Panduan Server AI Ubuntu 26.04 Berbasis pemuda.ai

Open WebUI · AnythingLLM · RAG · n8n · WAHA · MCP Kali · MCP Intel · MCP Wazuh · MCP File · WhatsApp Bot · Help Desk

Onno W. Purbo
onno@indo.net.id
onno@itts.ac.id
Twitter onnowpurbo

Institut Teknologi Tangerang Selatan
OnnoCenter
2026

Lisensi: Karya ini dilisensikan di bawah **Creative Commons Attribution (CC BY) 4.0 International**. Anda bebas menggunakan, membagikan, dan mengadaptasi karya ini untuk tujuan apa pun, termasuk komersial, dengan tetap mencantumkan atribusi yang layak kepada penulis.

Disclaimer: Buku ini disusun untuk tujuan edukasi berdasarkan sumber yang dipercaya kredibel. Penulis memanfaatkan AI sebagai alat bantu penulisan, namun seluruh isi telah diverifikasi dan menjadi tanggung jawab penuh penulis. Penulis berkomitmen menjaga akurasi, presisi, dan meminimalkan halusinasi AI, serta terbuka terhadap koreksi dan masukan untuk penyempurnaan.

Daftar Isi

Daftar Isi **3**
Tool AI yang digunakan **4**
Tujuan dan batas sistem 5
Arsitektur layanan 6
Alur RAG dan pengolahan dokumen 7
Persiapan perangkat dan instalasi Ubuntu 8
Persiapan host dan jaringan 9
Instalasi Docker dan Compose 10
Ekstraksi paket dan struktur direktori 11
Konfigurasi .env 12
Uji API pemuda.ai 13
Compose Open WebUI dan PostgreSQL 15
Compose n8n dan WAHA 17
Compose AnythingLLM dan MCP 19
Startup dan akses dashboard 21
Pengaturan Open WebUI dan AnythingLLM 22
Membuat Knowledge Open WebUI 23
Konversi PDF dan OCR 24
Skrip menjalankan konversi per topik 25
Skrip upload Markdown ke RAG 26
Menjalankan uploader dan reindex 30
Instalasi dan penggunaan awal n8n 31
Instalasi WAHA dan pairing 32
WhatsApp bot: validasi pesan 33
WhatsApp bot RAG Open WebUI 34
Help desk sederhana: database 36
Help desk: workflow dan parameter SQL 37
MCP Kali: 27 tool bawaan 39
Menambah Kali menjadi 33 tool 41
MCP Intel untuk OSINT dan berita 43
MCP File dan file sharing 44
Kode MCP File 45
Instalasi MCP Wazuh 50
Environment dan service Wazuh MCP 51
Integrasi MCP ke AnythingLLM 53
Cara menggunakan MCP dan menghasilkan dokumen 55
Backup dan pemeliharaan 56
Troubleshooting dan penerimaan 57
Tentang Penulis **59**

Tool AI yang digunakan

Dokumentasi ini mengikuti konfigurasi terakhir yang diminta:

Kebutuhan	Layanan	Model
LLM Open WebUI	pemuda.ai	`demo-qwen3.7-plus`
LLM AnythingLLM	pemuda.ai	`demo-qwen3.7-plus`
Embedding RAG Open WebUI	pemuda.ai	`demo-pemuda-embedding-4b`
Penyusunan jawaban dan teks dokumen	pemuda.ai	`demo-qwen3.7-plus`

Tujuan dan batas sistem

Server ini menyediakan:

1. Percakapan AI melalui Open WebUI.
2. Workspace dan Agent melalui AnythingLLM.
3. Pencarian dokumen menggunakan RAG.
4. Penyusunan jawaban dan teks dokumen.
5. Otomasi menggunakan n8n.
6. WhatsApp bot menggunakan WAHA.
7. Help desk sederhana berbasis tiket.
8. Akses tool melalui MCP.
9. Pembacaan file sharing, Excel, DOCX, dan PDF.
10. Pembacaan informasi keamanan dari Wazuh.

Parameter API:

Base URL:
https://api.pemuda.ai/v1

API key:
sk-pmd-1234567890

Model LLM:
demo-qwen3.7-plus

Model embedding:
demo-pemuda-embedding-4b

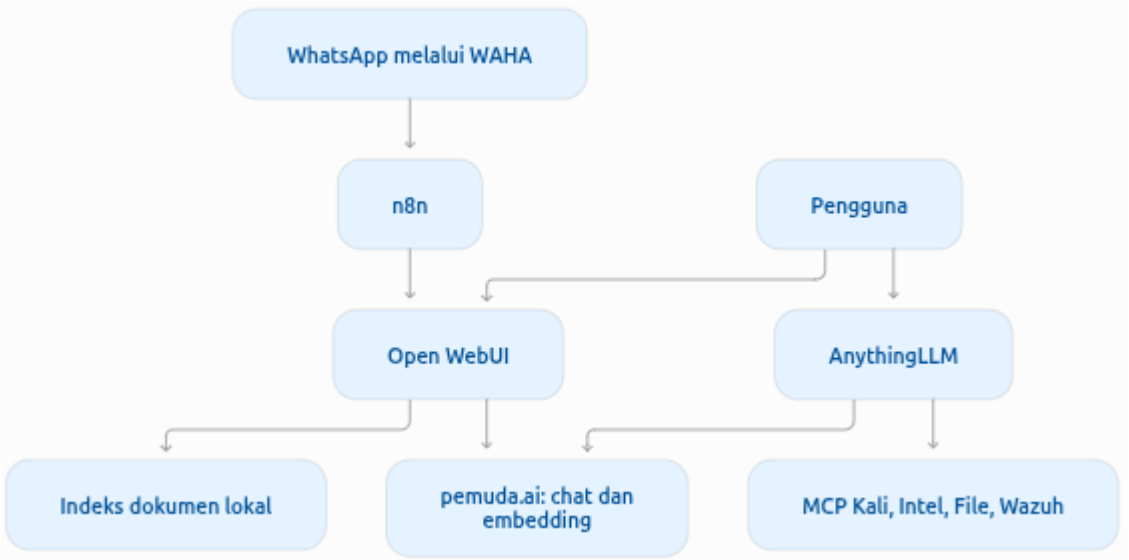
Ollama dan GPU tidak diperlukan untuk konfigurasi ini, karena generasi teks dan embedding dijalankan melalui API.

Server lokal tetap melakukan ekstraksi dokumen, OCR jika diperlukan, penyimpanan vektor, pencarian dokumen, otomasi, dan eksekusi tool.

Nama model mengikuti informasi yang Anda berikan. Keberadaan model pada akun, kompatibilitas endpoint, kuota, dan dukungan tool calling perlu dibuktikan melalui pengujian.

Pembuatan “dokumen” pada panduan berarti penyusunan teks dokumen melalui LLM. Ekspor menjadi DOCX/PDF merupakan fitur tambahan yang terpisah.

Arsitektur layanan



Pembagian peran:

Komponen	Peran
Open WebUI	Percakapan, dokumen, Knowledge, retrieval
AnythingLLM	Workspace dan Agent dengan tool MCP
pemuda.ai LLM	Generasi jawaban
pemuda.ai embedding	Pembuatan vektor dokumen dan pertanyaan
n8n	Pengaturan alur kerja
WAHA	Koneksi dan pengiriman WhatsApp
PostgreSQL	Database n8n dan help desk
MCP	Antarmuka pemanggilan tool

Tidak semua layanan perlu saling mengakses.

Contohnya:

- WAHA memanggil webhook n8n.
- n8n memanggil Open WebUI untuk RAG.
- Open WebUI memanggil API pemuda.ai.
- AnythingLLM memanggil MCP melalui jaringan internal Docker.
- MCP Wazuh mengakses Manager dan Indexer Wazuh.

Alamat `localhost` di dalam container menunjuk container itu sendiri. Gunakan nama service untuk komunikasi antar-container.

Alur RAG dan pengolahan dokumen

RAG bekerja melalui dua jalur.

Jalur pengindeksan:

> PDF
> → ekstraksi/OCR
> → Markdown per topik
> → upload Open WebUI
> → pemecahan chunk
> → embedding pemuda.ai
> → penyimpanan vektor lokal

Jalur pertanyaan:

> Pertanyaan
> → embedding pemuda.ai
> → pencarian vektor lokal
> → potongan dokumen relevan
> → LLM pemuda.ai
> → jawaban

Model yang digunakan:

> Embedding:
> `demo-pemuda-embedding-4b`
>
> Generasi:
> `demo-qwen3.7-plus`

Embedding bukan proses merangkum dokumen. Embedding menghasilkan representasi numerik untuk pencarian.

LLM tidak otomatis mengetahui semua file di server. Dokumen harus dimasukkan ke Knowledge atau diakses melalui tool yang sesuai.

> **Teks yang di embedding dan potongan dokumen yang dipakai sebagai konteks akan dikirim ke pemuda.ai.**

Indeks Open WebUI dan indeks AnythingLLM terpisah. Mengunggah dokumen ke salah satunya tidak otomatis mengunggah ke yang lain.

Jika model embedding berubah, vektor dokumen harus dibangun ulang menggunakan model baru.

Persiapan perangkat dan instalasi Ubuntu

Contoh kebutuhan awal:

Komponen	Kebutuhan praktis
OS	Ubuntu Server 26.04 LTS amd64
RAM	16 GB untuk awal; 32 GB lebih leluasa
Penyimpanan	SSD/NVMe
CPU	Beberapa core untuk OCR dan container
Internet	Dibutuhkan untuk API dan sumber publik
GPU	Tidak wajib

Unduh installer dari Ubuntu Server.

Langkah instalasi:

1. Verifikasi checksum ISO.
2. Buat USB installer.
3. Boot menggunakan UEFI.
4. Pilih disk tujuan.
5. Buat akun operator.
6. Aktifkan OpenSSH.
7. Selesaikan instalasi.
8. Reboot.

Pemilihan disk dapat menghapus isinya. Periksa model dan kapasitas disk sebelum menyetujui partisi.

Setelah login:

```
cat /etc/os-release
uname -r

lsblk -o NAME,SIZE,MODEL,FSTYPE,MOUNTPOINTS
df -h
free -h
```

Contoh IP server dalam panduan:

```
192.168.0.135
```

Ganti dengan IP sebenarnya.

Untuk server NVMe portabel, lakukan shutdown sebelum media dipindahkan:

```
sudo shutdown -h now
```

Persiapan host dan jaringan

Perbarui sistem:

```
sudo apt update
sudo apt full-upgrade -y
```

Pasang paket:

```
sudo apt install -y \
  ca-certificates curl gnupg git unzip jq \
  python3 python3-venv python3-pip \
  openssh-server ripgrep nano \
  poppler-utils \
  ocrmypdf \
  tesseract-ocr \
  tesseract-ocr-ind \
  tesseract-ocr-eng \
  cifs-utils
```

Atur zona waktu:

```
sudo timedatectl set-timezone Asia/Jakarta
timedatectl status
```

Aktifkan SSH:

```
sudo systemctl enable --now ssh
systemctl status ssh --no-pager
```

Periksa jaringan:

```
ip -br address
ip route
resolvectl status
```

Uji koneksi:

```
curl -I --connect-timeout 15 https://pemuda.ai
curl -I --connect-timeout 15 https://download.docker.com
```

Jika kernel diperbarui:

```
sudo reboot
```

Gunakan DHCP reservation jika server perlu mempunyai IP yang konsisten.

Konfigurasi dashboard pada panduan diikat ke loopback host. Operator mengaksesnya melalui SSH tunnel, sehingga tidak perlu langsung membuka port dashboard ke seluruh LAN.

Instalasi Docker dan Compose

Gunakan repository resmi Docker untuk Ubuntu. Ubuntu 26.04 tercantum pada sistem yang didukung dalam dokumentasi Docker. docs.docker.com

```
sudo install -m 0755 -d /etc/apt/keyrings

sudo curl -fsSL \
  https://download.docker.com/linux/ubuntu/gpg \
  -o /etc/apt/keyrings/docker.asc

sudo chmod a+r /etc/apt/keyrings/docker.asc
. /etc/os-release

sudo tee /etc/apt/sources.list.d/docker.sources >/dev/null <<EOF
Types: deb
URIs: https://download.docker.com/linux/ubuntu
Suites: ${UBUNTU_CODENAME:-$VERSION_CODENAME}
Components: stable
Architectures: $(dpkg --print-architecture)
Signed-By: /etc/apt/keyrings/docker.asc
EOF
sudo apt update

sudo apt install -y \
  docker-ce \
  docker-ce-cli \
  containerd.io \
  docker-buildx-plugin \
  docker-compose-plugin

sudo systemctl enable --now docker
sudo usermod -aG docker "$USER"
```

Logout dan login kembali:

```
docker run --rm hello-world
docker compose version
```

Jika Docker sudah terpasang dan bekerja, periksa versi serta sumber paket sebelum menggantinya.

Akun anggota grup Docker mempunyai kemampuan administratif terhadap host. Gunakan akun operator yang memang bertanggung jawab atas server.

Ekstraksi paket dan struktur direktori

Untuk instalasi baru:

```
sudo mkdir -p /srv/ai
sudo chown -Rf 1000:1000 /srv/ai
```

Ekstrak file zip berisi script, source code, configurasi:

```
unzip Panduan-Server-AI-Ubuntu-26.04-Berbasis-pemuda.ai.zip -d /srv/ai/

cd /srv/ai/
mv /srv/ai/ai-pemuda/* .
mv ai-pemuda/.env.example .
```

Siapkan secret:

```
/srv/ai/
sudo python3 init-env.py
```

Skrip menolak menimpa `.env` yang sudah ada.

Direktori:

```
mkdir -p \
  data/open-webui \
  data/anythingllm/plugins \
  data/postgres \
  data/n8n \
  data/waha \
  ebooks/input \
  ebooks/ocr \
  ebooks/md \
  artifacts \
  sharing \
  scripts
```

Izin baseline:

```
cd /srv/ai/
sudo chown -R 1000:1000 \
  data/n8n data/anythingllm
sudo chown 1000:1000 anythingllm.env
sudo chmod 600 anythingllm.env

sudo chmod 600 .env
```

Untuk instalasi aktif:

```
cd /srv/ai/
BACKUP_DIR="backup-config-(date + %Y%m%d − %H%M%S)"mkdir − m700"BACKUP_DIR"
cp -p compose.yml .env anythingllm.env "$BACKUP_DIR/"
```

Jangan mengekstrak ZIP menimpa konfigurasi aktif tanpa pemeriksaan.

Skrip dan konfigurasi lama yang menyebut Ollama akan disesuaikan pada bagian berikut.

Konfigurasi .env

Buka:

```
sudo chown 1000:1000 /srv/ai/.env
nano /srv/ai/.env
```

Tambahkan atau perbarui:

```
PEMUDA_BASE_URL=https://api.pemuda.ai/v1
PEMUDA_API_KEY=sk-pmd-1234567890
PEMUDA_CHAT_MODEL=demo-qwen3.7-plus
PEMUDA_EMBED_MODEL=demo-pemuda-qwen3-embedding-4b

ADMIN_EMAIL=admin@example.local

MCP_ALLOWED_HOSTS=feeds.bbci.co.uk,bbc.com,www.bbc.com,rdap.org
MCP_FEEDS=https://feeds.bbci.co.uk/news/world/rss.xml
MCP_SOCIAL_FEEDS=
MCP_DATABASE_DSN=

KALI_ALLOWED_TARGETS=192.168.0.137
KALI_TOOL_TIMEOUT=120

WA_ALLOWED_CHATS=GANTI_CHAT_ID_PENGUJI
```

Pertahankan secret yang dibuat skrip:

```
WEBUI_SECRET_KEY
POSTGRES_PASSWORD
N8N_ENCRYPTION_KEY
WAHA_API_KEY
WA_WEBHOOK_SECRET
OPENWEBUI_ADMIN_PASSWORD
ANYTHINGLLM_PASSWORD
ANYTHINGLLM_JWT_SECRET
ANYTHINGLLM_SIG_KEY
ANYTHINGLLM_SIG_SALT
WAHA_DASHBOARD_PASSWORD
```

Jangan membuat variabel aktif dengan nama sama pada beberapa baris.

Bedakan kunci:

Kunci	Fungsi
pemuda.ai	Chat dan embedding
Open WebUI	API Knowledge, upload, chat RAG
AnythingLLM	API aplikasi
WAHA	Session dan pengiriman
MCP Wazuh	Akses endpoint MCP

Kunci pemuda.ai tidak menggantikan API key Open WebUI.

Uji API pemuda.ai

Masukkan key:

```
read -rsp 'API key pemuda.ai: ' PEMUDA_API_KEY
echo
export PEMUDA_API_KEY
```

Uji chat:

```
curl --fail-with-body --silent --show-error \
  --connect-timeout 15 \
  --max-time 120 \
  https://api.pemuda.ai/v1/chat/completions \
  -H "Authorization: Bearer $PEMUDA_API_KEY" \
  -H 'Content-Type: application/json' \
  -d '{
    "model": "demo-qwen3.7-plus",
    "messages": [
      {
        "role": "user",
        "content": "Jelaskan fungsi RAG dalam dua kalimat."
      }
    ],
    "stream": false
  }' | jq
```

Uji embedding:

```
umask 077

curl --fail-with-body --silent --show-error \
  --connect-timeout 15 \
  --max-time 120 \
  https://api.pemuda.ai/v1/embeddings \
  -H "Authorization: Bearer $PEMUDA_API_KEY" \
  -H 'Content-Type: application/json' \
  -d '{
    "model": "demo-pemuda-qwen3-embedding-4b",
    "input": ["Dokumen uji RAG bahasa Indonesia."]
  }' \
  > embedding-test.json

jq -e '
  (.data | type == "array") and
  (.data | length > 0) and
  (.data[0].embedding | type == "array") and
  (.data[0].embedding | length > 0)
' embedding-test.json
jq '{
  model,
  dimensions: (.data[0].embedding | length)
}' embedding-test.json

unset PEMUDA_API_KEY
```

Jangan menebak dimensi vektor dari nama “4b”.

Jika API tidak menerima format tersebut, periksa dokumentasi akun dan body error. Penggunaan berikut mengasumsikan kompatibilitas OpenAI pada endpoint chat dan embedding.

Compose Open WebUI dan PostgreSQL

Gabungkan kode halaman 10–12 menjadi satu `/srv/ai/compose.yml`.

```
name: ai-portable

x-logging: &logging
  driver: json-file
  options:
    max-size: '10m'
    max-file: '3'

services:
  open-webui:
    image: ${OPENWEBUI_IMAGE:-ghcr.io/open-webui/open-webui:main}
    restart: unless-stopped
    ports:
      - '127.0.0.1:3000:8080'
    environment:
      ENABLE_OPENAI_API: 'true'
      OPENAI_API_BASE_URLS: ${PEMUDA_BASE_URL:?set in .env}
      OPENAI_API_KEYS: ${PEMUDA_API_KEY:?set in .env}
      DEFAULT_MODELS: ${PEMUDA_CHAT_MODEL:?set in .env}

      ENABLE_OLLAMA_API: 'false'

      RAG_EMBEDDING_ENGINE: openai
      RAG_EMBEDDING_MODEL: ${PEMUDA_EMBED_MODEL:?set in .env}
      RAG_OPENAI_API_BASE_URL: ${PEMUDA_BASE_URL:?set in .env}
      RAG_OPENAI_API_KEY: ${PEMUDA_API_KEY:?set in .env}
      RAG_EMBEDDING_BATCH_SIZE: '1'

      CHUNK_SIZE: '1000'
      CHUNK_OVERLAP: '150'
      RAG_TOP_K: '4'

      WEBUI_SECRET_KEY: ${WEBUI_SECRET_KEY:?set in .env}
      WEBUI_ADMIN_EMAIL: ${ADMIN_EMAIL:-admin@example.local}
      WEBUI_ADMIN_PASSWORD: ${OPENWEBUI_ADMIN_PASSWORD:?set in .env}
      ENABLE_SIGNUP: 'false'
      ENABLE_API_KEYS: 'true'
    volumes:
      - './data/open-webui:/app/backend/data'
    logging: *logging

  postgres:
    image: postgres:16
    restart: unless-stopped
    environment:
      POSTGRES_DB: n8n
      POSTGRES_USER: n8n
      POSTGRES_PASSWORD: ${POSTGRES_PASSWORD:?set in .env}
    volumes:
      - './data/postgres:/var/lib/postgresql/data'
    healthcheck:
      test: ['CMD-SHELL', 'pg_isready -U n8n -d n8n']
      interval: 10s
```

```
    timeout: 5s
    retries: 20
  logging: *logging
```

Engine `openai` berarti protokol OpenAI-compatible. Tujuan request tetap pemuda.ai melalui base URL.

Pengaturan embedding, batch, dan konfigurasi persisten didokumentasikan Open WebUI. docs.openwebui.com

Compose n8n dan WAHA

Lanjutkan pada file yang sama:

```
  n8n:
    image: docker.n8n.io/n8nio/n8n:1.123.9
    restart: unless-stopped
    ports:
      - '127.0.0.1:5678:5678'
    environment:
      DB_TYPE: postgresdb
      DB_POSTGRESDB_HOST: postgres
      DB_POSTGRESDB_PORT: '5432'
      DB_POSTGRESDB_DATABASE: n8n
      DB_POSTGRESDB_USER: n8n
      DB_POSTGRESDB_PASSWORD: ${POSTGRES_PASSWORD:?set in .env}
      N8N_ENCRYPTION_KEY: ${N8N_ENCRYPTION_KEY:?set in .env}

      N8N_HOST: localhost
      N8N_PORT: '5678'
      N8N_PROTOCOL: http
      N8N_EDITOR_BASE_URL: http://localhost:5678
      WEBHOOK_URL: http://localhost:5678/
      N8N_SECURE_COOKIE: 'false'
      N8N_ENFORCE_SETTINGS_FILE_PERMISSIONS: 'true'

      GENERIC_TIMEZONE: Asia/Jakarta
      TZ: Asia/Jakarta
      EXECUTIONS_DATA_PRUNE: 'true'
      EXECUTIONS_DATA_MAX_AGE: '168'

      N8N_BLOCK_ENV_ACCESS_IN_NODE: 'false'
      WAHA_API_KEY: ${WAHA_API_KEY:?set in .env}
      WA_WEBHOOK_SECRET: ${WA_WEBHOOK_SECRET:?set in .env}
      WA_ALLOWED_CHATS: ${WA_ALLOWED_CHATS:-}
    volumes:
      - './data/n8n:/home/node/.n8n'
      - './workflows:/workflows:ro'
    depends_on:
      postgres:
        condition: service_healthy
    logging: *logging

  waha:
    image: ${WAHA_IMAGE:-devlikeapro/waha:latest}
    restart: unless-stopped
    ports:
      - '127.0.0.1:3002:3000'
    environment:
      WHATSAPP_DEFAULT_ENGINE: WEBJS
      WAHA_API_KEY: ${WAHA_API_KEY:?set in .env}
      WAHA_DASHBOARD_ENABLED: 'true'
      WAHA_DASHBOARD_USERNAME: admin
      WAHA_DASHBOARD_PASSWORD: ${WAHA_DASHBOARD_PASSWORD:?set in .env}
      WHATSAPP_SWAGGER_USERNAME: admin
      WHATSAPP_SWAGGER_PASSWORD: ${WAHA_DASHBOARD_PASSWORD:?set in .env}
      WHATSAPP_RESTART_ALL_SESSIONS: 'True'
      WAHA_PRINT_QR: 'True'
```

```
volumes:
  - './data/waha:/app/.sessions'
shm_size: 1gb
logging: *logging
```

Versi n8n mengikuti baseline lampiran. Ini bukan klaim bahwa versi tersebut adalah versi produksi terbaru. Uji upgrade pada salinan data sebelum penggunaan publik.

Code node diberi akses environment untuk validasi secret webhook. Batasi editor workflow kepada operator tepercaya.

Compose AnythingLLM dan MCP

Contoh ini menyelaraskan embedding AnythingLLM ke pemuda.ai agar tidak menyisakan dependensi Ollama.

```
 anythingllm:
    image: ${ANYTHINGLLM_IMAGE:-mintplexlabs/anythingllm:latest}
    restart: unless-stopped
    ports:
      - '127.0.0.1:3001:3001'
    environment:
      STORAGE_DIR: /app/server/storage

      LLM_PROVIDER: generic-openai
      GENERIC_OPEN_AI_BASE_PATH: ${PEMUDA_BASE_URL:?set in .env}
      GENERIC_OPEN_AI_API_KEY: ${PEMUDA_API_KEY:?set in .env}
      GENERIC_OPEN_AI_MODEL_PREF: ${PEMUDA_CHAT_MODEL:?set in .env}
      GENERIC_OPEN_AI_MODEL_TOKEN_LIMIT: '4096'

      EMBEDDING_ENGINE: generic-openai
      EMBEDDING_BASE_PATH: ${PEMUDA_BASE_URL:?set in .env}
      EMBEDDING_MODEL_PREF: ${PEMUDA_EMBED_MODEL:?set in .env}
      EMBEDDING_MODEL_MAX_CHUNK_LENGTH: '1000'
      GENERIC_OPEN_AI_EMBEDDING_API_KEY: ${PEMUDA_API_KEY:?set in .env}
      GENERIC_OPEN_AI_EMBEDDING_MAX_CONCURRENT_CHUNKS: '1'

      VECTOR_DB: lancedb
      AUTH_TOKEN: ${ANYTHINGLLM_PASSWORD:?set in .env}
      JWT_SECRET: ${ANYTHINGLLM_JWT_SECRET:?set in .env}
      SIG_KEY: ${ANYTHINGLLM_SIG_KEY:?set in .env}
      SIG_SALT: ${ANYTHINGLLM_SIG_SALT:?set in .env}
    volumes:
      - './data/anythingllm:/app/server/storage'
      - './anythingllm.env:/app/server/.env'
    logging: *logging

  mcp-intel:
    build:
      context: ./mcp
      dockerfile: Dockerfile.intel
    restart: unless-stopped
    environment:
      MCP_ALLOWED_HOSTS:
${MCP_ALLOWED_HOSTS:-feeds.bbci.co.uk,bbc.com,www.bbc.com,rdap.org}
      MCP_FEEDS: ${MCP_FEEDS:-https://feeds.bbci.co.uk/news/world/rss.xml}
      MCP_SOCIAL_FEEDS: ${MCP_SOCIAL_FEEDS:-}
      MCP_DATABASE_DSN: ${MCP_DATABASE_DSN:-}
    expose: ['8000']
    logging: *logging

  kali-mcp:
    profiles: [kali]
    build:
      context: ./mcp
      dockerfile: Dockerfile.kali
    restart: unless-stopped
    environment:
```

```
      KALI_ALLOWED_TARGETS: ${KALI_ALLOWED_TARGETS:-}
      KALI_TOOL_TIMEOUT: ${KALI_TOOL_TIMEOUT:-120}
    volumes:
      - './artifacts:/work:ro'
    expose: ['8000']
    init: true
    pids_limit: 256
    cap_drop: [ALL]
    security_opt:
      - no-new-privileges:true
    logging: *logging

  mcp-files:
    build:
      context: ./mcp
      dockerfile: Dockerfile.files
    restart: unless-stopped
    environment:
      FILE_ROOT: /share
    volumes:
      - './sharing:/share:ro'
    expose: ['8000']
    init: true
    mem_limit: 1g
    cap_drop: [ALL]
    security_opt:
      - no-new-privileges:true
    logging: *logging
```

Variabel Generic OpenAI mengikuti contoh resmi AnythingLLM. GitHub

Startup dan akses dashboard

Validasi:

```
cd /srv/ai
docker compose config --quiet
```

Unduh dan jalankan aplikasi:

```
docker compose pull \
  open-webui anythingllm postgres n8n waha

docker compose up -d \
  postgres open-webui anythingllm n8n waha
docker compose ps

docker compose logs --tail=80 \
  open-webui anythingllm n8n waha
```

MCP dibangun setelah kode dan Dockerfile terkait disiapkan.

Dari komputer operator (IP server 192.168.0.135):

```
ssh \
  -L 3000:127.0.0.1:3000 \
  -L 3001:127.0.0.1:3001 \
  -L 3002:127.0.0.1:3002 \
  -L 5678:127.0.0.1:5678 \
  USER_SERVER@192.168.0.135
```

Akses:

Aplikasi	URL
Open WebUI	http://localhost:3000
AnythingLLM	http://localhost:3001
WAHA	http://localhost:3002
n8n	http://localhost:5678

Untuk server lama, jangan memakai `scripts/06-start-stack.sh` sebelum menghapus perintah yang mengharuskan Ollama.

Perubahan environment diterapkan dengan recreate container:

```
docker compose up -d --force-recreate open-webui anythingllm
```

`restart` saja tidak memuat environment Compose yang baru.

Pengaturan Open WebUI dan AnythingLLM

Open WebUI, bagian Connections:

Base URL:	https://api.pemuda.ai/v1
Model:	demo-qwen3.7-plus

Bagian Documents/RAG:

Embedding engine:	OpenAI
Base URL:	https://api.pemuda.ai/v1
Model:	demo-pemuda-embedding-4b
Batch:	1

Masukkan API key pada pengaturan terkait.

Periksa dashboard setelah migrasi, karena konfigurasi lama dapat tersimpan pada database.

AnythingLLM:

LLM Provider:	Generic OpenAI
LLM model:	demo-qwen3.7-plus
Embedding Provider:	Generic OpenAI
Embedding model:	demo-pemuda-embedding-4b

Periksa juga:

- Pengaturan workspace.
- Pengaturan Agent.
- Model khusus tugas tambahan.
- API key embedding yang dapat terpisah dari API key LLM.

Batas konteks awal `4096` pada contoh merupakan pengaturan aplikasi. Sesuaikan setelah kemampuan API diketahui.

Dukungan tool calling model belum diverifikasi. Chat biasa yang berhasil belum membuktikan model dapat menjalankan MCP.

Membuat Knowledge Open WebUI

Melalui UI:

1. Login sebagai pengguna yang mempunyai izin.
2. Buka Workspace/Knowledge.
3. Buat Knowledge baru.
4. Beri nama `Ebooks Pemuda`.
5. Isi deskripsi.
6. Simpan.

Buat API key Open WebUI melalui pengaturan akun yang tersedia pada versi aplikasi.

1. Aktifkan API key di Open WebUI

Login sebagai admin melalui `http://IP-SERVER:3000`, kemudian:

1. Buka **Settings** → **Admin** → **Authentication**.
2. Aktifkan **API Keys**.
3. Klik **Save**. Pada versi lain, pengaturan ini mungkin berada di **Admin** → **Settings** → **General**.
4. Jika **API Key Endpoint Restrictions** aktif, izinkan `/api/v1/knowledge/create`. docs.openwebui.com

2. Buat API key akun

Buka **Settings** → **Account** → **API Keys**, kemudian buat dan salin key. Jika memakai akun non-admin, admin perlu memberikan izin **API Keys** dan akses **Workspace** → **Knowledge** kepada akun tersebut

Alternatif CLI:

```
read -rsp 'API key Open WebUI: ' RAG_API_KEY
echo
export RAG_API_KEY
umask 077

curl --fail-with-body --silent --show-error \
  http://127.0.0.1:3000/api/v1/knowledge/create \
  -H "Authorization: Bearer $RAG_API_KEY" \
  -H 'Content-Type: application/json' \
  -d '{
    "name": "Ebooks Pemuda",
    "description": "Dokumen dengan embedding demo-pemuda-embedding-4b"
  }' \
  > knowledge-pemuda.json
jq '{id, name, description}' knowledge-pemuda.json
KNOWLEDGE_ID=$(jq -er '.id' knowledge-pemuda.json)
```

Simpan ID untuk uploader dan workflow n8n.

Knowledge kosong belum dapat dipakai sebagai sumber RAG. Dokumen harus diunggah, diproses, dan dihubungkan.

Endpoint internal dapat berubah antar-image. Jika respons tidak sesuai, periksa API instance yang digunakan.

Konversi PDF dan OCR

Paket menyediakan `pdf-to-topics.py` yang:

- Memakai daftar isi PDF bila tersedia.
- Memakai heuristik heading sebagai fallback.
- Memecah teks panjang.
- Mencatat sumber dan halaman.
- Membuat manifest.
- Melewati hasil yang sudah lengkap.
- Mendukung `--force`.

Siapkan environment:

```
cd /srv/ai
python3 -m venv .venv-rag
.venv-rag/bin/python -m pip install --upgrade pip
.venv-rag/bin/python -m pip install \
  -r requirements-rag.txt
.venv-rag/bin/python -m pip check
```

Salin PDF ke:

```
/srv/ai/ebooks/input
```

Konversi:

```
.venv-rag/bin/python pdf-to-topics.py \
  ebooks/input \
  ebooks/md \
  --max-chars 7000
```

OCR untuk PDF scan:

```
ocrmypdf \
  --skip-text \
  -l ind+eng \
  ebooks/input/buku.pdf \
  ebooks/ocr/buku.pdf
```

Lalu konversi folder OCR:

```
.venv-rag/bin/python pdf-to-topics.py \
  ebooks/ocr \
  ebooks/md \
  --max-chars 7000
```

Jangan memproses PDF asli dan salinan OCR sebagai dua sumber tanpa pengelolaan duplikasi.

Topik yang dihasilkan perlu diperiksa. Struktur PDF yang buruk dapat menghasilkan pembagian yang kurang tepat.

Skrip menjalankan konversi per topik

Buat `/srv/ai/scripts/convert-ebooks.sh`:

```
#!/usr/bin/env bash
set -Eeuo pipefail

cd /srv/ai

INPUT_DIR="1:- ebooks/input"OUTPUT_DIR = "{2:-ebooks/md}"

if [[ ! -d "$INPUT_DIR" ]]; then
  echo "Direktori input tidak ditemukan:
INPUT_DIR" > &2exit1fiif[[! - x.venv - rag/bin/python]]; thenecho"Virtualenv.venv - ragbelumtersedia." > &2exit
OUTPUT_DIR"

.venv-rag/bin/python pdf-to-topics.py \
  "$INPUT_DIR" \
  "$OUTPUT_DIR" \
  --max-chars 7000
```

Jalankan:

```
bash scripts/convert-ebooks.sh \
  ebooks/input \
  ebooks/md
```

Untuk OCR:

```
bash scripts/convert-ebooks.sh \
  ebooks/ocr \
  ebooks/md
```

Periksa hasil:

```
rg --files ebooks/md
```

Metadata penting:

```
source_pdf
document_sha256
topic
page_start
page_end
chunk_id
```

Batas `7000` mengacu pada bagian Markdown hasil konversi. Open WebUI masih melakukan chunking pada tahap indexing.

Jika ingin mengubah algoritma konversi, catat versinya dan evaluasi ulang identitas hasil agar state lama tidak dianggap selalu cocok.

Skrip upload Markdown ke RAG

Kode berikut merupakan uploader khusus Open WebUI. Simpan sebagai:

```
/srv/ai/upload-md-openwebui.py
```

Uploader ini dapat dipakai sebagai alternatif skrip `upload-rag.py` yang sudah tersedia pada ZIP.

```
#!/usr/bin/env python3
import argparse
import hashlib
import json
import os
import time
import uuid
from pathlib import Path
from urllib.error import HTTPError
from urllib.parse import quote
from urllib.request import Request, urlopen


def api(url, key, method="GET", data=None, file=None):
    headers = {
        "Authorization": "Bearer " + key,
        "Accept": "application/json",
    }
    body = None

    if file is not None:
        content = file.read_bytes()
        digest = hashlib.sha256(content).hexdigest()
        boundary = "----rag" + uuid.uuid4().hex

        body = (
            f"--{boundary}\r\n"
            f'Content-Disposition: form-data; name="file"; '
            f'filename="{digest[:24]}.md"\r\n'
            "Content-Type: text/markdown\r\n\r\n"
        ).encode() + content + f"\r\n--{boundary}--\r\n".encode()

        headers["Content-Type"] = (
            "multipart/form-data; boundary=" + boundary
        )

    elif data is not None:
        body = json.dumps(data).encode()
        headers["Content-Type"] = "application/json"

    request = Request(
        url,
        data=body,
        headers=headers,
        method=method,
    )

    try:
```

```
        with urlopen(request, timeout=900) as response:
            raw = response.read()
            result = json.loads(raw) if raw else {}
    except HTTPError as error:
        message = error.read().decode(errors="replace")[:1500]
        raise RuntimeError(
            f"HTTP {error.code}: {message}"
        ) from error

    if isinstance(result, dict):
        if result.get("success") is False or result.get("error"):
            raise RuntimeError(str(result))

    return result


def save_state(path, state):
    path.parent.mkdir(parents=True, exist_ok=True)
    temporary = path.with_name(path.name + ".tmp")
    temporary.write_text(
        json.dumps(state, indent=2),
        encoding="utf-8",
    )
    temporary.replace(path)


def main():
    parser = argparse.ArgumentParser()
    parser.add_argument("directory", type=Path)
    parser.add_argument("--url", required=True)
    parser.add_argument("--knowledge", required=True)
    parser.add_argument(
        "--state",
        type=Path,
        default=Path("upload-pemuda-state.json"),
    )
    parser.add_argument(
        "--embedding-model",
        default="demo-pemuda-embedding-4b",
    )
    parser.add_argument("--processing-timeout", type=int, default=900)
    args = parser.parse_args()

    key = os.environ.get("RAG_API_KEY")
    if not key:
        raise SystemExit("Set RAG_API_KEY terlebih dahulu.")

    root = args.directory.resolve()
    files = sorted(
        path for path in root.rglob("*.md")
        if path.name != "index.md"
    )

    if not files:
        raise SystemExit("Tidak menemukan file Markdown.")

    base = args.url.rstrip("/")
    knowledge = quote(args.knowledge, safe="")
```

```
state = (
    json.loads(args.state.read_text())
    if args.state.exists()
    else {}
)

scope = (
    f"{base}|{args.knowledge}|{args.embedding_model}"
)
entries = state.setdefault(scope, {})
failures = []

for path in files:
    checksum = hashlib.sha256(path.read_bytes()).hexdigest()
    identity = f"{path.relative_to(root)}|{checksum}"
    entry = entries.setdefault(identity, {})

    if entry.get("done"):
        print("SKIP", path)
        continue

    try:
        if not entry.get("file_id"):
            uploaded = api(
                base + "/api/v1/files/",
                key,
                method="POST",
                file=path,
            )
            entry["file_id"] = uploaded["id"]
            save_state(args.state, state)

        file_id = quote(entry["file_id"], safe="")
        deadline = time.monotonic() + args.processing_timeout

        while True:
            result = api(
                base
                + f"/api/v1/files/{file_id}/process/status",
                key,
            )

            status = result.get("status")

            if status == "completed":
                break

            if status in {"failed", "error"}:
                raise RuntimeError(str(result))

            if time.monotonic() > deadline:
                raise TimeoutError(
                    "Pemrosesan belum selesai; jalankan lagi."
                )

            time.sleep(2)

        api(
            base
```

```
                + f"/api/v1/knowledge/{knowledge}/file/add",
                key,
                method="POST",
                data={"file_id": entry["file_id"]},
            )

            entry["done"] = True
            save_state(args.state, state)
            print("OK", path)

        except Exception as error:
            failures.append({
                "file": str(path),
                "error": str(error),
            })
            print("ERROR", path, error)

    if failures:
        error_path = args.state.with_name(
            args.state.stem + "-errors.json"
        )
        error_path.write_text(
            json.dumps(failures, indent=2),
            encoding="utf-8",
        )
        raise SystemExit(1)


if __name__ == "__main__":
    main()
```

Skrip ini menyimpan file ID sebelum proses berikutnya agar pengulangan tidak selalu mengunggah file dari awal.

Menjalankan uploader dan reindex

Jalankan:

```
cd /srv/ai

read -rsp 'API key Open WebUI: ' RAG_API_KEY
echo
export RAG_API_KEY

read -rp 'Knowledge ID: ' KNOWLEDGE_ID
python3 upload-md-openwebui.py ebooks/md \
  --url http://127.0.0.1:3000 \
  --knowledge "$KNOWLEDGE_ID" \
  --embedding-model demo-pemuda-embedding-4b \
  --state upload-pemuda-state.json
unset RAG_API_KEY
```

Status:

Status	Makna
OK	Pemrosesan dan pengaitan selesai
SKIP	State lokal mencatat selesai
ERROR	Periksa error dan log

Batas uploader:

- Jangan menjalankan dua proses pada state yang sama.
- State lokal tidak membuktikan file masih ada di server.
- Skrip tidak menghapus versi dokumen lama.
- Kegagalan setelah server sukses tetapi sebelum checkpoint tersimpan masih dapat menyebabkan pengulangan.

Migrasi embedding lama memerlukan reindex.

Pilihan yang terpisah dari data lama:

```
mkdir -p data/open-webui-pemuda
```

Ubah volume:

```
volumes:
  - './data/open-webui-pemuda:/app/backend/data'
```

Kemudian recreate, buat akun/API key/Knowledge baru, dan upload ulang.

Data lama tetap berada pada direktori sebelumnya. Jangan menghapusnya sampai hasil migrasi diterima.

Instalasi dan penggunaan awal n8n

Buka:

```
http://localhost:5678
```

Buat akun owner melalui UI.

Pada instalasi baru dengan baseline paket:

```
python3 bootstrap-accounts.py n8n
```

Buat credentials:

Nama contoh	Jenis
Pemuda LLM	Header Auth
OpenWebUI RAG	Header Auth
WAHA Internal	Header Auth
Helpdesk PostgreSQL	PostgreSQL

Untuk Open WebUI:

Header: Authorization
Value: Bearer API_KEY_OPENWEBUI

Untuk WAHA:

Header: X-Api-Key
Value: API_KEY_WAHA

Webhook produksi:

```
http://n8n:5678/webhook/wa-in
```

Webhook pengujian:

```
http://n8n:5678/webhook-test/wa-in
```

Agar terus berjalan:

1. Simpan workflow.
2. Aktifkan workflow.
3. Gunakan webhook produksi.
4. Pantau Executions.

Pertahankan `N8N_ENCRYPTION_KEY`. Menggantinya tanpa migrasi dapat membuat credentials lama tidak terbaca.

Instalasi WAHA dan pairing

Jalankan:

```
docker compose up -d waha
docker compose logs --tail=80 waha
```

Buka:

```
http://localhost:3002
```

Konfigurasi session menggunakan skrip ZIP:

```
python3 configure-waha.py
```

Skrip mengatur:

```
Session: default
Webhook: http://n8n:5678/webhook/wa-in
Event: message
Header: X-WA-Webhook-Secret
```

Pairing:

1. Buka session `default`.
2. Start session.
3. Tampilkan QR.
4. Buka perangkat tertaut pada WhatsApp.
5. Pindai QR.
6. Tunggu status siap.

WAHA menggunakan session untuk mengelola koneksi WhatsApp. waha.devlike.pro

Data pairing tersimpan pada:

```
/srv/ai/data/waha
```

Jangan menghapusnya sebagai langkah umum memperbaiki webhook.

Periksa format payload nyata sebelum menyesuaikan filter bot. Chat ID dan message ID harus diambil dari event yang diterima, bukan ditebak.

WhatsApp bot: validasi pesan

Workflow:

WA In → Validasi pesan → Routing → Layanan → Kirim WA

Webhook:

```
Method: POST
Path: wa-in
Response: On Received
```

Code node `Validasi pesan`:

```
const request = $input.first().json;
const body = request.body || {};
const payload = body.payload || {};

if (
  request.headers?.['x-wa-webhook-secret']
  !==
env.WA_WEBHOOK_SECRET)return[];if(body.event!== 'message')return[];if(payload.fromMe === true)return[];con
env.WA_ALLOWED_CHATS || '')
    .split(',')
    .map(value => value.trim())
    .filter(Boolean)
);

if (!allowed.has(chatId)) return [];

const text = String(payload.body || '').trim();
if (!text || text.length > 4000) return [];

const rawId = payload.id;
const messageId =
  typeof rawId === 'string'
    ? rawId
    : rawId?._serialized || rawId?.id;

if (!messageId) return [];

return [{
  json: {
    session: body.session || 'default',
    chatId,
    messageId: String(messageId),
    text
  }
}];
```

Mode Code node: **Run Once for All Items**.

Setelah mengubah allowlist:

```
docker compose up -d --force-recreate n8n
```

Pesan di luar daftar penguji tidak diproses.

WhatsApp bot RAG Open WebUI

HTTP Request node:

```
Name: Tanya RAG
Method: POST
URL: http://open-webui:8080/api/chat/completions
Credential: OpenWebUI RAG
Timeout: 120000 ms
```

Body expression:

```
={{
  {
    model: "demo-qwen3.7-plus",
    stream: false,
    messages: [
      {
        role: "system",
        content:
          "Jawab dalam bahasa Indonesia berdasarkan dokumen. " +
          "Jika informasi tidak ditemukan, nyatakan dengan jelas. " +
          "Jangan mengarang sumber."
      },
      {
        role: "user",
        content: $json.text
      }
    ],
    files: [
      {
        type: "collection",
        id: "GANTI_KNOWLEDGE_ID"
      }
    ]
  }
}}
```

Node kirim WhatsApp:

```
Method: POST
URL: http://waha:3000/api/sendText
Credential: WAHA Internal
```

Body:

```
={{
  {
    session: $("Validasi pesan").first().json.session,
    chatId: $("Validasi pesan").first().json.chatId,
    text: String(
      $json.choices?.[0]?.message?.content
      || "Jawaban belum tersedia. Silakan hubungi petugas."
    ).slice(0, 8000)
  }
}}
```

Gunakan error branch bila request gagal.

Memanggil pemuda.ai langsung hanya menjalankan LLM. Memanggil Open WebUI dengan referensi Knowledge memungkinkan retrieval RAG dijalankan terlebih dahulu.

Help desk sederhana: database

Contoh perintah:

```
/tiket Printer lantai dua tidak dapat digunakan
/status 123
```

Buat role dan database:

```
docker compose exec postgres \
  psql -U n8n -d postgres
CREATE ROLE helpdesk LOGIN;
```

Atur password:

```
\password helpdesk
CREATE DATABASE helpdesk OWNER helpdesk;
\q
```

Masuk:

```
docker compose exec postgres \
  psql -h 127.0.0.1 -U helpdesk -d helpdesk -W
```

Buat tabel:

```
CREATE TABLE tickets (
    id BIGSERIAL PRIMARY KEY,
    wa_message_id TEXT UNIQUE NOT NULL,
    requester TEXT NOT NULL,
    description TEXT NOT NULL,
    category TEXT NOT NULL DEFAULT 'umum',
    status TEXT NOT NULL DEFAULT 'open',
    created_at TIMESTAMPTZ NOT NULL DEFAULT now(),
    updated_at TIMESTAMPTZ NOT NULL DEFAULT now(),
    CHECK (status IN ('open', 'in_progress', 'resolved'))
);

CREATE INDEX tickets_requester_created_idx
ON tickets (requester, created_at DESC);
```

Credential PostgreSQL n8n:

```
Host: postgres
Port: 5432
Database: helpdesk
User: helpdesk
Password: password yang dibuat
```

Database tiket dipisahkan dari tabel internal n8n.

Help desk: workflow dan parameter SQL

Code node:

```
const input = $input.first().json;
const text = input.text.trim();

if (/^\/tiket\s+/i.test(text)) {
  const description = text.replace(/^\/tiket\s+/i, '').trim();
  if (!description) return [];

  return [{
    json: {
      ...input,
      action: "create",
      description: description.slice(0, 4000)
    }
  }];
}

const match = text.match(/^\/status\s+([1-9][0-9]{0,12})$/i);

if (match) {
  return [{
    json: {
      ...input,
      action: "status",
      ticketId: match[1]
    }
  }];
}

return [];
```

Switch berdasarkan `action`.

Query tiket:

```
INSERT INTO tickets (
    wa_message_id,
    requester,
    description
)
VALUES ($1, $2, $3)
ON CONFLICT (wa_message_id)
DO UPDATE SET wa_message_id = EXCLUDED.wa_message_id
RETURNING id, requester, status, created_at;
```

Query Parameters:

```
{{ [$json.messageId, $json.chatId, $json.description] }}
```

Query status:

```
SELECT COALESCE(
    (
      SELECT 'Tiket #' || id::text || ': ' || status
      FROM tickets
      WHERE id = $1::bigint
        AND requester = $2
    ),
    'Tiket tidak ditemukan untuk nomor Anda.'
) AS reply;
```

Parameter:

```
{{ [$json.ticketId, $json.chatId] }}
```

AI dapat menyarankan kategori atau merangkum masalah. Nomor tiket dan status tetap berasal dari database.

Kunci unik mencegah tiket ganda. Pengiriman balasan tepat satu kali memerlukan pengelolaan outbox/retry tersendiri.

MCP Kali: 27 tool bawaan

Hasil pemeriksaan `kali.py` pada ZIP menunjukkan 27 tool:

No.	Tool	Fungsi
1	nmap_connect	Port TCP
2	dns_lookup	DNS A
3	whois_lookup	WHOIS
4	tool_inventory	Inventaris executable
5	dns_records	DNS beberapa tipe
6	host_lookup	DNS lookup
7	dns_recon	Enumerasi DNS standar
8	nmap_service_scan	Deteksi layanan
9	tcp_connect	Satu port TCP
10	trace_route	Jalur jaringan
11	http_headers	Header HTTP
12	http_fetch	Isi HTTP
13	whatweb_scan	Teknologi web
14	waf_detect	Deteksi WAF
15	nikto_scan	Audit web
16	ssl_scan	TLS
17	tls_protocols	Protokol TLS
18	file_identify	Jenis file
19	file_metadata	Metadata
20	file_strings	String file
21	elf_headers	Header ELF
22	pdf_metadata	Metadata PDF
23	dirb_scan	Direktori web
24	metasploit_version	Versi framework
25	metasploit_module_info	Informasi modul

26	metasploit_scan	Scanner versi
27	sqlmap_scan	Pengujian SQL injection lab

Ini satu server MCP dengan banyak tool.

Allowlist:

```
KALI_ALLOWED_TARGETS=192.168.0.137,lab.example.org
```

Kode memakai pencocokan host/IP persis. CIDR tidak otomatis berarti seluruh subnet diizinkan.

Menambah Kali menjadi 33 tool

Tambahkan sebelum baris startup `if __name__...` pada `mcp/kali.py`:

```
@mcp.tool()
def dns_reverse(target: str) -> dict:
    """Reverse DNS untuk IP dalam allowlist."""
    value = target_ok(target)
    ipaddress.ip_address(value)
    return run(["dig", "+time=5", "+tries=1", "-x", value])


@mcp.tool()
def http_options(url: str) -> dict:
    """Respons OPTIONS tanpa mengikuti redirect."""
    return run([
        "curl", "--noproxy", "*",
        "--silent", "--show-error",
        "--max-time", "20",
        "-X", "OPTIONS", "-i",
        url_ok(url)
    ])


@mcp.tool()
def robots_txt(base_url: str) -> dict:
    """Baca robots.txt pada host berizin."""
    u = urlsplit(url_ok(base_url))
    url = f"{u.scheme}://{u.netloc}/robots.txt"
    return run([
        "curl", "--noproxy", "*",
        "--silent", "--show-error",
        "--max-time", "20", url
    ])


@mcp.tool()
def security_txt(base_url: str) -> dict:
    """Baca security.txt pada lokasi well-known."""
    u = urlsplit(url_ok(base_url))
    url = f"{u.scheme}://{u.netloc}/.well-known/security.txt"
    return run([
        "curl", "--noproxy", "*",
        "--silent", "--show-error",
        "--max-time", "20", url
    ])


@mcp.tool()
def tls_certificate(target: str, port: int = 443) -> dict:
    """Baca sertifikat dengan verifikasi TLS aktif."""
    import ssl

    host = target_ok(target)
    port_ok(port)
    context = ssl.create_default_context()
```

```
    with socket.create_connection((host, port), timeout=10) as raw:
        with context.wrap_socket(raw, server_hostname=host) as connection:
            return {
                "target": host,
                "protocol": connection.version(),
                "certificate": connection.getpeercert()
            }


@mcp.tool()
def file_sha256(filename: str) -> dict:
    """SHA-256 file dalam /work."""
    import hashlib

    path = Path(artifact_ok(filename))
    digest = hashlib.sha256()

    with path.open("rb") as stream:
        for block in iter(lambda: stream.read(1024 * 1024), b""):
            digest.update(block)

    return {"file": filename, "sha256": digest.hexdigest()}
```

Perluas paket APT Dockerfile Kali:

```
RUN apt-get update && apt-get install -y --no-install-recommends \
    python3 python3-venv ca-certificates \
    nmap dnsutils whois curl openssl \
    netcat-openbsd traceroute \
    whatweb wafw00f dnsrecon sslscan testssl.sh \
    file libimage-exiftool-perl binutils poppler-utils jq \
    dirb nikto sqlmap metasploit-framework \
    && rm -rf /var/lib/apt/lists/*
```

Build:

```
python3 -m py_compile mcp/kali.py

docker compose --profile kali build kali-mcp
docker compose --profile kali up -d kali-mcp
```

Periksa nama executable `testssl` dan sesuaikan wrapper jika paket menyediakan nama berbeda.

MCP Intel untuk OSINT dan berita

Tool bawaan:

```
domain_osint
scrape_web
news_intelligence
breaking_news
social_intelligence
database_query
```

Build:

```
docker compose build mcp-intel
docker compose up -d mcp-intel
```

Pengaturan:

```
MCP_ALLOWED_HOSTS=feeds.bbci.co.uk,bbc.com,www.bbc.com,rdap.org
MCP_FEEDS=https://feeds.bbci.co.uk/news/world/rss.xml
MCP_SOCIAL_FEEDS=
```

Untuk menambah media Indonesia:

1. Temukan RSS resmi yang masih aktif.
2. Uji responsnya.
3. Masukkan URL feed.
4. Tambahkan hostname feed pada allowlist.
5. Tambahkan hostname artikel bila ingin scraping artikel.

Kode paket:

- Hanya HTTPS publik.
- Tidak mengikuti redirect.
- Membatasi ukuran unduhan.
- Tidak menjalankan JavaScript.
- Tidak melewati login/paywall.

`breaking_news` merupakan snapshot RSS, bukan verifikasi editorial.

Media sosial terbatas pada feed publik yang tersedia dan dikonfigurasi. Pencarian menyeluruh platform sosial memerlukan API atau konektor tersendiri.

Jika SDK MCP pada Dockerfile tidak cocok dengan import kode, perbarui SDK secara terukur, jalankan `pip check`, dan catat versi yang berhasil. Jangan menyamakan sukses build dengan sukses pemanggilan tool.

MCP File dan file sharing

Folder lokal:

```
mkdir -p /srv/ai/sharing
```

Berikan izin baca kepada UID container `10001`.

Untuk SMB:

```
sudo mkdir -p /mnt/ai-share
sudo nano /etc/ai-share.credentials
username=USER_SHARING
password=PASSWORD_SHARING
domain=WORKGROUP
sudo chmod 600 /etc/ai-share.credentials
sudo mount -t cifs \
  //192.168.0.10/dokumen \
  /mnt/ai-share \
  -o
credentials=/etc/ai-share.credentials,vers=3.0,ro,uid=10001,gid=10001,file_m
ode=0440,dir_mode=0550,nosuid,nodev,noexec
```

Periksa:

```
mountpoint /mnt/ai-share
ls -la /mnt/ai-share
```

Volume MCP File jika menggunakan SMB:

```
volumes:
  - '/mnt/ai-share:/share:ro'
```

Mount harus tersedia sebelum container dibuat. Recreate jika mount baru ditambahkan.

Format yang didukung contoh berikut:

```
TXT, MD, CSV, TSV, LOG
XLSX
DOCX
PDF teks
```

Excel tidak dihitung ulang dan PDF scan tidak di-OCR oleh MCP ini.

Kode MCP File

Buat `/srv/ai/mcp/files.py`:

```
import os
import zipfile
from pathlib import Path

from mcp.server.fastmcp import FastMCP
from mcp.server.transport_security import TransportSecuritySettings
from openpyxl import load_workbook
from docx import Document
from pypdf import PdfReader

ROOT = Path(os.getenv("FILE_ROOT", "/share")).resolve()
MAX_FILE = 32 * 1024 * 1024
MAX_TEXT = 24000

mcp = FastMCP(
    "Read Only Files",
    host="0.0.0.0",
    port=8000,
    stateless_http=True,
    transport_security=TransportSecuritySettings(
        enable_dns_rebinding_protection=False
    ),
)


def checked(relative: str, directory=False) -> Path:
    if Path(relative).is_absolute():
        raise ValueError("Gunakan path relatif.")

    path = (ROOT / relative).resolve()

    if not path.is_relative_to(ROOT):
        raise ValueError("Path di luar sharing.")

    if directory:
        if not path.is_dir():
            raise ValueError("Direktori tidak ditemukan.")
    else:
        if not path.is_file():
            raise ValueError("File tidak ditemukan.")
        if path.stat().st_size > MAX_FILE:
            raise ValueError("File melebihi 32 MB.")

    return path


def office(relative: str, suffix: str) -> Path:
    path = checked(relative)

    if path.suffix.lower() != suffix:
        raise ValueError("Format tidak sesuai.")

    with zipfile.ZipFile(path) as archive:
```

```
        size = sum(item.file_size for item in archive.infolist())
        if size > 100 * 1024 * 1024:
            raise ValueError("Isi Office melebihi batas.")

    return path


@mcp.tool()
def list_files(directory: str = ".") -> dict:
    """Daftar maksimal 200 file/folder."""
    path = checked(directory, directory=True)
    items = []

    for item in sorted(path.iterdir(), key=lambda value: value.name):
        if item.is_symlink():
            continue

        items.append({
            "name": item.name,
            "directory": item.is_dir()
        })

        if len(items) > 200:
            break

    return {
        "items": items[:200],
        "truncated": len(items) > 200
    }


@mcp.tool()
def read_text(filename: str) -> dict:
    """Baca TXT, MD, CSV, TSV atau log."""
    path = checked(filename)

    if path.suffix.lower() not in {".txt", ".md", ".csv", ".tsv", ".log"}:
        raise ValueError("Ekstensi tidak didukung.")

    with path.open("r", encoding="utf-8", errors="replace") as stream:
        text = stream.read(MAX_TEXT + 1)

    return {
        "text": text[:MAX_TEXT],
        "truncated": len(text) > MAX_TEXT
    }


@mcp.tool()
def read_excel(
    filename: str,
    sheet: str = "",
    start_row: int = 1,
    max_rows: int = 30
) -> dict:
    """Baca jendela XLSX; maksimal 100 baris dan 30 kolom."""
    if start_row < 1:
        raise ValueError("start_row minimal 1.")
```

```
    workbook = load_workbook(
        office(filename, ".xlsx"),
        read_only=True,
        data_only=True,
    )

    try:
        worksheet = workbook[sheet] if sheet else workbook.worksheets[0]
        count = max(1, min(max_rows, 100))
        rows = []
        size = 0
        truncated = False

        for row in worksheet.iter_rows(
            min_row=start_row,
            max_row=start_row + count - 1,
            max_col=30,
            values_only=True,
        ):
            values = [None if v is None else str(v)[:1000] for v in row]
            added = sum(len(v or "") for v in values)

            if size + added > MAX_TEXT:
                truncated = True
                break

            rows.append(values)
            size += added

        return {
            "sheets": workbook.sheetnames,
            "sheet": worksheet.title,
            "start_row": start_row,
            "rows": rows,
            "truncated": truncated,
            "note": "Rumus memakai cache; tidak dihitung ulang."
        }
    finally:
        workbook.close()


@mcp.tool()
def read_docx(filename: str) -> dict:
    """Baca paragraf dan tabel utama DOCX."""
    document = Document(office(filename, ".docx"))
    parts = []
    used = 0
    truncated = False

    def add(text):
        nonlocal used, truncated
        if used + len(text) > MAX_TEXT:
            parts.append(text[:max(0, MAX_TEXT - used)])
            truncated = True
            return False
        parts.append(text)
        used += len(text)
        return True
```

```
    for paragraph in document.paragraphs:
        if not add(paragraph.text):
            break

    if not truncated:
        for table in document.tables:
            for row in table.rows:
                if not add(" | ".join(cell.text for cell in row.cells)):
                    break
            if truncated:
                break

    return {"text": "\n".join(parts), "truncated": truncated}


@mcp.tool()
def read_pdf(
    filename: str,
    start_page: int = 1,
    max_pages: int = 5
) -> dict:
    """Baca maksimal 10 halaman PDF teks."""
    path = checked(filename)

    if path.suffix.lower() != ".pdf":
        raise ValueError("File harus PDF.")
    if start_page < 1:
        raise ValueError("start_page minimal 1.")

    reader = PdfReader(path)

    if reader.is_encrypted:
        raise ValueError("PDF terenkripsi belum didukung.")

    stop = min(
        len(reader.pages),
        start_page - 1 + min(max(max_pages, 1), 10),
    )

    text = "\n\n".join(
        f"[Halaman {i + 1}]\n"
        + (reader.pages[i].extract_text() or "")
        for i in range(start_page - 1, stop)
    )

    return {
        "total_pages": len(reader.pages),
        "text": text[:MAX_TEXT],
        "truncated": len(text) > MAX_TEXT
    }


if __name__ == "__main__":
    mcp.run(transport="streamable-http")
```

Dockerfile:

```
FROM python:3.12-slim

WORKDIR /app

RUN pip install --no-cache-dir \
    'mcp>=1.12,<2' \
    'openpyxl>=3.1,<4' \
    'python-docx>=1.1,<2' \
    'pypdf>=5,<7' \
    && pip check

RUN useradd -m -u 10001 mcpuser

COPY files.py check-mcp.py /app/

USER mcpuser

CMD ["python", "/app/files.py"]
```

Simpan sebagai `mcp/Dockerfile.files`, lalu:

```
docker compose build mcp-files
docker compose up -d mcp-files
```

Instalasi MCP Wazuh

Alur:

> Wazuh Agent/sensor
> → Wazuh Manager dan Indexer
> → MCP Wazuh
> → AnythingLLM

MCP tidak menggantikan instalasi sensor atau Agent.

Repository:

> INFOKOM-KI/Wazuh-MCP-Server

ZIP sudah memuat salinannya. Jika belum ada:

```
cd /srv/ai

git clone \
  https://github.com/INFOKOM-KI/Wazuh-MCP-Server.git
cd /srv/ai/Wazuh-MCP-Server
git status
git rev-parse HEAD > /srv/ai/wazuh-mcp-commit.txt
```

Instalasi mengikuti baseline:

```
sudo bash setup.sh
```

Periksa:

```
sudo test -x /opt/blue-team-mcp/venv/bin/python3
sudo test -f /opt/blue-team-mcp/main.py
```

Entry point yang diperiksa mendukung transport `streamable_http`, host, dan port. github.com

Gunakan akun Wazuh khusus dengan hak baca.

Alamat Manager, Indexer, dan host AI dapat berbeda. Isi alamat sebenarnya pada konfigurasi berikut.

Environment dan service Wazuh MCP

Buat user:

```
id blueteam-mcp >/dev/null 2>&1 || \
  sudo useradd --system --user-group \
    --home-dir /opt/blue-team-mcp \
    --shell /usr/sbin/nologin \
    blueteam-mcp
sudo install -d \
  -o blueteam-mcp \
  -g blueteam-mcp \
  -m 750 \
  /var/log/blue-team-mcp
```

Buat API key:

```
python3 -c \
  "import secrets; print('btm_' + secrets.token_urlsafe(32))"
```

Isi `/etc/wazuh-mcp.env`:

```
WAZUH_INDEXER_URL=https://IP_INDEXER:9200
WAZUH_INDEXER_USER=USER_INDEXER_READONLY
WAZUH_INDEXER_PASSWORD="PASSWORD_INDEXER"

WAZUH_API_URL=https://IP_MANAGER:55000
WAZUH_API_USER=USER_MANAGER_API
WAZUH_API_PASSWORD="PASSWORD_MANAGER_API"

WAZUH_INDEXER_VERIFY_SSL=true
WAZUH_API_VERIFY_SSL=true

WAZUH_READ_ONLY=true
MCP_API_KEY=GANTI_KEY_BTM
MCP_API_KEY_SCOPES=wazuh:read

BLUETEAM_RERANK_ENABLED=false
BLUETEAM_RAG_ENABLED=false
BLUETEAM_AUDIT_LOG=/var/log/blue-team-mcp/audit.log

OMP_NUM_THREADS=2
MKL_NUM_THREADS=2
OPENBLAS_NUM_THREADS=2
TOKENIZERS_PARALLELISM=false
sudo chmod 600 /etc/wazuh-mcp.env
```

Buat `/etc/systemd/system/wazuh-mcp.service`:

```
[Unit]
Description=Wazuh MCP for AnythingLLM
Wants=network-online.target
After=network-online.target

[Service]
Type=simple
User=blueteam-mcp
Group=blueteam-mcp
WorkingDirectory=/opt/blue-team-mcp
EnvironmentFile=/etc/wazuh-mcp.env
Environment=PYTHONUNBUFFERED=1
ExecStart=/opt/blue-team-mcp/venv/bin/python3 /opt/blue-team-mcp/main.py
--transport streamable_http --host 192.168.0.135 --port 8001
Restart=on-failure
RestartSec=5
UMask=0077

[Install]
WantedBy=multi-user.target
sudo systemctl daemon-reload
sudo systemctl enable --now wazuh-mcp
sudo journalctl -u wazuh-mcp -n 80 --no-pager
```

Ganti IP host sesuai lingkungan dan gunakan CA yang dipercaya.

Integrasi MCP ke AnythingLLM

File:

/srv/ai/data/anythingllm/plugins/anythingllm_mcp_servers.json

Alamat MCP harus dapat dijangkau dari container AnythingLLM. docs.anythingllm.com

Gunakan skrip merge:

```
cd /srv/ai

sudo python3 - <<'PY'
import json
import os
import shutil
from datetime import datetime
from getpass import getpass
from pathlib import Path

path = Path(
    "/srv/ai/data/anythingllm/plugins/"
    "anythingllm_mcp_servers.json"
)
path.parent.mkdir(parents=True, exist_ok=True)

config = json.loads(path.read_text()) if path.exists() else {}
servers = config.setdefault("mcpServers", {})

if not isinstance(servers, dict):
    raise SystemExit("mcpServers harus berupa objek JSON.")

if path.exists():
    backup = path.with_name(
        path.name + ".bak-" + datetime.now().strftime("%Y%m%d-%H%M%S")
    )
    shutil.copy2(path, backup)
    os.chmod(backup, 0o600)

servers.update({
    "intel": {
        "type": "streamable",
        "url": "http://mcp-intel:8000/mcp"
    },
    "kali-lab": {
        "type": "streamable",
        "url": "http://kali-mcp:8000/mcp"
    },
    "files": {
        "type": "streamable",
        "url": "http://mcp-files:8000/mcp"
    }
})

key = getpass("API key Wazuh MCP; kosongkan jika belum siap: ")

if key:
```

```
    servers["wazuh-mcp"] = {
        "type": "streamable",
        "url": "http://192.168.0.135:8001/mcp",
        "headers": {"Authorization": "Bearer " + key}
    }

uid = path.stat().st_uid if path.exists() else 1000
gid = path.stat().st_gid if path.exists() else 1000

temporary = path.with_name(path.name + ".tmp")
temporary.write_text(json.dumps(config, indent=2) + "\n")
os.chmod(temporary, 0o600)
os.chown(temporary, uid, gid)
os.replace(temporary, path)

print("Konfigurasi MCP tersimpan.")
PY
docker compose restart anythingllm
```

Refresh Agent Skills/MCP.

Hapus sementara entri server yang belum berjalan agar tidak menimbulkan error discovery.

Cara menggunakan MCP dan menghasilkan dokumen

Contoh prompt Kali:

> @agent Gunakan `tool_inventory`, lalu jalankan `nmap_connect` pada 192.168.0.137 untuk port 80 dan 443. Ringkas hanya berdasarkan output tool.

Contoh Intel:

> @agent Gunakan `news_intelligence` untuk mengambil lima berita per feed. Tampilkan judul, URL, waktu publikasi, dan waktu pengambilan.

Contoh file:

> @agent Gunakan `list_files` pada folder laporan. Baca `laporan/operasional.xlsx` menggunakan `read_excel`. Sebutkan sheet dan rentang baris yang dibaca.

Contoh Wazuh:

> @agent Periksa tool Wazuh yang tersedia, lalu gunakan tool baca untuk merangkum agent aktif dan alert terbaru. Jangan menjalankan active response.

Contoh penyusunan dokumen RAG Open WebUI:

> Berdasarkan Knowledge yang dipilih, susun SOP penanganan gangguan jaringan. Buat tujuan, ruang lingkup, diagnosis, eskalasi, dan pencatatan. Gunakan informasi sumber. Tandai usulan tambahan serta informasi yang belum tersedia. Tampilkan langsung di layar.

Periksa tiga tingkat keberhasilan:

Tingkat	Bukti
Discovery	Tool tampil
Eksekusi	Tool benar-benar dipanggil
Interpretasi	Jawaban cocok dengan output

Jawaban naratif tanpa panggilan tool belum membuktikan MCP bekerja.

Untuk workbook besar, jangan menyimpulkan seluruh file dari sedikit baris pertama.

Backup dan pemeliharaan

Data penting:

```
.env
compose.yml
anythingllm.env
data/
ebooks/
workflows/
mcp/
scripts/
artifacts/
```

Wazuh:

```
/etc/wazuh-mcp.env
/etc/systemd/system/wazuh-mcp.service
/opt/blue-team-mcp
```

Gunakan prosedur backup paket setelah menyesuaikan perubahan path dan service:

```
bash scripts/11-backup.sh /mnt/backup
```

Pastikan tujuan merupakan media backup yang benar.

Pemeriksaan rutin:

```
docker compose ps
df -h
free -h
docker compose logs --tail=100 \
  open-webui anythingllm n8n waha mcp-intel
```

Catat versi:

```
docker compose images
docker compose exec -T mcp-files pip freeze
```

Skrip `freeze-images.py` bawaan memuat daftar layanan lama, termasuk Ollama. Sesuaikan sebelum dipakai untuk konfigurasi ini.

Biaya API berasal dari embedding dokumen, embedding pertanyaan, jawaban, tool loop, dan reindex.

Hindari reindex seluruh koleksi berulang tanpa perubahan. Tetap uji pemulihan backup pada lingkungan terpisah.

Troubleshooting dan penerimaan

Gejala	Pemeriksaan
API `401`	Key dan Bearer header
API `403`	Hak akses key/model
API `404`	Endpoint dan model
API `429`	Kuota/rate limit pada body error
Embedding kosong	Respons `/embeddings`
Dimension mismatch	Indeks lama belum dibangun ulang
Open WebUI masih memakai Ollama	Konfigurasi persisten
RAG tidak relevan	Knowledge, indexing, retrieval
Webhook `404`	Workflow aktif dan URL produksi
WhatsApp tidak membalas	Session, filter, execution
MCP tidak tampil	URL, transport, service, JSON
Kali executable hilang	Paket image belum lengkap
File permission denied	UID dan izin sharing
Wazuh gagal	TLS, kredensial, routing, port

Pemeriksaan penerimaan:

- API chat berhasil.
- API embedding menghasilkan vektor.
- Open WebUI memakai dua model yang benar.
- Knowledge diindeks menggunakan model baru.
- Jawaban RAG cocok dengan sumber.
- AnythingLLM memakai pemuda.ai.
- MCP diuji dengan panggilan nyata.
- WhatsApp memakai webhook produksi.
- Help desk menyimpan tiket dan membatasi akses status.
- Backup berhasil dipulihkan.

Rujukan:

- Docker Ubuntu
- Open WebUI environment
- AnythingLLM configuration
- AnythingLLM MCP
- n8n
- WAHA
- Wazuh MCP
- pemuda.ai

Status verifikasi: lampiran dan sintaks Python sumber utama telah diperiksa. Konfigurasi server, kode tambahan dalam dokumentasi, validitas API key, akses model, serta kemampuan tool calling belum diuji langsung pada server target.

Tentang Penulis



Onno W. Purbo, saat ini bertugas sebagai rektor di Institut Teknologi Tangerang Selatan (ITTS). Onno memperoleh gelar Ph.D bidang Electrical Engineering dari University of Waterloo, Canada, adalah seorang copyleft, educator dan ICT evangelist. Dia sudah mempublikasikan ratusan buku, termasuk free ICT ebook untuk sekolah tahun 2008. 23 eBook AI untuk sekolah di tahun 2025. Sekitar 300 eBook tulisannya di sharing free di singkat.in/free-owp-ebooks / s.id/free-owp-ebooks/

Beberapa buku terakhirnya antara lain "Internet-TCP/IP: Konsep Dan Implementasi", 2018; "Sistem Operasi, Konsep Dan Membuat Linux OpenWRT Dan ROM Android", 2019; "IPv6 Untuk Mendukung Operasi Jaringan Dan Domain Name System", 2019; "Kubernetes untuk Pemula", 2024 dan "Membuat Operator Seluler 5G Sendiri", 2024.

Dia memimpin sambungan pertama Internet di Institut Teknologi Bandung, tahun 1993-2000, dan menggunakannya untuk membuat jaringan Internet pendidikan yang pertama di Indonesia. Dia membebaskan frekuensi WiFi, memperkenalkan RT/RW-net, antenna Wajanbolic dan jaringan selular OpenBTS dan private 5G sendiri. Dia memimpin jaringan telepon pertama di atas Internet, VoIP Merdeka, yang kemudian hari dikenal sebagai VoIP Rakyat berbasis SIP dan menggunakan kode area +62520 dan +62521. Dia saat ini aktif memperkenalkan e-Learning, dan menjalankan server e-Learning di http://lms.onnocenter.or.id/moodle/ 66,000++ siswa / mahasiswa dan https://opencourse.itts.ac.id 39.000+ mahasiswa secara gratis. Email: onno@indo.net.id