Program ini adalah **REST API sederhana untuk autentikasi**:

- Menyimpan user dan password ke file users.json.
- Mengenkripsi password dengan bcrypt untuk keamanan.
- Validasi yang cukup ketat untuk mencegah password lemah dan username tidak valid.
- Belum menggunakan database, cocok untuk pembelajaran dasar.

users.json => menyimpan data username dan password.

server.js

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

const app = express();
const PORT = 3000;

app.use(bodyParser.json());

// Load data user dari file
const loadUsers = () => {
  if (!fs.existsSync('users.json')) return [];
  return JSON.parse(fs.readFileSync('users.json'));
};

// Simpan data user ke file
const saveUsers = (users) => {
  fs.writeFileSync('users.json', JSON.stringify(users, null, 2));
};

// Validasi password
const validatePassword = (password, username) => {
  const hasSpecialChar = /[!@#$%^&*]/.test(password);
  const isLongEnough = password.length >= 8 && password.length <= 20;
  const doesNotIncludeUsername = !password.includes(username);

  return hasSpecialChar && isLongEnough && doesNotIncludeUsername;
};
```

```
// Registrasi user
app.post('/register', async (req, res) => {
  const { username, password } = req.body;

  // Validasi input
  if (!/^[a-zA-Z0-9]+$/.test(username)) {
    return res.status(400).json({ message: 'Username hanya boleh huruf/angka ASCII' });
  }
  if (!validatePassword(password, username)) {
    return res.status(400).json({ message: 'Password harus 8-20 karakter, mengandung simbol, dan tidak boleh mengandung username' });
  }

  const users = loadUsers();
  const userExists = users.find(u => u.username === username);
  if (userExists) {
    return res.status(400).json({ message: 'Username sudah terdaftar' });
  }

  const hashedPassword = await bcrypt.hash(password, 10);
  users.push({ username, password: hashedPassword });
  saveUsers(users);

  res.status(201).json({ message: 'Registrasi berhasil' });
});
```

```
// Login user
app.post('/login', async (req, res) => {
  const { username, password } = req.body;

  const users = loadUsers();
  const user = users.find(u => u.username === username);
  if (!user) {
    return res.status(400).json({ message: 'Username tidak ditemukan' });
  }

  const passwordMatch = await bcrypt.compare(password, user.password);
  if (!passwordMatch) {
    return res.status(400).json({ message: 'Password salah' });
  }

  res.status(200).json({ message: 'Login berhasil' });
});

app.listen(PORT, () => {
  console.log(`Server berjalan di http://localhost:${PORT}`);
});
```

Library yang digunakan:

- express: Framework untuk membuat server HTTP.
- body-parser: Untuk menguraikan request body dalam format JSON.
- fs: Untuk membaca/menulis file JSON (penyimpanan data user).
- bcrypt: Untuk meng-hash password dengan aman (mencegah penyimpanan plaintext password).

Fungsi utama program:

loadUsers()

- Membaca file users.json (jika ada).
- Mengembalikan array berisi daftar user.
- Jika file tidak ada, mengembalikan array kosong ([]).

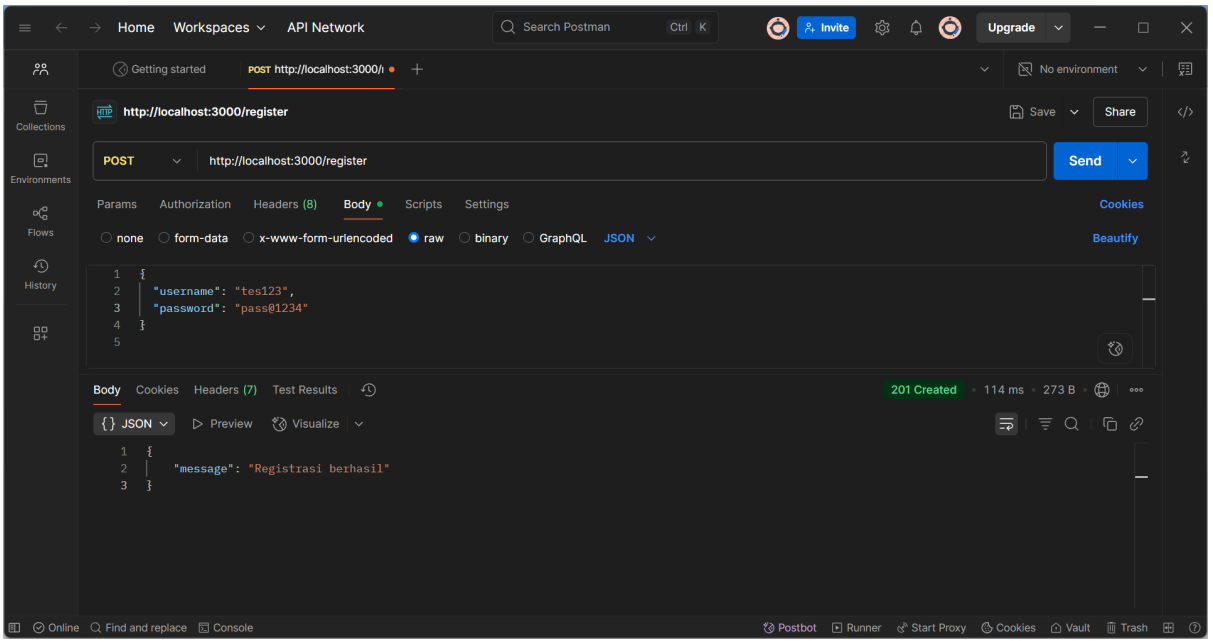
saveUsers(users)

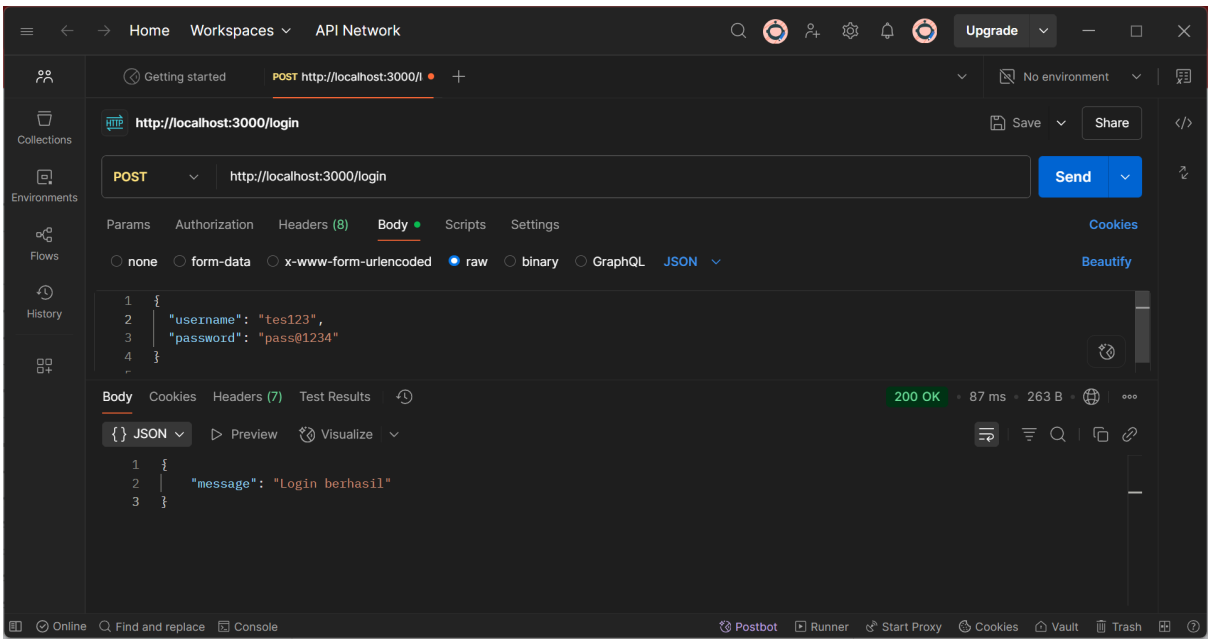
- Menyimpan data user ke file users.json dalam format JSON terstruktur (indent 2).

validatePassword(password, username)

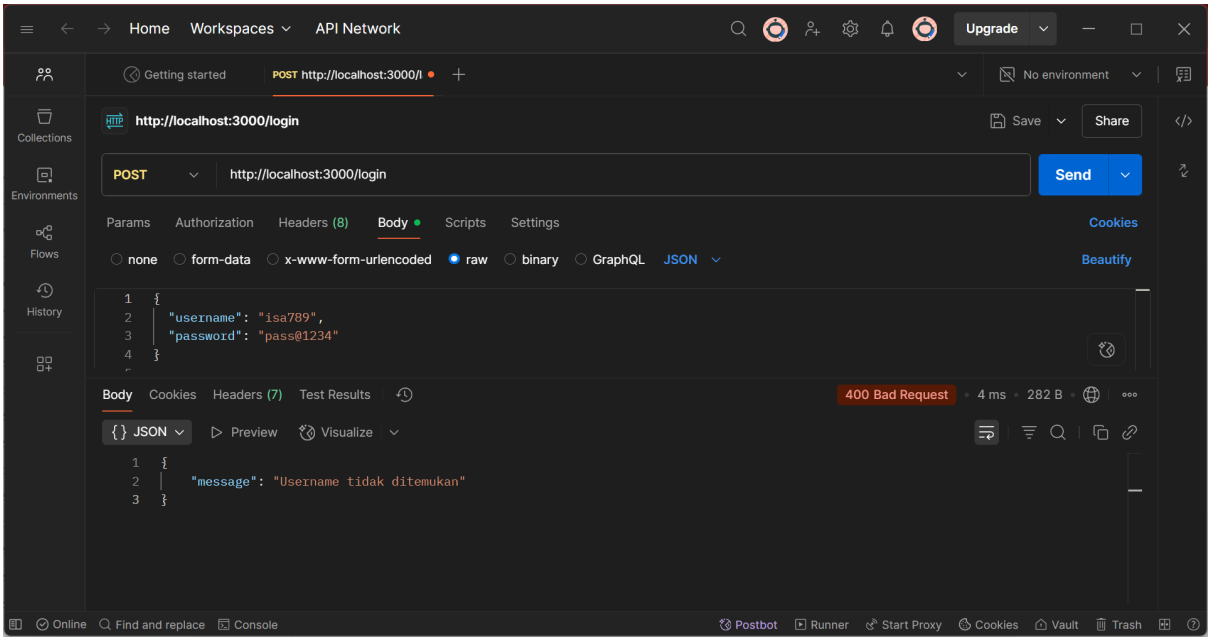
- Memastikan password:
  - **Panjang** 8–20 karakter.
  - **Mengandung simbol** (!@#$%^&*).
  - **Tidak mengandung username** di dalamnya.

Output

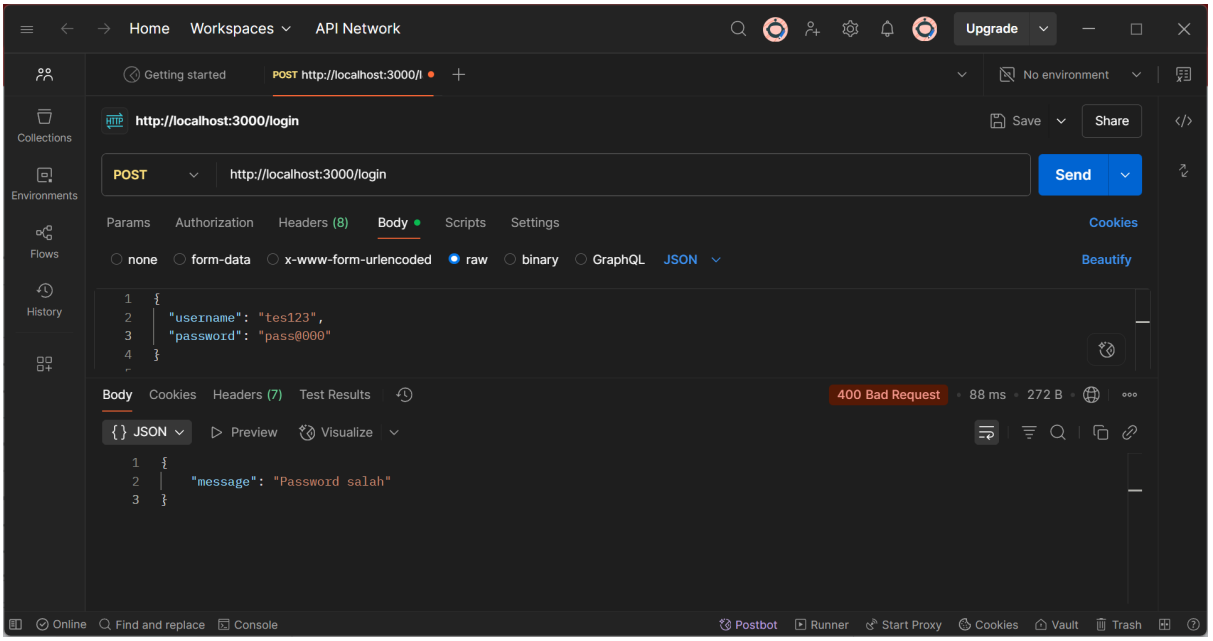




Jika username salah:



Jika password salah:



Sebelum username dan password dimasukkan ke users.json:

Setelah username dan password dimasukkan ke users.json: