

# UAS II3160 Tugas 2 - Mengembangkan Layanan Microservices

## Nutrition Service API

Disusun oleh: Ahmad Evander Ruizhi Xavier (18223064) Rekan Kelompok: Favian Rafi Laftiyanto

### A. Deskripsi Microservice

Nutrition Service API merupakan sebuah **microservice** yang bertugas untuk mengelola **perhitungan kebutuhan nutrisi** dan **otentikasi berbasis token**. Layanan ini dikembangkan sebagai bagian dari **Personal Fitness Management System**, di mana Nutrition Service berperan sebagai **microservice** inti yang menyediakan informasi nutrisi harian berdasarkan kondisi fisik pengguna.

Microservice ini dirancang menggunakan arsitektur **RESTful API**, dibangun dengan framework **CodeIgniter 4**, serta dikemas menggunakan **Docker** agar mudah dideploy pada berbagai environment, termasuk **Set Top Box (STB)**.

### B. Fitur Utama

- Otentikasi pengguna menggunakan **JWT (JSON Web Token)**
- Perhitungan kebutuhan nutrisi harian
  - Kalori
  - Makronutrien (karbohidrat, protein, lemak)
  - Mikronutrien (gula, garam, serat)
- Endpoint terproteksi menggunakan **JWT Filter**
- Dapat diintegrasikan dengan microservice lain

### C. Teknologi yang Digunakan

- Language: PHP 8.2
- Backend Framework: CodeIgniter 4
- Web Server: Apache HTTP Server
- Containerization: Docker & Docker Compose
- Authentication: JSON Web Token (JWT)

### D. Arsitektur Sistem

```
Client (Web based)
↓
Authentication Service (JWT Token)
↓
Nutrition Service API (Docker - STB)
```

### E. Struktur Direktori Penting

```

nutrition-serviceAPI/
├── app/
│   ├── Controllers/
│   │   └── Api/
│   │       ├── AuthController.php
│   │       │   # Controller untuk autentikasi user (login)
│   │       │   # Menghasilkan JWT token sebagai akses ke endpoint terproteksi
│   │       └── NutritionController.php
│   │           # Controller utama layanan nutrisi
│   │           # Mengelola perhitungan dan validasi kebutuhan nutrisi pengguna
│   ├── Filters/
│   │   └── JwtFilter.php
│   │       # Filter JWT untuk memvalidasi token pada request
│   │       # Digunakan pada endpoint yang membutuhkan autentikasi
│   └── Config/
│       └── Routes.php
│           # Konfigurasi routing API
│           # Mendefinisikan endpoint publik dan endpoint yang dilindungi JWT
├── public/
│   └── index.php
│       # Entry point utama aplikasi CodeIgniter
│       # Seluruh request HTTP diproses melalui file ini
├── docker-compose.yml
│   # Konfigurasi Docker Compose
│   # Mengatur port, build image, dan lifecycle container
├── Dockerfile
│   # Instruksi build image Docker
│   # Menggunakan PHP 8.2 + Apache sebagai web server
├── composer.json
│   # Dependency manager PHP
│   # Memuat CodeIgniter 4 dan library JWT
└── README.md
    # Dokumentasi microservice
    # Berisi arsitektur, endpoint API, dan cara menjalankan layanan

```

## F. Konfigurasi Environment

```

CI_ENVIRONMENT = development
app.baseURL = 'http://localhost:8085/'

JWT_SECRET = supersecretkey1234567890supersecretkey

```

Environment sebenarnya sudah dikonfigurasi pada docker-compose dan secara otomatis di-build saat menjalankan microservice sehingga tidak perlu membuat file `.env` lagi (tetapi konfigurasi environment diatas dapat digunakan apabila tidak di-build otomatis)

## G. Daftar Endpoint API

Method	Endpoint	Deskripsi	Autentikasi
GET	/api/ping	API check untuk menguji keaktifan API	-
POST	/api/login	Autentikasi pengguna dan generate JWT	-
POST	/api/nutrition/constraints	Mengolah kebutuhan nutrisi pengguna	JWT

## H.Cara Menjalankan Microservice dan Deployment pada STB dengan Docker

1. Clone Repository pada STB

```
git clone https://github.com/evanderruizhi2/nutrition-serviceAPI.git
cd nutrition-serviceAPI
```

atau dengan melakukan SSH ke STB terlebih dahulu jika ingin diakses dari PC/Laptop pribadi

```
ssh -o ProxyCommand="cloudflared access ssh --hostname %h" root@<stb-domain>
```

## 2. Build Image Docker

```
docker compose build
```

## 3. Jalankan Container

```
docker compose up -d
```

Pastikan container berjalan dengan:

```
docker ps
```

Contoh output jika berjalan:

```
0.0.0.0:8085 -> 80/tcp
```

# I. Cara Menggunakan API Microservice

Method	Endpoint	Contoh Request	Contoh Response
GET	/api/ping	curl https://evanrzh.theokaitou.my.id/api/ping	{"status": "ok"}
POST	/api/login	curl -X POST https://evanrzh.theokaitou.my.id/api/login	{"token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUz"
POST	/api/nutrition/constraints	curl -X POST https://evanrzh.theokaitou.my.id/api/nutrition/constraints -H "Authorization: Bearer <JWT_Token>" -H "Content-Type: application/json" --data-binary "@body.json"	terlampir

untuk menggunakan endpoint API /api/nutrition/constraints disarankan membuat dan menggunakan body.json agar sintaks lebih mudah untuk dieksekusi. berikut contoh isi dari body.json

```
{
  "age": 45, \\ wajib
  "weight": 82,
  "height": 168,
  "gender": "female",
  "activity_level": "moderate",
  "conditions": {
    "diabetes": true,
    "hypertension": true,
    "heart_disease": false
  }
}
```

- age merupakan umur dalam tahun
- weight merupakan berat badan dalam kg
- height merupakan tinggi badan dalam cm
- gender merupakan jenis kelamin (diisi dengan male/female)
- activity\_level merupakan tingkat intensitas aktivitas fisik (default value sedentary, dapat diisi dengan light/moderate/active)
- conditions berisi riwayat penyakit (diabetes, hipertensi, penyakit jantung)
- age sampai gender wajib diisi, sisanya opsional

berikut adalah contoh response untuk endpoint API /api/nutrition/constraints menggunakan input dari body.json

```
{
  "meta": {
    "age": 45,
    "gender": "female",
    "bmi": 29.1,
    "bmi_category": "overweight",
    "bmr": 1484,
    "daily_calorie_needs": 2300
  },
  "constraints": {
    "max_calories_per_serving": 300,
    "macros": {
      "carbohydrates": {
        "max_g": 37
      },
      "protein": {
        "min_g": 18
      },
      "fat": {
        "max_g": 8
      }
    },
    "micros": {
      "sodium_mg_max": 300,
      "sugars_g_max": 5,
      "dietary_fiber_g_min": 6
    }
  },
  "diet_flags": {
    "low_sugar": true,
    "low_sodium": true,
    "heart_friendly": false,
    "high_fiber": true
  }
}
```