

Praktikum 9 - 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. Menyediakan API Enpoints

1. Lanjutkan Project Praktikum 8, dengan menggunakan file yang sama (copy)
2. Tambahkan pada database, sebuah tabel produk
Isi tabel produk seperti pada tabel berikut: (Buat 10 item)

id	nama	deskripsi	harga	foto
1.	Indomie Goreng		2500	images/miegor eng.jpg
2.
10.

3. Seperti pada perintah praktikum 8 buatkan beberapa endpoints
GET : localhost:8001/api/products/
GET : localhost:8001/api/products/:id
POST : localhost:8001/api/products/
PUT : localhost:8001/api/products/:id
DELETE: localhost:8001/api/products/:id
4. Pastikan memiliki file dengan struktur sebagai berikut.
controllers/[products.controller.js](#)
routes/[products.routes.js](#)
models/[products.model.js](#)
5. Test API dengan menggunakan **POSTMAN**

B. Menambahkan Proteksi API (Simple - Bearer Method)

1. Buat sebuah folder bernama middlewares, kemudian buat didalamnya sebuah file dengan nama [auth.middleware.js](#) dengan kode program seperti dibawah ini

```
export const authBearer = (req, res, next) => {

    const authHeader = req.headers.authorization;

    // Tidak ada Authorization
    if (!authHeader) {
        return res.status(401).json({ message: "No authorization header" });
    }

    // Harus Bearer
    if (!authHeader.startsWith("Bearer ")) {
        return res.status(401).json({ message: "Bearer token required" });
    }

    // Ambil token
    const token = authHeader.split(" ")[1];

    // Token yang benar (misal hardcode)
    const VALID_TOKEN = "12345TOKENRAHASIA";

    if (token !== VALID_TOKEN) {
        return res.status(403).json({ message: "Invalid token" });
    }

    next();
};
```

2. Pada file [product.routes.js](#) panggil [auth.middleware.js](#)
import { authBearer } from "../middleware/auth.middleware.js"
3. Lalu tambahkan authBearer ke endpoints yang perlu di proteksi
router.post("/", authBearer, createProduct);
router.put("/:id", authBearer, updateProduct);
router.delete("/:id", authBearer, deleteProduct);
4. Gunakan POSTMAN untuk akses 3 endpoints ini dengan menambahkan bearer
12345TOKENRAHASIA

F. Github + Visual Code

- Buat proyek di Github dengan nama **Latihan9**

```
git init
```

```
git add .
```

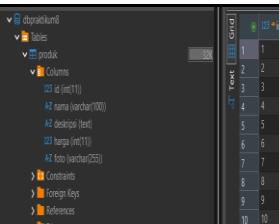
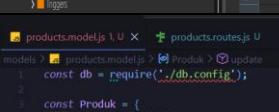
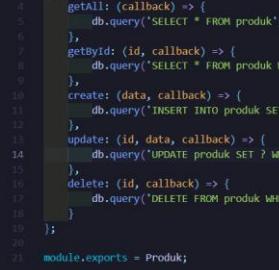
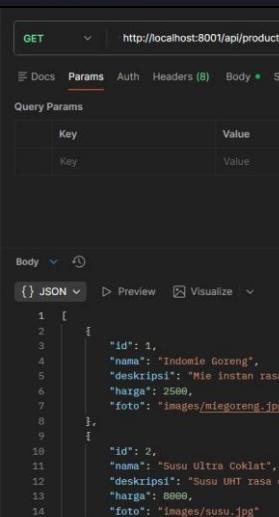
```
git commit -m "first commit"
```

```
git branch -M main
```

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

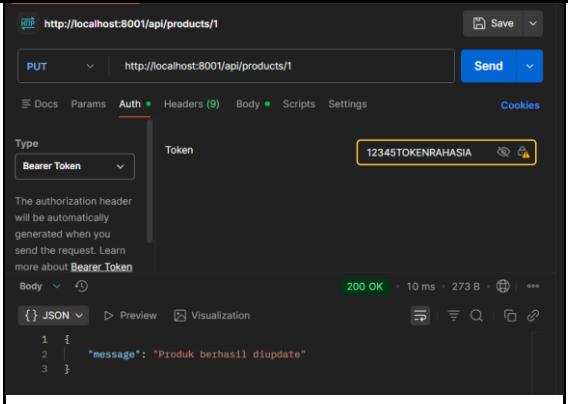
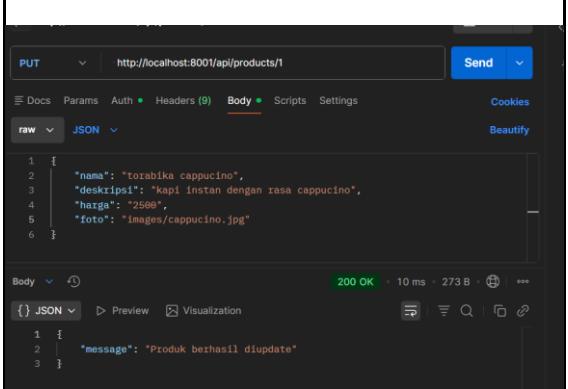
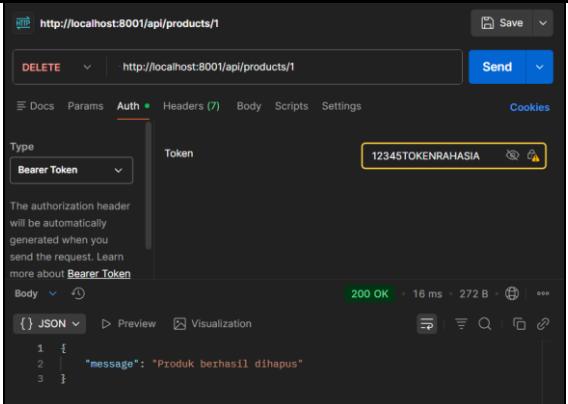
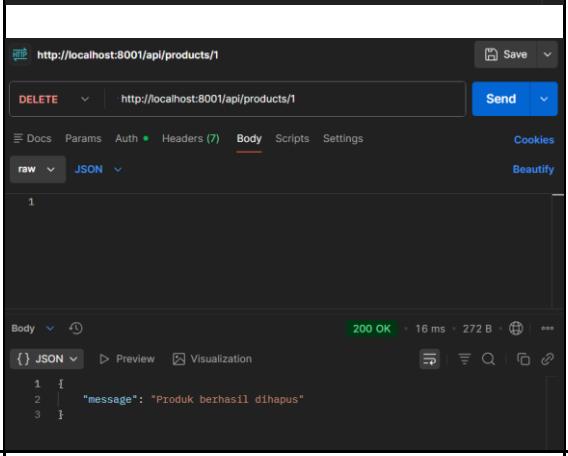
```
git push -u origin main
```

Hasil Pengajaran

No.	Instruksi	Screenshot	Kendala/Saran
A.	Installasi dan Konfigurasi		
1.	Membuat tabel produk beserta isi data		
2.	Buat struktur controller, routes, dan model produk		
3.	Tes postman melihat semua produk		

4.	Tes postman melihat produk berdasarkan id	<pre> 1 { 2 "id": 3, 3 "nama": "Roti Sari Gandum", 4 "deskripsi": "Roti gandum tinggi serat, 2 potong", 5 "harga": 5000, 6 "foto": "images/roti.jpg" 7 }</pre>	
5.	Tes postman menambah produk baru	<pre> 1 { 2 "nama": "Kopi Kapal Api", 3 "deskripsi": "Kopi instan rasa original", 4 "harga": "2000", 5 "foto": "images/kopi.jpg" 6 }</pre> <pre> 1 { 2 "id": 11, 3 "nama": "Kopi Kapal Api", 4 "deskripsi": "Kopi instan rasa original", 5 "harga": "2000", 6 "foto": "images/kopi.jpg" 7 }</pre>	
6.	Tes postman mengedit produk berdasarkan id	<pre> 1 { 2 "nama": "Roti Top Baker Bulat", 3 "deskripsi": "Roti enak rasa coklat", 4 "harga": "3500", 5 "foto": "images/roticoklat.jpg" 6 }</pre> <pre> 1 { 2 "message": "Produk berhasil diupdate" 3 }</pre>	
7.	Tes postman menghapus produk berdasarkan id	<pre> 1 Ctrl+Alt+P to Ask AI</pre> <pre> 1 { 2 "message": "Produk berhasil dihapus" 3 }</pre>	

8.	Membuat auth.middleware.js	<pre> auth.middleware.js export const authBearer = (req, res, next) => { const authHeader = req.headers.authorization; // Tidak ada Authorization if (!authHeader) { return res.status(401).json({ message: "No authorization header"}); } // Harus Bearer if (!authHeader.startsWith("Bearer ")) { return res.status(401).json({ message: "Bearer token required"}); } // Ambil token const token = authHeader.split(" ")[1]; // Token yang benar (misal hardcode) const VALID_TOKEN = "12345TOKENRAHASIA"; if (token !== VALID_TOKEN) { return res.status(403).json({ message: "Invalid token" }); } next(); } </pre>
9.	Memanggil auth.middleware.js dari product.routes.js dan menambahkan authBearer ke endpoint yang perlu di Proteksi	<pre> product.routes.js const express = require('express'); const router = express.Router(); const productController = require('../controllers/products.controller'); const { authBearer } = require('../middlewares/auth.middleware'); router.get('/', productController.getAllProducts); router.get('/:id', productController.getProductById); router.post('/', productController.createProduct); router.put('/:id', productController.updateProduct); router.delete('/:id', productController.deleteProduct); // authBearer router.post("/", authBearer, productController.createProduct); router.put("/:id", authBearer, productController.updateProduct); router.delete("/:id", authBearer, productController.deleteProduct); module.exports = router; </pre>
10.	Tes postman menambah produk baru dengan menambahkan bearer	<p>http://localhost:8001/api/products/</p> <p>POST http://localhost:8001/api/products/</p> <p>Auth Headers (9)</p> <p>Type: Bearer Token</p> <p>Token: 12345TOKENRAHASIA</p> <p>Body: {} JSON</p> <p>201 Created 18 ms 352 B</p> <pre> 1: { 2: "id": 16, 3: "nama": "susu milku", 4: "deskripsi": "susu UHT rasa original", 5: "harga": "3500", 6: "foto": "images/susumilku.jpg" 7: } </pre> <p>POST http://localhost:8001/api/products/</p> <p>Auth Headers (9)</p> <p>Body: raw JSON</p> <p>201 Created 46 ms 352 B</p> <pre> 1: { 2: "nama": "susu milku", 3: "deskripsi": "susu UHT rasa original", 4: "harga": "3500", 5: "foto": "images/susumilku.jpg" 6: } </pre>

11.	Tes postman mengedit produk berdasarkan id dengan menambahkan bearer	 	
12.	Tes postman menghapus produk berdasarkan id dengan menambahkan bearer	 	
B.	Github dan Viscode		

1.

Upload di github

