Load Balanced Web API dengan Multi-Container Docker (Backend, Load Balancer dan Redis Database)

Disusun oleh:

Alayavaro Rachmadia	2410501095
Panji Anugerah Panengah	2410501105
Rakha Abyan Hertamtama	2410501089
Rizki Ramadhan	2410501091
Shidqi Athalla Arka Qafriyanto	2410501077

Jalan RS. Fatmawati Raya, Pd. Labu, Kec. Cilandak, Kota Jakarta Selatan, Daerah Khusus Ibukota Jakarta 12450

D3 Sistem Informasi

2025

1. Deskripsi Arsitektur

Project ini membangun sebuah sistem *multi-container* menggunakan Docker Compose yang terdiri dari empat service utama:

1. **Backend1 (Flask API instance 1)**
2. **Backend2 (Flask API instance 2)**
3. **Redis Database** untuk penyimpanan data (*total_visits*)
4. **Nginx Reverse Proxy** berfungsi sebagai load balancer

Ketika pengguna mengakses **http://localhost:8080**, request akan masuk ke Nginx → didistribusikan secara bergantian (*round-robin*) ke backend1 atau backend2. Setiap instance backend mencatat jumlah kunjungan ke Redis, sehingga data tetap tersimpan meskipun container backend dihentikan atau dihapus.

2. Diagram Arsitektur

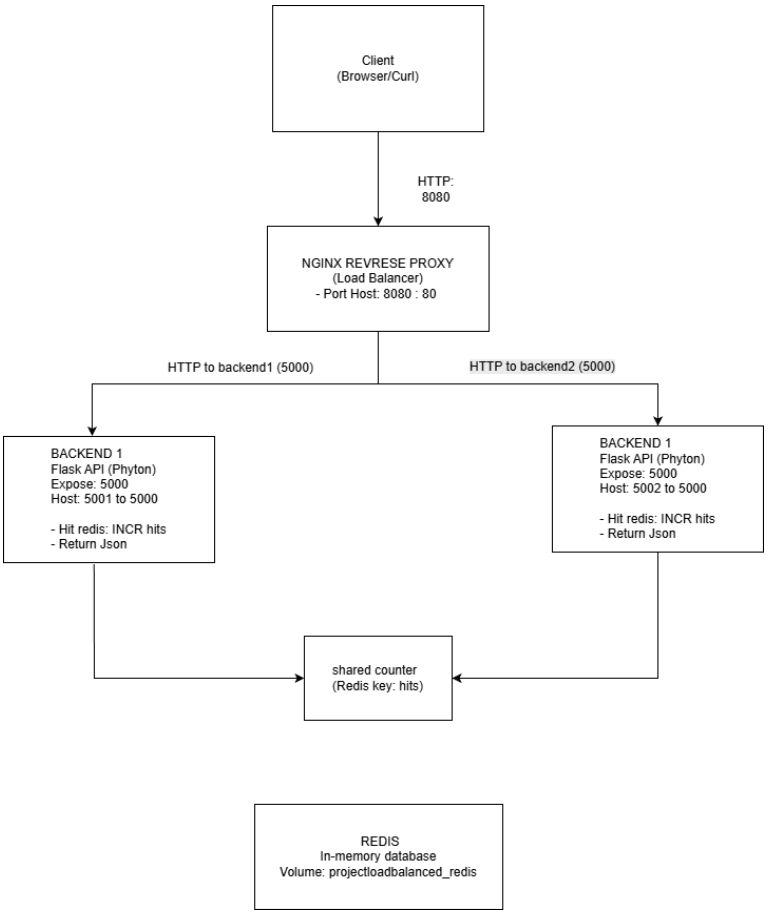
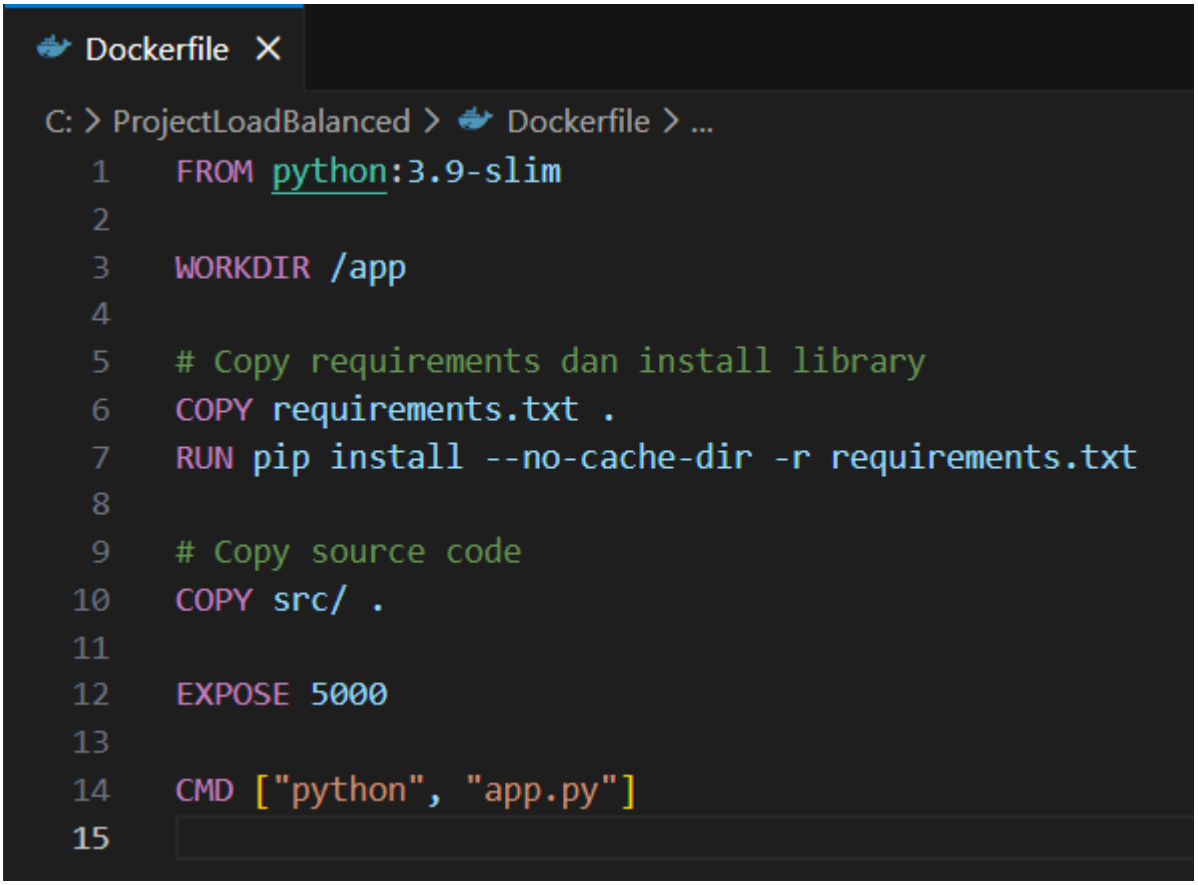


Diagram tersebut menunjukkan arsitektur Load Balanced Web API di mana client mengirim request ke Nginx (port 8080) sebagai load balancer. Nginx kemudian membagi request secara bergantian ke dua backend Flask API (Backend 1 dan Backend 2). Setiap backend mengakses Redis untuk menambah nilai counter (hits) yang digunakan bersama (shared). Hasilnya, beban kerja terbagi merata, performa lebih stabil, dan data tetap tersinkron antar backend.

3. Penjelasan Dockerfile



Penjelasan:

- **Base image:** python:3.9-slim (ringan & stabil)
- **WORKDIR /app:** lokasi file aplikasi
- **COPY src/:** menyalin app.py
- **Install dependency:** Flask & Redis client
- **Expose 5000:** port API
- **CMD:** menjalankan Flask server

Dockerfile ini digunakan untuk membangun image backend1 dan backend2.

4. Penjelasan docker-compose.yml

Dockerfile | docker-compose.yml

C: > ProjectLoadBalanced > docker-compose.yml

```yaml
version: '3.8'

Run All Services
services:
  Run Service
  redis:
    image: redis:alpine
    container_name: redis
    ports:
      - "6379:6379"
    volumes:
      - redis_data:/data

  Run Service
  backend1:
    build: ./src
    container_name: backend1
    ports:
      - "5001:5000"
    environment:
      - REDIS_HOST=redis
    depends_on:
      - redis

  Run Service
  backend2:
    build: ./src
    container_name: backend2
    ports:
      - "5002:5000"
    environment:
      - REDIS_HOST=redis
    depends_on:
      - redis

  Run Service
  nginx:
    image: nginx:alpine
    container_name: nginx
    ports:
      - "8080:80"
    volumes:
      - ./configs/nginx.conf:/etc/nginx/nginx.conf:ro
    depends_on:
      - backend1
      - backend2

volumes:
  redis_data:
```

Penjelasan Service

Service	Fungsi
redis	database penyimpanan total kunjungan
backend1	instance API Flask pertama

backend2	instance API Flask kedua
nginx	reverse proxy + load balancer

Port Mapping

- Backend1 → localhost:5001 → port internal 5000
- Backend2 → localhost:5002 → port internal 5000
- Redis → localhost:6379
- Nginx → localhost:8080

Volume

- redis_data:/data (persistent storage)

Network

Semua service berada dalam 1 internal network bernama mynet.

5. Cara Menjalankan Project

Build Image Backend:

```
PS C:\ProjectLoadBalanced> docker build -t backend-image .
[+] Building 3.3s (11/11) FINISHED                                                                  docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                                0.0s
 => => transferring dockerfile: 275B                                                                                0.0s
 => [internal] load metadata for docker.io/library/python:3.9-slim                                                  2.9s
 => [auth] library/python:pull token for registry-1.docker.io                                                       0.0s
 => [internal] load .dockerignore                                                                                   0.0s
 => => transferring context: 2B                                                                                     0.0s
 => [1/5] FROM docker.io/library/python:3.9-slim@sha256:2d97f6910b16bd338d3060f261f53f144965f755599aab1acda1e13cf1731b1b  0.0s
 => => resolve docker.io/library/python:3.9-slim@sha256:2d97f6910b16bd338d3060f261f53f144965f755599aab1acda1e13cf1731b1b  0.0s
 => [internal] load build context                                                                                   0.0s
 => => transferring context: 90B                                                                                    0.0s
 => CACHED [2/5] WORKDIR /app                                                                                       0.0s
 => CACHED [3/5] COPY requirements.txt .                                                                            0.0s
 => CACHED [4/5] RUN pip install --no-cache-dir -r requirements.txt                                                 0.0s
 => CACHED [5/5] COPY src/ .                                                                                        0.0s
 => exporting to image                                                                                              0.1s
 => => exporting layers                                                                                             0.0s
 => => exporting manifest sha256:1abeb6834327d5ff622e70892aef1960f9cce76e6219deb965f6544a62fd7309                   0.0s
 => => exporting config sha256:4fccfb993fb84e0cf3502db3c18a1c5ccae714949c4599d0940efb5bbc7106ab                     0.0s
 => => exporting attestation manifest sha256:448a10b7a84ed0a02f65bf03dbc1403ddfdc07e9de71058b86b6180c6a6e31df       0.0s
 => => exporting manifest list sha256:4416b3e0f2dd9b3c47d3547ab098302254de889723ebd10328f92d12486d8075              0.0s
 => => naming to docker.io/library/backend-image:latest                                                             0.0s
 => => unpacking to docker.io/library/backend-image:latest                                                          0.0s
PS C:\ProjectLoadBalanced>
```

Docker Compose Up -d:

```
PS C:\ProjectLoadBalanced> docker compose up -d
time="2025-12-03T21:40:36+07:00" level=warning msg="C:\\ProjectLoadBalanced\\docker-compose.yml: the attribute `version` is obsolete,
 it will be ignored, please remove it to avoid potential confusion"
[+] Running 5/5
 ✓ Network projectloadbalanced_default  Created                                                                    0.1s
 ✓ Container redis                      Started                                                                    0.8s
 ✓ Container backend1                   Started                                                                    1.0s
 ✓ Container backend2                   Started                                                                    1.0s
 ✓ Container nginx                      Started                                                                    1.2s
PS C:\ProjectLoadBalanced>
```

Cek Kontainer Proses:

```
PS C:\ProjectLoadBalanced> docker ps
CONTAINER ID   IMAGE                          COMMAND                  CREATED          STATUS          PORTS
                NAMES
de5e21e7597d   nginx:alpine                   "/docker-entrypoint.…"   32 seconds ago   Up 30 seconds   0.0.0.0:8080->80/tcp, [::]:80
80->80/tcp      nginx
fe65b4364c13   projectloadbalanced-backend2   "python app.py"          32 seconds ago   Up 30 seconds   0.0.0.0:5002->5000/tcp, [::]:
5002->5000/tcp   backend2
d6f53c5499a2   projectloadbalanced-backend1   "python app.py"          32 seconds ago   Up 30 seconds   0.0.0.0:5001->5000/tcp, [::]:
5001->5000/tcp   backend1
8dec3b4ad075   redis:alpine                   "docker-entrypoint.s…"   32 seconds ago   Up 31 seconds   0.0.0.0:6379->6379/tcp, [::]:
6379->6379/tcp   redis
PS C:\ProjectLoadBalanced>
```

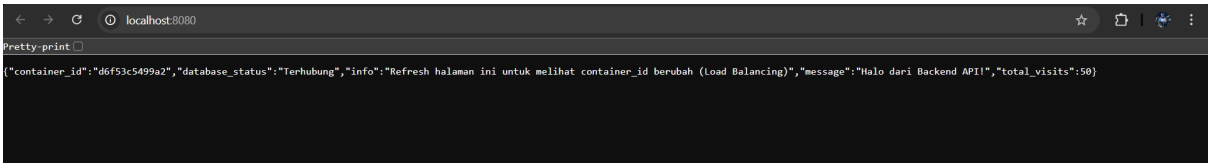
Log nginx:

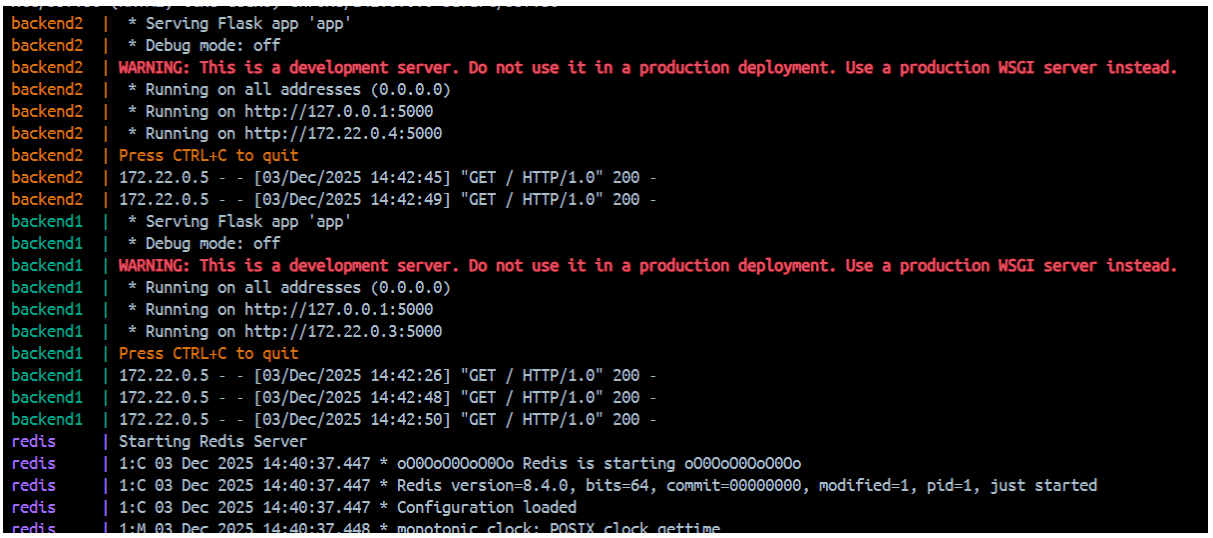
```
PS C:\ProjectLoadBalanced> docker compose logs nginx
time="2025-12-03T21:41:43+07:00" level=warning msg="C:\\ProjectLoadBalanced\\docker-compose.yml: the attribute `version` is obsolete,
 it will be ignored, please remove it to avoid potential confusion"
nginx  | /docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
nginx  | /docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
nginx  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
nginx  | 10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
nginx  | 10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
nginx  | /docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
nginx  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
nginx  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
nginx  | /docker-entrypoint.sh: Configuration complete; ready for start up
PS C:\ProjectLoadBalanced>
```

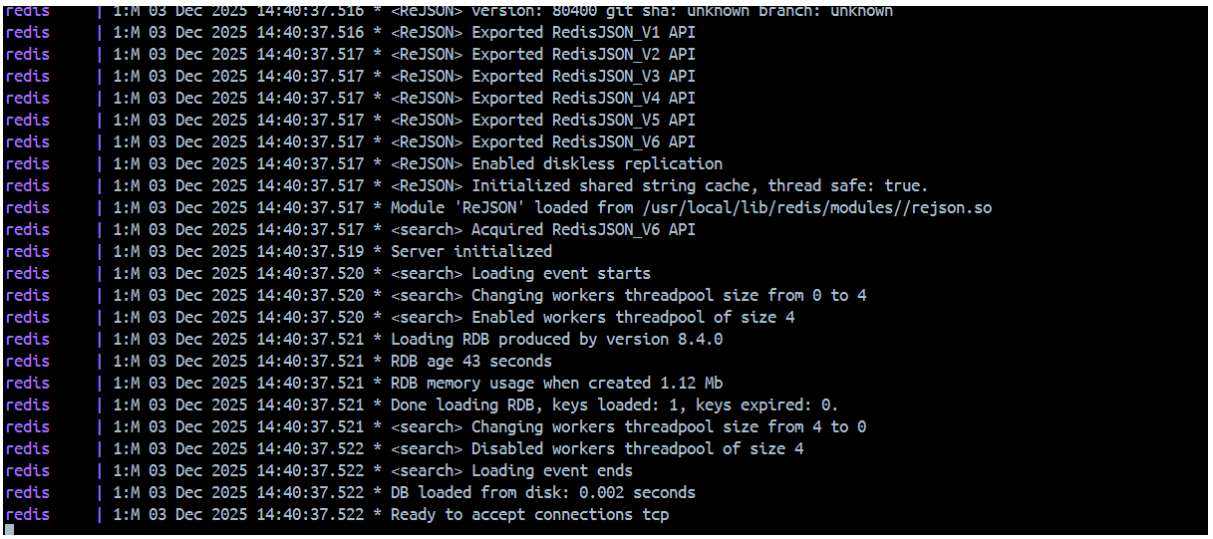
Test Api Via Load Balancer:

```
PS C:\ProjectLoadBalanced> curl http://localhost:8080


StatusCode        : 200
StatusDescription : OK
Content           : {"container_id":"d6f53c5499a2","database_status":"Terhubung","info":"Refresh halaman ini untuk melihat
                    container_id berubah (Load Balancing)","message":"Halo dari Backend API!","total_visits":46}

RawContent        : HTTP/1.1 200 OK
                    Connection: keep-alive
                    Content-Length: 196
                    Content-Type: application/json
                    Date: Wed, 03 Dec 2025 14:42:26 GMT
                    Server: nginx/1.29.3

                    {"container_id":"d6f53c5499a2","database_stat...
Forms             : {}
Headers           : {[Connection, keep-alive], [Content-Length, 196], [Content-Type, application/json], [Date, Wed, 03 Dec 2025
                    14:42:26 GMT]...}
Images            : {}
InputFields       : {}
Links             : {}
ParsedHtml        : mshtml.HTMLDocumentClass
RawContentLength  : 196
```

6. Screenshot Hasil Berjalan







Log di atas menunjukkan bahwa seluruh sistem **Nginx + Backend1 + Backend2 + Redis** telah berjalan dengan benar dan saling terhubung di dalam Docker Compose. Nginx berhasil memuat konfigurasi load balancer dan menerima request dari host (172.22.0.1), lalu meneruskannya bergantian ke Backend1 dan Backend2, terlihat dari log Flask yang menerima request bergantian.. Kedua backend berjalan pada alamat internal container (172.22.0.3 dan 172.22.0.4) serta merespons permintaan melalui port 5000 tanpa error. Redis juga berhasil berjalan dalam mode standalone dengan modul-modul tambahan aktif, menunjukkan bahwa database siap menerima dan menyimpan data secara persisten melalui volume yang telah dipasang.

7. Kendala dan Solusi

Kendala 1 Nginx 502 Bad Gateway

- Penyebab: backend berjalan di port 3000
- Solusi: mengubah app.run(... port=5000) sesuai dengan compose & Nginx

Kendala 2 Load balancer tidak merespons

Penyebab: salah upstream port di default.conf

Solusi: sesuaikan ke backend1:5000 dan backend2:5000

Kendala 3 Data tidak persistent

Penyebab: Redis belum memakai volume

Solusi: menambahkan
volumes:
- redis_data:/data
    - Data total_visits menjadi tetap setelah restart.

Kendala 4 Data tidak persistent

Request tidak bergantian ke Backend1/Backend2

Solusi: Memakai round_robin default pada Nginx upstream.