

# 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. Masukkan script berikut pada [auth.controller.js](#) yang telah dibuat

```
JS auth.controller.js U X
controllers > JS auth.controller.js > login > login > User.findByEmail() callback
1  const User = require('../models/user.model');
2  const bcrypt = require('bcryptjs');
3  const jwt = require('jsonwebtoken');
4
5  exports.login = (req, res) => {
6    const { email, password } = req.body;
7
8    User.findByEmail(email, (err, results) => {
9      if (err) return res.status(500).json({ message: err.message });
10     if (results.length === 0) return res.status(404).json({ message: "User not found" });
11
12     const user = results[0];
13
14     const match = bcrypt.compareSync(password, user.password);
15     if (!match) return res.status(400).json({ message: "Wrong password" });
16
17     const token = jwt.sign(
18       { id: user.id, email: user.email },
19       process.env.JWT_SECRET,
20       { expiresIn: "7d" }
21     );
22
23     res.json({
24       message: "Login success",
25       token,
26       user: { id: user.id, name: user.name, email: user.email }
27     });
28   });
29 };
```

8. Ubah [auth.middleware.js](#) yang sebelumnya menggunakan token biasa, menjadi json web token seperti gambar dibawah ini

```
h.controller.js U    JS user.model.js M    JS auth.middlewares.js M X
middlewares > JS 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" });
      }

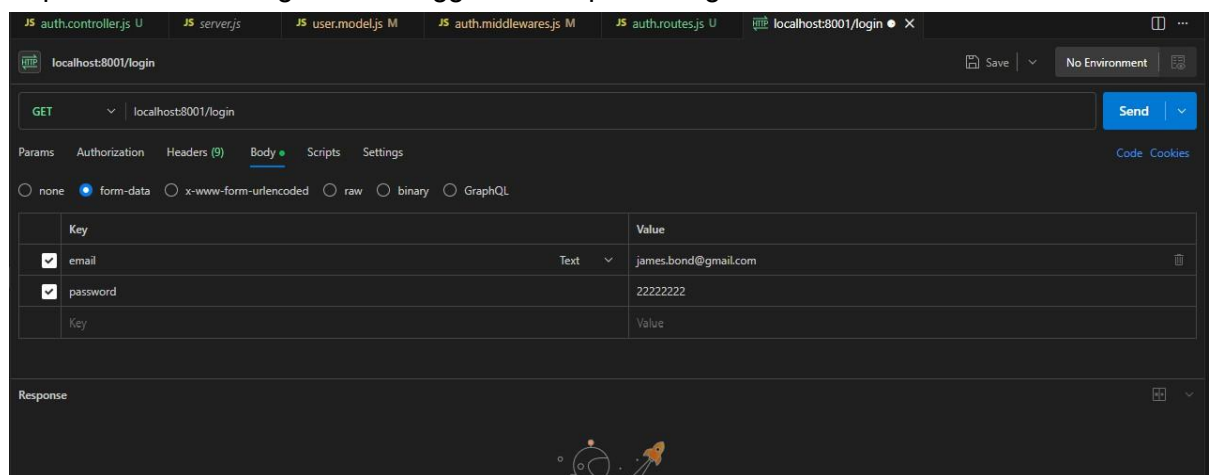
      req.user = results[0];
      next();
    });
  } catch (err) {
    return res.status(401).json({ message: "Invalid token" });
  }
};
```

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

```
JS auth.controller.js U JS user.model.js M JS auth.middlewares.js M JS auth.routes.js U X
routes > JS auth.routes.js > ...
1  const express = require("express");
2  const router = express.Router();
3  const authController = require("../controllers/auth.controller");
4
5  router.post("/login", authController.login);
6
7  module.exports = router;
```

## 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 /login



3. Catat bearer yang di dapatkan, lalu gunakan bearer tersebut untuk memanggil endpoints lainnya yang pada praktikum 9 telah di proteksi.

## F. Github + Visual Code

### 1. Buat proyek di Github dengan nama **Latihan9**

git init

git add .

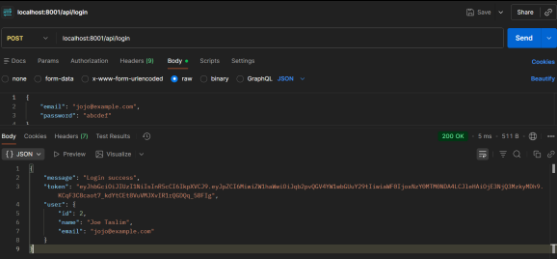
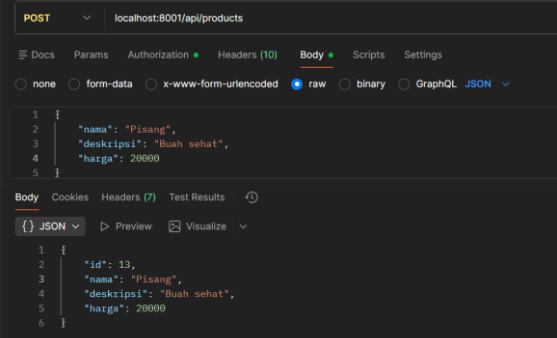
git commit -m "first commit"

git branch -M main

git remote add origin https://github.com/agunghakase/Latihan9.git

git push -u origin main

## Hasil Pengerjaan

No.	Instruksi	Screenshot	Kendala/Saran
A.	Generate token dan uji coba api		
1.	Generate token		Tidak ada kendala ataupun saran pada tahapan ini
2.	Uji coba api/products		Tidak ada kendala ataupun saran pada tahapan ini
B.	Github		
1.	Commit	<pre>incentt-0/latihan10.git PS D:\expressjs\APIproject10&gt; git push -u origin main Enumerating objects: 5446, done. Counting objects: 100% (5446/5446), done. Delta compression using up to 12 threads Compressing objects: 100% (5185/5185), done. Writing objects: 100% (5446/5446), 5.85 MiB   1.63 MiB/s, done. Total 5446 (delta 1893), reused 0 (delta 0), pack-reused 0 remote: Resolving deltas: 100% (1893/1893), done. remote: This repository moved. Please use the new location: remote: https://github.com/Vincentt-0/Latihan10.git To https://github.com/Vincentt-0/Latihan10.git  * [new branch] main -&gt; main branch 'main' set up to track 'origin/main'. PS D:\expressjs\APIproject10&gt;</pre>	
2.	Link Github	<a href="https://github.com/Vincentt-0/Latihan10">https://github.com/Vincentt-0/Latihan10</a>	