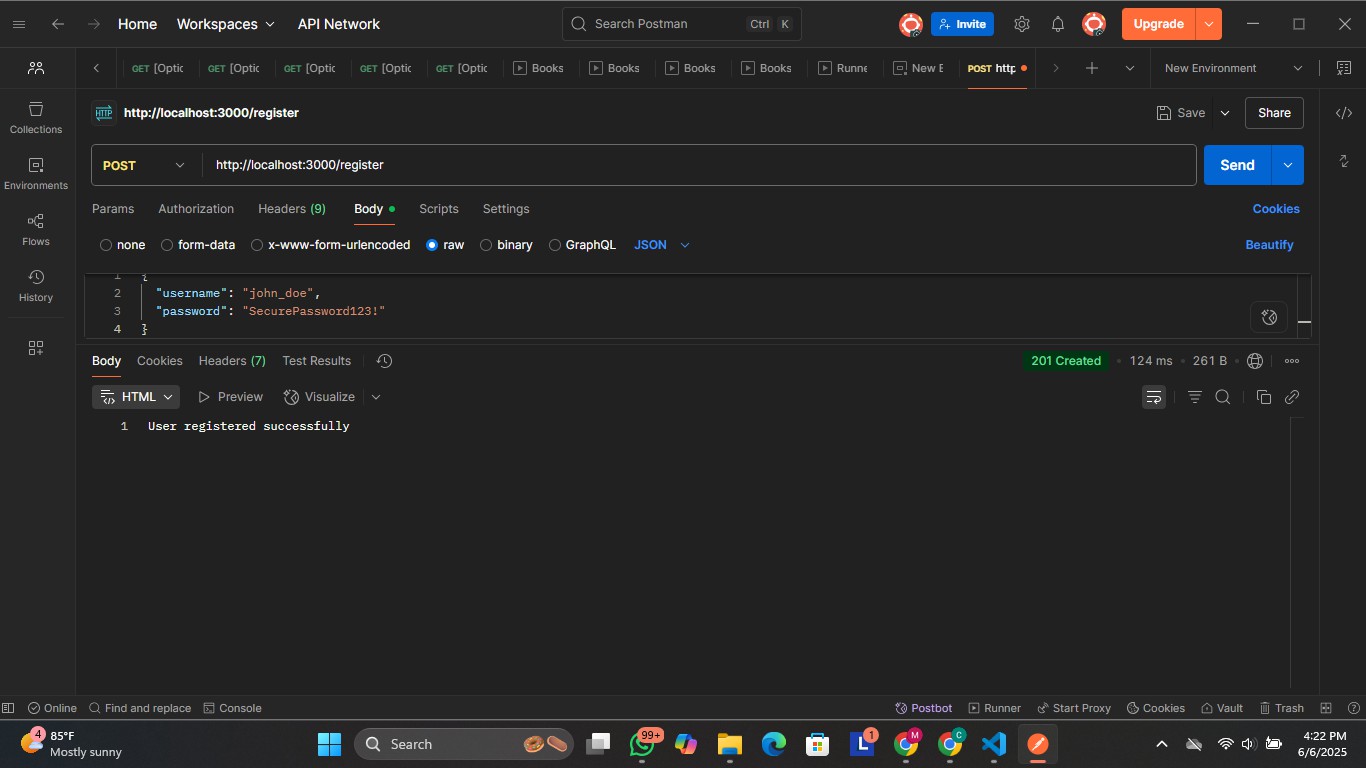
]

Penjelasan kode:

File users.json yang ditampilkan berisi satu objek dengan dua properti: username dan password. username menyimpan nilai "john\_doe", yang merupakan nama pengguna yang terdaftar.

Properti password berisi nilai yang dihasilkan oleh **hashing** menggunakan algoritma bcrypt. Nilai password yang di-hash, yaitu

$2b$10$NdCthFTpoMQAunDpHT2lHuuJUUz2wnGeR/7hR62MSl.uB.WyPe06W, adalah hasil pengolahan dari password asli yang dimasukkan oleh pengguna. Hashing password digunakan untuk meningkatkan keamanan, karena password asli tidak disimpan langsung dalam file, melainkan hanya nilai hash-nya yang tidak dapat dikembalikan ke bentuk aslinya, sehingga mengurangi risiko kebocoran data jika file tersebut diakses oleh pihak yang tidak berwenang.

Uji Register menggunakan POSTMAN

Hasil users.json setelah berhasil register

[

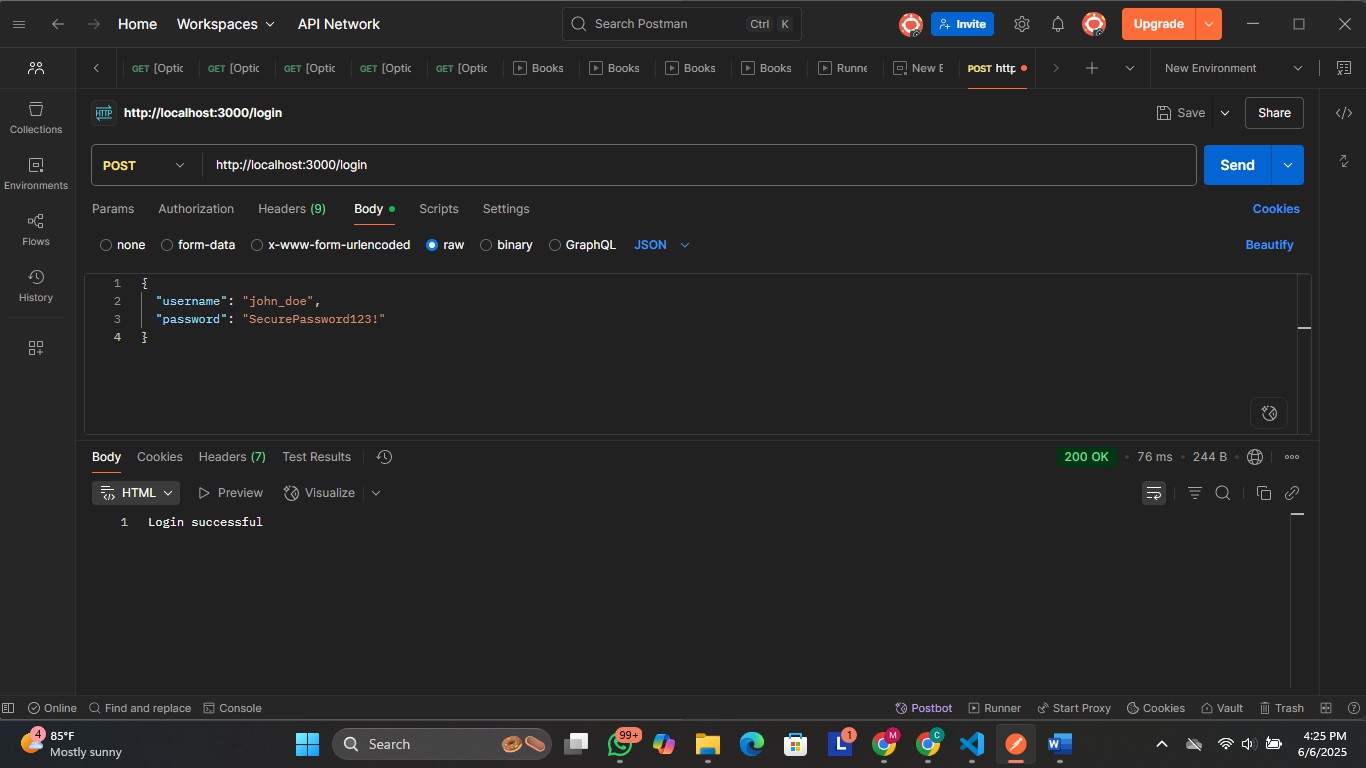
{

"username": "john\_doe", "password":

"$2b$10$NdCthFTpoMQAunDpHT2lHuuJUUz2wnGeR/7hR62MSl.uB.WyPe06W"

}

]

Uji Login menggunakan POSTMAN