

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/miegoreng.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

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

git init

git add

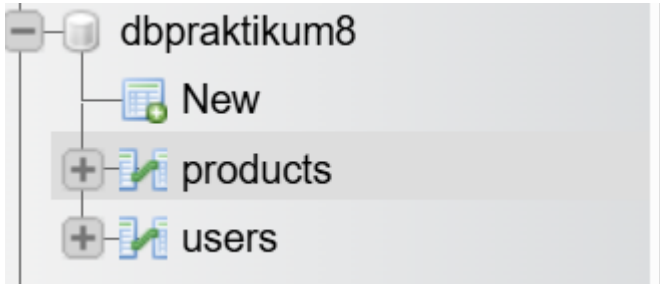

.

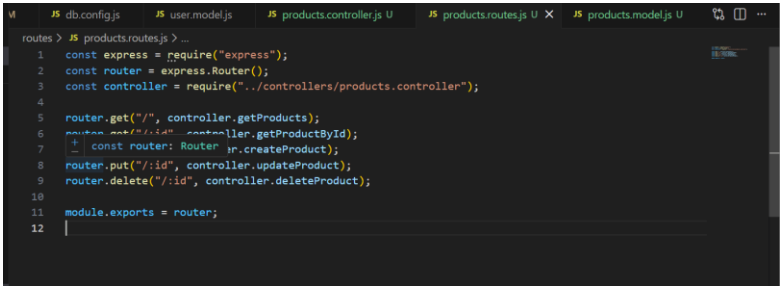
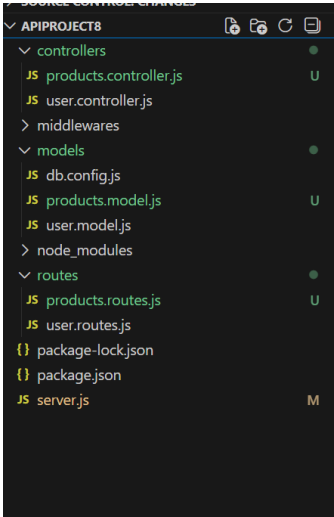
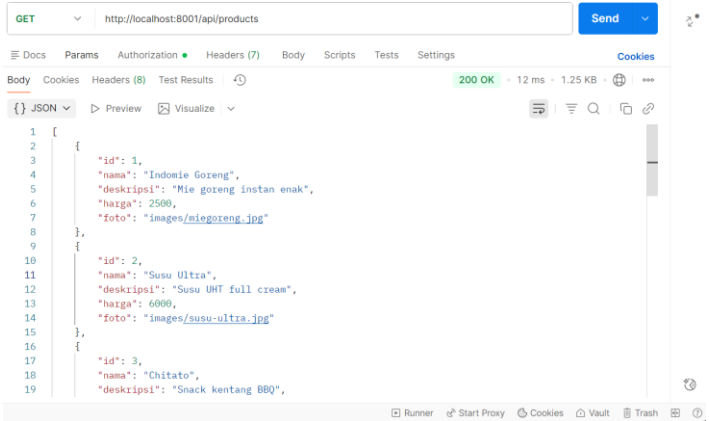
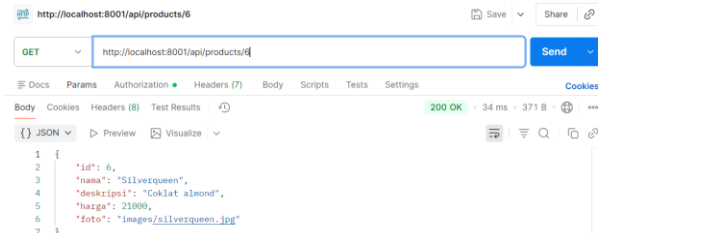
git commit -m "first commit" git branch -M main git remote add

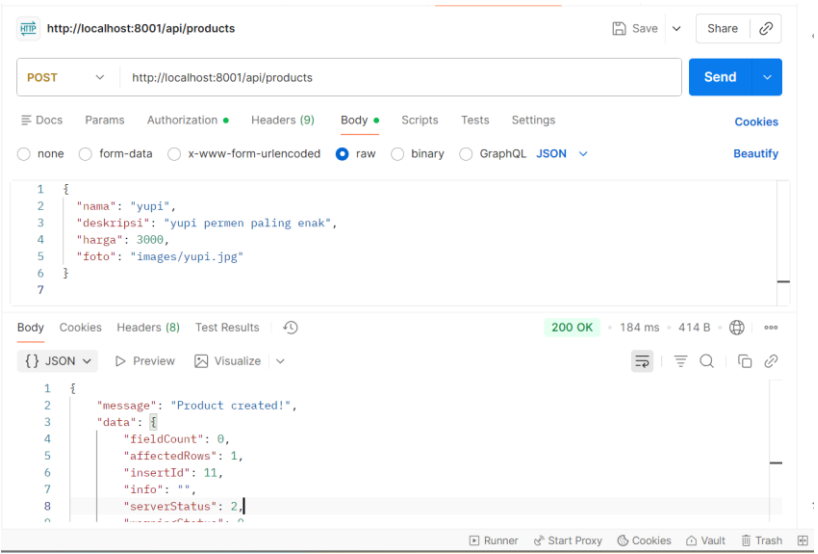
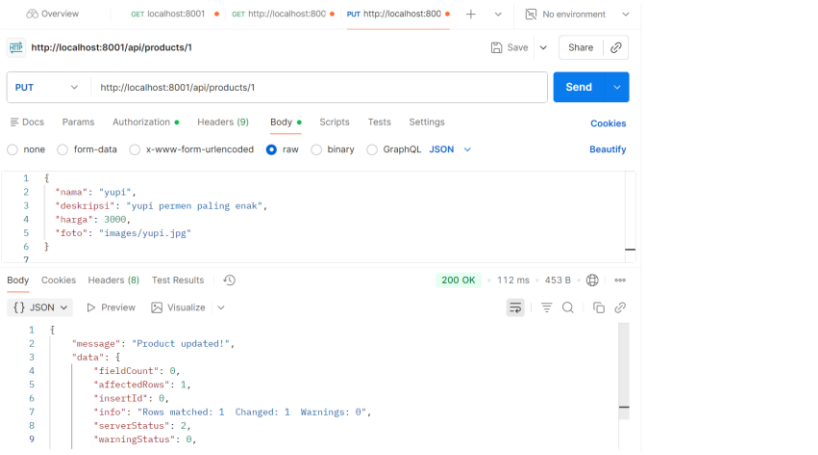
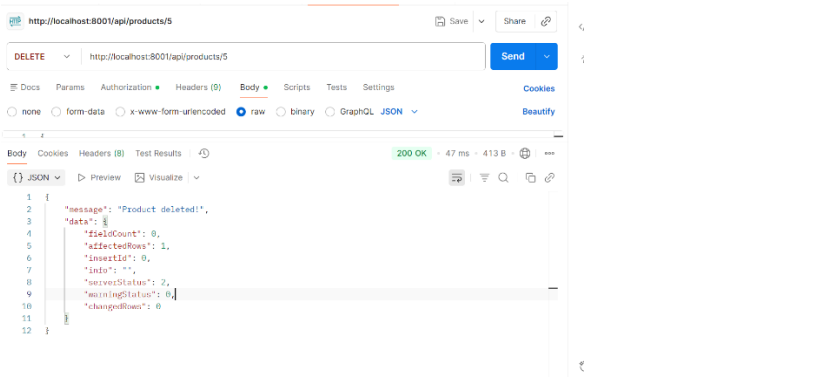
origin https://github.com/agunghakase/Latihan9.git git push -u

origin main

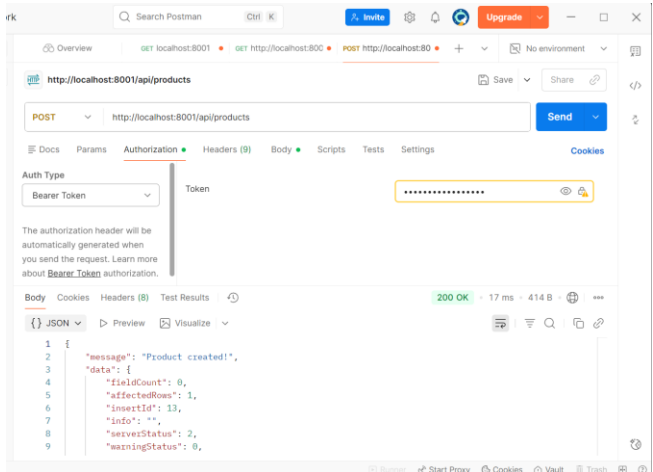
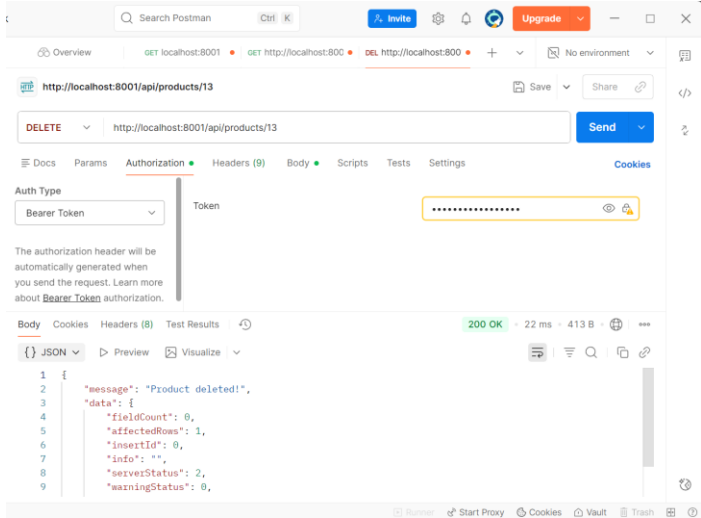
Hasil Pengerjaan

No.	Instruksi	Screenshot	Ke
A.	Instalasi dan Konfigurasi		
1.	Membuat table baru products		
2.	Mengisi table producst		

3.	Membuat endpoint	 <pre> 1 const express = require("express"); 2 const router = express.Router(); 3 const controller = require("../controllers/products.controller"); 4 5 router.get("/", controller.getProducts); 6 router.get("/:id", controller.getProductById); 7 + const router: Router in createProduct(); 8 router.put("/:id", controller.updateProduct); 9 router.delete("/:id", controller.deleteProduct); 10 11 module.exports = router; 12 </pre>
4.	Struktur file	 <pre> APIPROJECT8 ├── controllers │ ├── products.controller.js │ └── user.controller.js ├── middlewares ├── models │ ├── db.config.js │ ├── products.model.js │ └── user.model.js ├── node_modules ├── routes │ ├── products.routes.js │ └── user.routes.js ├── package-lock.json ├── package.json └── server.js </pre>
5.	Test API localhost:8001/api/products/	 <pre> 1 [2 { 3 "id": 1, 4 "nama": "Indomie Goreng", 5 "deskripsi": "Mie goreng instan enak", 6 "harga": 2500, 7 "foto": "images/miegoreng.jpg" 8 }, 9 { 10 "id": 2, 11 "nama": "Susu Ultra", 12 "deskripsi": "Susu UHT full cream", 13 "harga": 6000, 14 "foto": "images/susu-ultra.jpg" 15 }, 16 { 17 "id": 3, 18 "nama": "Chitato", 19 "deskripsi": "Snack kentang BBQ", </pre>
6.	Test API localhost:8001/api/products/	 <pre> 1 { 2 "id": 6, 3 "nama": "Silverqueen", 4 "deskripsi": "Coklat almond", 5 "harga": 21000, 6 "foto": "images/silverqueen.jpg" 7 } </pre>

7.	POST : localhost:8001/api/products/	 <pre>POST http://localhost:8001/api/products/ { "nama": "yupi", "deskripsi": "yupi peimen paling enak", "harga": 3000, "foto": "images/yupi.jpg" }</pre> <pre>{ "message": "Product created!", "data": { "fieldCount": 0, "affectedRows": 1, "insertId": 11, "info": "", "serverStatus": 2 } }</pre>
	PUT : localhost:8001/api/products/:id	 <pre>PUT http://localhost:8001/api/products/1 { "nama": "yupi", "deskripsi": "yupi peimen paling enak", "harga": 3000, "foto": "images/yupi.jpg" }</pre> <pre>{ "message": "Product updated!", "data": { "fieldCount": 0, "affectedRows": 1, "insertId": 0, "info": "Rows matched: 1 Changed: 1 Warnings: 0", "serverStatus": 2, "warningStatus": 0 } }</pre>
	DELETE: localhost:8001/api/products/:id	 <pre>DELETE http://localhost:8001/api/products/5</pre> <pre>{ "message": "Product deleted!", "data": { "fieldCount": 0, "affectedRows": 1, "insertId": 0, "info": "", "serverStatus": 2, "warningStatus": 0, "changedRows": 0 } }</pre>
B.	Github dan Viscode	

1.	Membuat folder minndelware dan file auth.middleware.js	
2.	Memambahkan autbearer di products.routes.js	
3.	Mengakses put setelah di bearer	
4.	Mengakses post setelah di bearer	

5.	Mengakses post setelah di bearer	 <p>The screenshot shows the Postman interface for a POST request to <code>http://localhost:8001/api/products</code>. The request is authenticated using a Bearer Token. The response is a 200 OK status with a response time of 17 ms and a body size of 414 B. The JSON response body is as follows:</p> <pre> 1 { 2 "message": "Product created!", 3 "data": { 4 "fieldCount": 0, 5 "affectedRows": 1, 6 "insertId": 13, 7 "info": "", 8 "serverStatus": 2, 9 "warningStatus": 0, </pre>
6.	Mengakses delete setelah di bearer	 <p>The screenshot shows the Postman interface for a DELETE request to <code>http://localhost:8001/api/products/13</code>. The request is authenticated using a Bearer Token. The response is a 200 OK status with a response time of 22 ms and a body size of 413 B. The JSON response body is as follows:</p> <pre> 1 { 2 "message": "Product deleted!", 3 "data": { 4 "fieldCount": 0, 5 "affectedRows": 1, 6 "insertId": 0, 7 "info": "", 8 "serverStatus": 2, 9 "warningStatus": 0, </pre>
E.	GITHUB	
1.	Link github	https://github.com/ftnaa/latihan-8.git