

NIM : 3312411077

Nama : Anisa Frity Amelia

Kelas : IF3A Web Pagi

Link Github : <https://github.com/anisafrityamelia/praktikum8.git>

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**

C. Github + Visual Code

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

git init

git add .


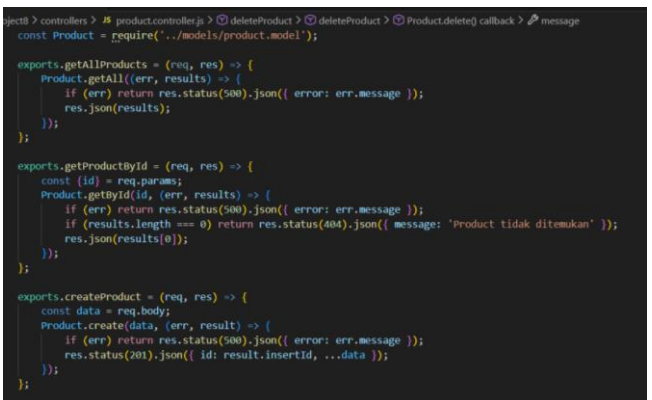
git commit -m "first commit"

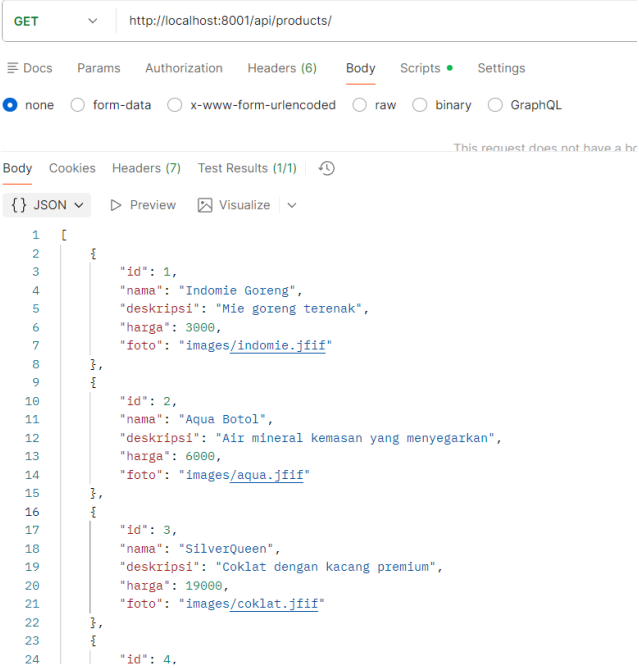
git branch -M main

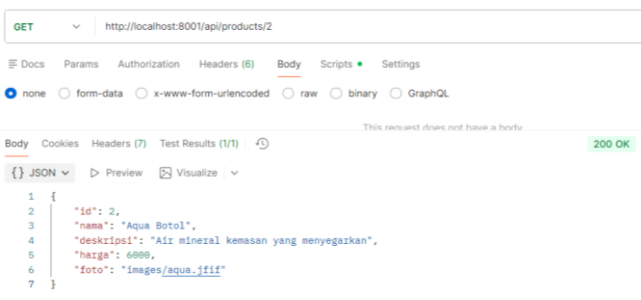
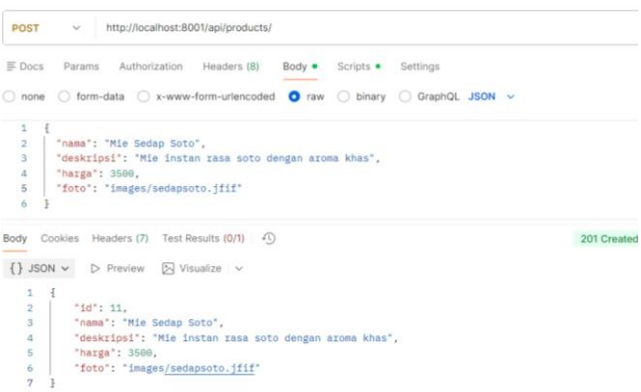
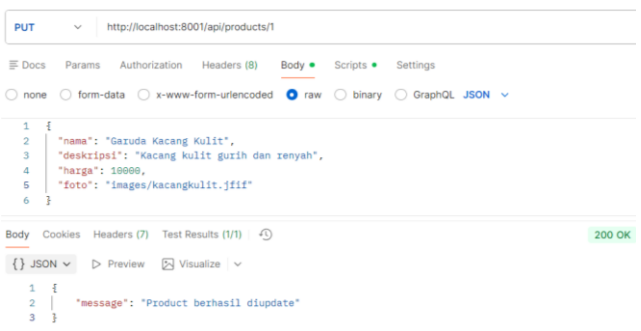
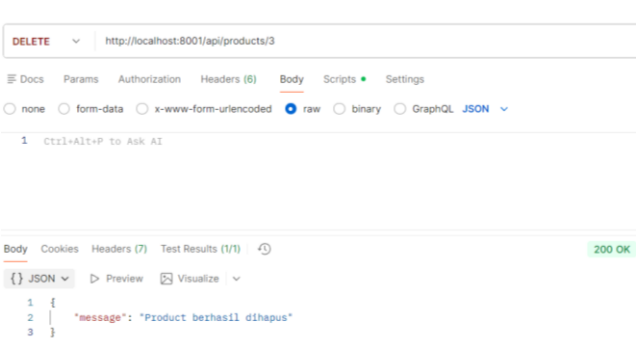
git remote add origin <https://github.com/agunghakase/Latihan9.git>

git push -u origin main

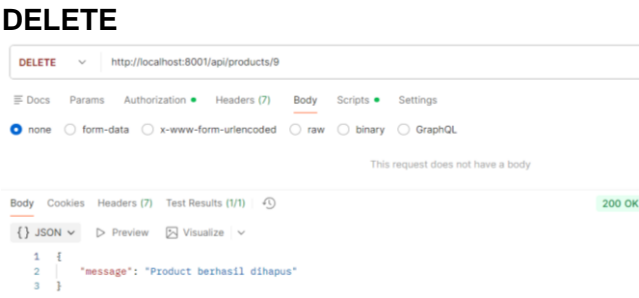
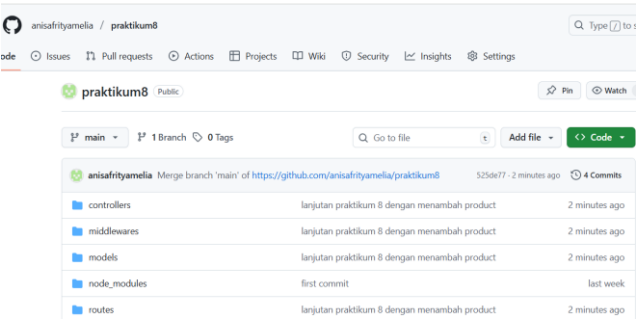
Hasil Pengerjaan

No	Instruksi	Screenshot	Kendala/ Saran
A	Menyediakan API Endpoints		
1.	Pada database, tambahkan sebuah tabel products		
2.	Di dalam folder routes buat file dengan nama product.routes.js dan ketik kode program seperti gambar disamping		
3.	Buat file dalam folder controllers dengan nama product.controller.js dan ketik kode program seperti gambar disamping		

		<pre> exports.updateProduct = (req, res) => { const {id} = req.params; const data = req.body; Product.update(id, data, (err, result) => { if (err) return res.status(500).json({ error: err.message }); if (result.affectedRows === 0) return res.status(404).json({ message: 'Product tidak ditemukan' }); res.json({ message: 'Product berhasil diupdate' }); }); }; exports.deleteProduct = (req, res) => { const {id} = req.params; Product.delete(id, (err, result) => { if (err) return res.status(500).json({ error: err.message }); if (result.affectedRows === 0) return res.status(404).json({ message: 'Product tidak ditemukan' }); res.json({ message: 'Product berhasil dihapus' }); }); }; </pre>	
4.	Update file server.js dengan menambahkan kode seperti gambar disamping	<pre> project8 > JS server.js > ... 1 const express = require('express'); 2 const app = express(); 3 const PORT = 8001; 4 5 app.use(express.json()); 6 7 app.get('/', (req, res) => { 8 res.send('Hello, World!'); 9 }); 10 11 // Routes 12 const userRoutes = require('./routes/user.routes'); 13 app.use('/api/users', userRoutes); 14 15 const productRoutes = require('./routes/product.routes'); 16 app.use('/api/products', productRoutes); 17 18 app.listen(PORT, () => { 19 console.log(`Server berjalan di http://localhost:\${PORT}`); 20 }); </pre>	
5.	Buat file untuk model product di dalam folder models dengan nama product.model.js	<pre> project8 > models > JS product.model.js > Product 1 const db = require('../db.config'); 2 3 const Product = { 4 getAll: callback => { 5 db.query('SELECT * FROM products', callback); 6 }, 7 getById: (id, callback) => { 8 db.query('SELECT * FROM products WHERE id = ?', [id], callback); 9 }, 10 create: (data, callback) => { 11 db.query('INSERT INTO products (nama, deskripsi, harga, foto) VALUES (?, ?, ?, ?)', [data.nama, data.deskripsi, data.harga, data.foto], callback); 12 }, 13 update: (id, data, callback) => { 14 db.query('UPDATE products SET nama = ?, deskripsi = ?, harga = ?, foto = ? WHERE id = ?', [data.nama, data.deskripsi, data.harga, data.foto, id], callback); 15 }, 16 delete: (id, callback) => { 17 db.query('DELETE FROM products WHERE id = ?', [id], callback); 18 } 19 }; 20 21 module.exports = Product; </pre>	
6.	Jalankan perintah node server.js dan gunakan postman untuk mendapatkan data getAll products dengan mengunjungi endpoints /api/products	 <p>The screenshot shows a Postman interface with a GET request to <code>http://localhost:8001/api/products/</code>. The response is a JSON array containing four product objects. The first product is 'Indomie Goreng' with a price of 3000. The second is 'Aqua Botol' with a price of 6000. The third is 'SilverQueen' with a price of 19000. The fourth is 'Coklat dengan kacang premium' with a price of 19000.</p> <pre> [{ "id": 1, "nama": "Indomie Goreng", "deskripsi": "Mie goreng terenak", "harga": 3000, "foto": "images/indomie.jfif" }, { "id": 2, "nama": "Aqua Botol", "deskripsi": "Air mineral kemasan yang menyegarkan", "harga": 6000, "foto": "images/aqua.jfif" }, { "id": 3, "nama": "SilverQueen", "deskripsi": "Coklat dengan kacang premium", "harga": 19000, "foto": "images/coklat.jfif" }, { "id": 4, </pre>	

7.	Dengan menggunakan postman lakukan pengujian pada endpoint /api/products/2 (method: GET)		
8.	Dengan menggunakan postman lakukan pengujian pada endpoint /api/products (method: POST)		
9.	Dengan menggunakan postman lakukan pengujian pada endpoint /api/products/1 (method: PUT)		
10.	Dengan menggunakan postman lakukan pengujian pada endpoint /api/products/3 (method: DELETE)		
B	Menambahkan Proteksi API (Simple - Bearer Method)		

1.	<p>Buat sebuah folder bernama <code>middlewares</code>, kemudian di dalamnya buat file dengan nama <code>auth.middleware.js</code> dan ketik kode program seperti gambar disamping</p>	<pre> project8 > middlewares > JS auth.middleware.js > authBearer 1 const authBearer = (req, res, next) => { 2 const authHeader = req.headers.authorization; 3 4 if (!authHeader) { 5 return res.status(401).json({ message: "No authorization header" }); 6 } 7 8 if (authHeader.startsWith("Bearer ")) { 9 return res.status(401).json({ message: "Bearer token required" }); 10 } 11 12 const token = authHeader.split(" ")[1]; 13 14 const VALID_TOKEN = "12345TOKENRAHASIA"; 15 16 if (token !== VALID_TOKEN) { 17 return res.status(403).json({ message: "Invalid token" }); 18 } 19 20 next(); 21 }; 22 23 module.exports = { authBearer }; </pre>	
2.	<p>Pada file <code>product.routes.js</code> panggil <code>auth.middleware.js</code>, lalu tambahkan <code>authBearer</code> ke endpoints yang perlu di proteksi, tambahkan kode program seperti pada gambar disamping</p>	<pre> project8 > routes > JS product.routes.js > ... 1 const express = require('express'); 2 const router = express.Router(); 3 const productController = require('../controllers/product.controller'); 4 const { authBearer } = require('../middlewares/auth.middleware'); 5 6 router.get('/', productController.getAllProducts); 7 router.get('/:id', productController.getProductById); 8 router.post('/', authBearer, productController.createProduct); 9 router.put('/:id', authBearer, productController.updateProduct); 10 router.delete('/:id', authBearer, productController.deleteProduct); 11 12 module.exports = router; </pre>	
3.	<p>Gunakan postman untuk akses 3 endpoints (POST, PUT, DELETE) dengan menambahkan bearer 12345TOKENRAHASIA</p>	<p>POST</p>  <p>PUT</p> 	

			
C	Github + Visual Code		
1.	Membuat repository baru di Github dan commit proyek ke dalam Github		
2.	Link Github	https://github.com/anisafriyamelia/praktikum8.git	