

PANDUAN PERMULAAN | EDISI OKTOBER 2026

Bina Bot Laporan Harian WhatsApp

Automasi WhatsApp dan penjanaan laporan PDF
dengan AI, langkah demi langkah.

Disediakan oleh Ir. Mohd Lukman Ali

Contoh utama: laporan harian kemajuan tapak bina

Kandungan

Bahagian 1: Mari Mulakan	3
Apa yang Anda Akan Bina	3
Pilih Kes Penggunaan Anda	5
Sebelum Anda Mula	5
Bahagian 2: Bina dalam Lima Milestone	6
Persediaan Sistem	6
Milestone 1: Sambungkan WhatsApp	7
Milestone 2: Simpan Mesej dan Foto	8
Milestone 3: Ringkaskan dengan AI	11
Milestone 4: Jana Laporan PDF	15
Milestone 5: Hantar dan Jadualkan	17
Bahagian 3: Jadikan Ia Milik Anda	20
Menyesuaikan Prompt	20
Menukar Reka Letak Laporan	20
Idea Pengembangan	21
Bahagian 4: Pastikan Ia Terus Berjalan	22
Operasi Sentiasa Hidup	22
Penyelesaian Masalah	23
Keselamatan dan Privasi	23
Langkah Seterusnya	24
Lampiran	25
A. Glosari	25
B. Rujukan Pantas Arahan	25
C. Struktur Fail dan Nota Versi	26

Bahagian 1: Mari Mulakan

Apa yang Anda Akan Bina

Panduan ini mengajar anda membina sebuah sistem kecil yang "mendengar" kumpulan WhatsApp sepanjang hari, kemudian setiap petang menjana **laporan PDF** yang kemas, lengkap dengan ringkasan oleh AI dan galeri foto. Anda tidak perlu mahir pengaturcaraan. Anda hanya perlu mengikut langkah, menyalin kod, dan menyemak **checkpoint** pada hujung setiap milestone.

Hasil akhirnya ialah satu aliran kerja lima peringkat:



Peringkat	Tugas	Alat yang digunakan
1. Mesej WhatsApp	Menyambung ke WhatsApp dan menerima mesej serta foto	Node.js + Baileys
2. Simpan	Menyimpan teks dalam pangkalan data dan foto dalam folder mengikut tarikh	SQLite
3. Ringkas	Meringkaskan perbualan dan menghuraikan foto	Google Gemini API
4. Dokumen	Menyusun laporan DOCX dan menukarnya kepada PDF	python-docx + LibreOffice
5. Hantar	Menghantar PDF ke Telegram dan menjalankannya secara automatik	Telegram Bot + cron

Contoh hasil akhir

Rajah di bawah ialah contoh laporan yang dijana oleh kod dalam panduan ini. Semua data dan foto dalam contoh ialah rekaan.

LAPORAN HARIAN KEMAJUAN TAPAK BINA

Tarikh: 2026-10-06 (CONTOH)

RINGKASAN EKSEKUTIF

Kerja konkrit papak Aras 3 selesai dan kerja pemasangan paip M&E diteruskan di Aras 2. Satu isu bekalan simen dilaporkan petang tadi.

KEMAJUAN KERJA MENGIKUT TRED

- Archi: Pemasangan rangka dinding partition di Aras 2 (Blok A).
- C&S + Infra: Konkrit papak Aras 3 siap dituang; kerja pembentukan parit longkang diteruskan.
- M&E: Pemasangan paip air sejuk di Aras 2 mencapai 60%.
- ICT: Tiada maklumat

LOGISTIK & JENTERA

- Lori simen tiba pukul 2 petang; 1 unit forklift digunakan di Blok A.

ISU TAPAK, DEFECTS & HALANGAN

- Penghantaran bahan tulangan lewat satu hari.

TENAGA KERJA & SUBKONTRAKTOR

- 42 pekerja hadir; 3 subkontraktor aktif.

Galeri Foto

FOTO CONTOH 1

FOTO CONTOH 2

Aras 2, Archi: pemasangan rangka dinding partition.

Aras 3, C&S: papak konkrit selepas penyuangan.

Muka surat pertama laporan contoh (data rekaan)

Cara menggunakan panduan ini

Setiap milestone mempunyai matlamat, kod, penerangan ringkas dan satu **checkpoint**. Jangan teruskan ke milestone berikutnya sebelum checkpoint berjaya. Kotak berwarna digunakan seperti berikut:

CHECKPOINT

Cara untuk mengesahkan bahawa langkah anda berjaya.

TIP

Petua yang menjimatkan masa atau kos.

AMARAN

Perkara yang boleh menyebabkan kerugian data, akaun disekat atau isu keselamatan.

NOTA

Maklumat tambahan untuk difahami.

Pilih Kes Penggunaan Anda

Corak yang sama boleh digunakan untuk banyak tujuan: *mesej kumpulan, simpan, AI ringkaskan, dokumen, hantar*. Yang berbeza hanyalah **prompt** yang memberitahu AI bagaimana laporan ditulis. Panduan ini menyediakan tiga set prompt siap pakai, dan anda boleh menulis sendiri yang keempat.

Kes	Kumpulan WhatsApp	Laporan yang dijana	Nilai USE_CASE
Tapak bina	Pasukan tapak, kontraktor, subkontraktor	Laporan Harian Kemajuan Tapak	construction
Kedai	Staf kedai atau gerai	Laporan Harian Kedai (jualan, stok, aduan)	shop
Perjalanan	Keluarga atau rakan semasa percutian	Diari Perjalanan	trip
Lain-lain	Kelas, sukarelawan, projek komuniti	Tulis prompt anda sendiri (Bahagian 3)	nama pilihan anda

Panduan ini menggunakan **tapak bina** sebagai contoh utama kerana ia kaya dengan unsur yang menarik: banyak tred kerja, banyak foto, dan isu yang perlu direkodkan. Menukar kepada kes lain hanya memerlukan satu baris dalam fail `.env`.

Sebelum Anda Mula

Apa yang anda perlukan

Item	Penerangan	Kos
Nombor WhatsApp khas	SIM atau eSIM kedua. Nombor ini dimasukkan ke dalam kumpulan sasaran.	Mengikut pembekal talian
Komputer atau VPS	Ubuntu 22.04 atau 24.04. Untuk percubaan, laptop biasa (Windows guna WSL2) sudah memadai.	Laptop sedia ada: tiada. VPS: yuran bulanan penyedia
Gemini API key	Dijana daripada Google AI Studio.	Ada had percuma; semak harga semasa
Akaun Telegram	Untuk menerima laporan melalui bot buatan sendiri (@BotFather).	Percuma
Perisian	Node.js 22+, Python 3.10+, LibreOffice, SQLite. Dipasang dalam panduan ini.	Percuma

Anda hanya perlu tahu menggunakan terminal secara asas: berpindah folder (`cd`), melihat fail (`ls`), dan menyunting fail dengan `nano`. Kalau belum biasa, luangkan 15 minit untuk mempelajarinya dahulu.

AMARAN

Baileys bukan API rasmi WhatsApp. Ia meniru WhatsApp Web, jadi WhatsApp boleh menyekat nombor yang dianggap mencurigakan. Gunakan nombor khas (bukan nombor peribadi atau perniagaan utama anda), hanya *mendengar* dan jangan menghantar mesej puka.

AMARAN

Privasi. Mesej kumpulan mengandungi data peribadi. Maklumkan semua ahli kumpulan bahawa mesej direkodkan dan diringkaskan, dan dapatkan persetujuan mereka. Baca bahagian Keselamatan dan Privasi sebelum menggunakannya pada kumpulan sebenar.

Bahagian 2: Bina dalam Lima Milestone

Persediaan Sistem

Matlamat: Komputer anda sedia dengan Node.js, Python, LibreOffice dan folder projek.

Arahan di bawah untuk Ubuntu. Pengguna Windows: pasang WSL2 dengan Ubuntu dan jalankan semua arahan di dalamnya.

1. Kemas kini sistem dan pasang pakej asas:

terminal

```
sudo apt update && sudo apt upgrade -y
sudo apt install -y python3 python3-venv python3-pip sqlite3 libreoffice \
  fonts-dejavu-core fonts-liberation curl git build-essential
```

2. Pasang Node.js 22 LTS:

terminal

```
curl -fsSL https://deb.nodesource.com/setup_22.x | sudo -E bash -
sudo apt install -y nodejs
node --version
```

3. Cipta folder projek dan pasang pakej Node.js:

terminal

```
mkdir -p ~/whatsapp-reporter && cd ~/whatsapp-reporter
npm init -y
npm install @whiskeysockets/baileys qrcode better-sqlite3
```

NOTA

Baileys memerlukan Node.js 20 atau lebih baharu, tetapi Node.js 20 sudah tamat sokongan, jadi gunakan versi 22 LTS atau yang lebih baharu. Semak nodejs.org untuk versi LTS terkini. Pakej **build-essential** diperlukan jika **better-sqlite3** perlu dibina semula.

CHECKPOINT

node --version memaparkan **v22.x** atau lebih tinggi, dan **libreoffice --version** memaparkan nombor versi.

Milestone 1: Sambungkan WhatsApp

Matlamat: Skrip anda boleh "mendengar" mesej WhatsApp dan memaparkannya di terminal.

Kita menggunakan **Baileys** untuk menyambung ke WhatsApp seperti WhatsApp Web. Anda hanya perlu mengimbas **QR code** sekali. Sesi disimpan dalam folder `auth_info`, jadi larian seterusnya tidak memerlukan QR lagi.

AMARAN

Gunakan nombor telefon khas untuk bot ini, bukan nombor utama anda.

1. Cipta fail `collector.js` dengan `nano collector.js`, tampal kod berikut, kemudian simpan (Ctrl+O, Enter, Ctrl+X).

collector.js

```
const { default: makeWASocket, useMultiFileAuthState,
  DisconnectReason, Browsers } = require('@whiskeysockets/baileys');
const QRCode = require('qrcode');

async function start() {
  const { state, saveCreds } = await useMultiFileAuthState('auth_info');
  const sock = makeWASocket({ auth: state, browser: Browsers.ubuntu('Chrome') });
  sock.ev.on('creds.update', saveCreds);

  sock.ev.on('connection.update', async ({ connection, lastDisconnect, qr }) => {
    if (qr) console.log(await QRCode.toString(qr, { type: 'terminal', small: true }));
    if (connection === 'open') console.log('Berjaya disambungkan!');
    if (connection === 'close') {
      const code = lastDisconnect?.error?.output?.statusCode;
      if (code !== DisconnectReason.loggedOut) start();
      else console.log('Logged out. Padam folder auth_info dan imbas QR semula.');
```

Bagaimana kod ini berfungsi

- `useMultiFileAuthState` menyimpan sesi log masuk dalam folder `auth_info`.
- Acara `connection.update` memaparkan QR, mengesahkan sambungan, dan menyambung semula secara automatik jika terputus (kecuali anda telah logged out).
- Acara `messages.upsert` dicetuskan untuk setiap mesej baru. Teks mesej boleh berada dalam `conversation` atau `extendedTextMessage`.

2. Jalankan skrip:

terminal

```
node collector.js
```

3. Imbas QR di telefon khas anda: **WhatsApp > Settings > Linked Devices > Link a Device**.
4. Minta seorang admin memasukkan nombor khas itu ke dalam kumpulan ujian. Hantar satu mesej dalam kumpulan itu daripada telefon lain.

CHECKPOINT

Terminal memaparkan "Berjaya disambungkan!", dan mesej yang anda hantar muncul sebagai **Nama: teks mesej**. Tekan Ctrl+C untuk berhenti.

TIP

Mulakan dengan kumpulan ujian kecil yang mengandungi 2 atau 3 orang yang anda percayai. Bot hanya menerima mesej *selepas* ia dimasukkan ke dalam kumpulan; ia tidak boleh membaca sejarah lama.

Masalah biasa

- QR tidak kelihatan jelas: lebarkan tetingkap terminal atau kecilkan saiz fon.
- Terputus sejeurus selepas imbas: hentikan skrip, padam folder `auth_info` (`rm -r auth_info`) dan imbas semula.

Milestone 2: Simpan Mesej dan Foto

Matlamat: Setiap mesej disimpan dalam pangkalan data **SQLite**, dan setiap foto disimpan dalam folder mengikut tarikh.

Gantikan kandungan `collector.js` dengan versi di bawah. Tiga bahagian baharu ditandakan dengan komen bernombor.

collector.js (versi Milestone 2)

```

const { default: makeWASocket, useMultiFileAuthState, DisconnectReason,
  Browsers, downloadMediaMessage } = require('@whiskeysockets/baileys');
const QRCode = require('qrcode');
const Database = require('better-sqlite3');
const fs = require('fs');
const path = require('path');

const TZ = 'Asia/Kuching'; // tukar ikut zon masa anda

// 1. Sediakan database
const db = new Database('chats.db');
db.exec(`
  CREATE TABLE IF NOT EXISTS messages (
    id TEXT PRIMARY KEY, chat_id TEXT, is_group INTEGER,
    sender_name TEXT, text TEXT, timestamp INTEGER, media_path TEXT
  );
  CREATE INDEX IF NOT EXISTS idx_ts ON messages(timestamp);
`);
const insert = db.prepare(`
  INSERT OR REPLACE INTO messages
    (id, chat_id, is_group, sender_name, text, timestamp, media_path)
  VALUES (?, ?, ?, ?, ?, ?, ?)
`);

async function start() {
  const { state, saveCreds } = await useMultiFileAuthState('auth_info');
  const sock = makeWASocket({ auth: state, browser: Browsers.ubuntu('Chrome') });
  sock.ev.on('creds.update', saveCreds);

  sock.ev.on('connection.update', async ({ connection, lastDisconnect, qr }) => {
    if (qr) console.log(await QRCode.toString(qr, { type: 'terminal', small: true }));
    if (connection === 'open') console.log('Berjaya disambungkan!');
    if (connection === 'close') {
      const code = lastDisconnect?.error?.output?.statusCode;
      if (code !== DisconnectReason.loggedOut) start();
      else console.log('Logged out. Padam folder auth_info dan imbas QR semula.');
    }
  });

  // 2. Simpan setiap mesej baru
  sock.ev.on('messages.upsert', async ({ messages, type }) => {
    if (type !== 'notify') return; // hanya mesej baru
    for (const m of messages) {
      if (!m.message || m.key.fromMe) continue;
      const chatId = m.key.remoteJid;
      if (chatId === 'status@broadcast') continue;

      const text = m.message.conversation
        || m.message.extendedTextMessage?.text
        || m.message.imageMessage?.caption || '';
      const ts = Number(m.messageTimestamp || Math.floor(Date.now() / 1000)) * 1000;

      // 3. Simpan gambar ke folder mengikut tarikh mesej
      let mediaPath = null;
      if (m.message.imageMessage) {
        try {
          const buffer = await downloadMediaMessage(m, 'buffer', {});
          const day = new Date(ts).toLocaleDateString('en-CA', { timeZone: TZ });
          const dir = path.join('media', day);
          fs.mkdirSync(dir, { recursive: true });
          mediaPath = path.join(dir, `${ts}_${m.key.id}.jpg`);
          fs.writeFileSync(mediaPath, buffer);
        } catch (e) { console.error('Gagal muat turun gambar:', e.message); }
      }

      insert.run(m.key.id, chatId, chatId.endsWith('@g.us') ? 1 : 0,
        m.pushName || 'Unknown', text, ts, mediaPath);
      const tag = mediaPath ? '[foto] ' : '';
      console.log(`Disimpan [${chatId}] ${m.pushName || 'Unknown'}: ${tag}${text}`);
    }
  });
}
start();

```

Apa yang baharu

- **Komen 1:** pangkalan data `chats.db` dan jadual `messages` dicipta secara automatik.
 - **Komen 2:** `type !== 'notify'` memastikan hanya mesej baru disimpan, bukan sejarah yang disegerakkan. Mesej daripada bot sendiri dan status WhatsApp diabaikan.
 - **Komen 3:** foto disimpan di `media/TARIKH/` mengikut tarikh *mesej* itu dihantar, dalam zon masa yang anda tetapkan pada pemalar `TZ`. Kapsyen foto juga disimpan sebagai teks.
1. Jalankan semula `node collector.js`. Sesi lama digunakan, jadi QR tidak diminta.
 2. Hantar satu mesej teks dan satu foto dalam kumpulan ujian daripada telefon lain.
 3. Buka terminal kedua dan semak data:

terminal

```
sqlite3 chats.db "SELECT datetime(timestamp/1000, 'unixepoch', 'localtime'), sender_name, text
  FROM messages ORDER BY timestamp DESC LIMIT 5;"
ls media/*
```

CHECKPOINT

Pertanyaan SQL memaparkan mesej anda, dan folder `media/TARIKH/` mengandungi fail `.jpg`. Log skrip memaparkan baris `Disimpan [ID-sembang] Nama: ....`

TIP

Catat nilai yang berakhir dengan `@g.us` dalam log; itu ialah **ID kumpulan**. Jika nombor khas anda berada dalam banyak kumpulan, anda akan memasukkannya ke dalam `.env` pada Milestone 3 supaya laporan hanya menggunakan satu kumpulan.

NOTA

Versi ini hanya menyimpan **teks dan foto**. Video, nota suara dan dokumen diabaikan. Lihat Idea Pengembangan pada Bahagian 3.

Milestone 3: Ringkaskan dengan AI

Matlamat: Mesej hari ini dihantar kepada **Gemini** dan anda menerima ringkasan dalam Bahasa Melayu.

Langkah 1: Dapatkan API key

Buka aistudio.google.com, log masuk dengan akaun Google, pilih **Get API key** dan salin kunci itu. Jangan kongsikannya dengan sesiapa atau tampilkan dalam kod.

Langkah 2: Cipta fail .env

Fail `.env` menyimpan tetapan dan rahsia di luar kod. Cipta dengan `nano .env`:

```
.env
```

```
GEMINI_API_KEY=isi_kunci_anda_di_sini
GEMINI_MODEL=gemini-2.5-flash
TIMEZONE=Asia/Kuching
USE_CASE=construction
TARGET_CHAT_ID=
```

```
terminal
```

```
chmod 600 .env
printf ".env\nauth_info/\nchats.db\nmedia/\nreports/\nnode_modules/\n.venv/\n" > .gitignore
```

NOTA

Nama model Gemini berubah dari semasa ke semasa. Jika anda menerima ralat 404 untuk model, semak senarai model semasa di dokumentasi Gemini API dan kemas kini `GEMINI_MODEL`. Isi `TARGET_CHAT_ID` dengan ID kumpulan (`...@g.us`) jika bot berada dalam banyak kumpulan; biarkan kosong untuk menggunakan semua sembang.

Langkah 3: Sediakan persekitaran Python

```
terminal
```

```
python3 -m venv .venv
source .venv/bin/activate
pip install requests python-dotenv python-docx
```

Selepas `source .venv/bin/activate`, prompt terminal akan bermula dengan (`.venv`). Aktifkan semula setiap kali anda membuka terminal baru.

Langkah 4: Cipta lima fail Python

config.py

```
import os
from dotenv import load_dotenv

load_dotenv() # baca fail .env

GEMINI_API_KEY = os.getenv("GEMINI_API_KEY", "")
GEMINI_MODEL = os.getenv("GEMINI_MODEL", "gemini-2.5-flash")
TIMEZONE = os.getenv("TIMEZONE", "Asia/Kuching")
USE_CASE = os.getenv("USE_CASE", "construction") # construction | shop | trip
MAX_PHOTOS = int(os.getenv("MAX_PHOTOS", "12"))
DB_PATH = os.getenv("DB_PATH", "chats.db")
TARGET_CHAT_ID = os.getenv("TARGET_CHAT_ID", "") # kosong = semua sembang
TELEGRAM_BOT_TOKEN = os.getenv("TELEGRAM_BOT_TOKEN", "")
TELEGRAM_CHAT_ID = os.getenv("TELEGRAM_CHAT_ID", "")
```

prompts.py

Semua prompt ditulis dalam English; output dihasilkan dalam Bahasa Melayu.

```
FORMAT_RULES = (
    "Formatting rules: plain text only. Start every section heading line with '## '. "
    "Use '- ' for bullet points. Do not use any other markdown symbols. "
    "Do not invent facts that are not in the transcript. "
    "If a section has no information, write 'Tiada maklumat'. "
    "Ignore greetings, jokes and chatter that is unrelated to the topic."
)

PROMPTS = {
    "construction": {
        "title": "LAPORAN HARIAN KEMAJUAN TAPAK BINA",
        "summary": (
            "You are a Senior Site Project Manager. Based on today's WhatsApp group "
            "transcript, write a Daily Site Progress Report in Bahasa Melayu with these "
            "sections: RINGKASAN EKSEKUTIF; KEMAJUAN KERJA MENGIKUT TRED (Archi, "
            "C&S + Infra, M&E, ICT); LOGISTIK & JENTERA; ISU TAPAK, DEFECTS & HALANGAN; "
            "TENAGA KERJA & SUBKONTRAKTOR. " + FORMAT_RULES
        ),
        "photo": (
            "Analyze this construction site photo. In 1 sentence in Bahasa Melayu state: "
            "(1) Location/Level, (2) Work trade (Archi/C&S/M&E/Infra), (3) Exact "
            "activity observed. If you are unsure about something, say so; do not guess."
        ),
    },
}
```

db_utils.py

```

import sqlite3
from datetime import datetime, timedelta
from zoneinfo import ZoneInfo
from config import DB_PATH, TIMEZONE, TARGET_CHAT_ID

def get_day_messages(date_str):
    """Ambil semua mesej pada satu tarikh (YYYY-MM-DD) ikut zon masa anda."""
    tz = ZoneInfo(TIMEZONE)
    start = datetime.fromisoformat(date_str).replace(tzinfo=tz)
    end = start + timedelta(days=1)
    conn = sqlite3.connect(DB_PATH)
    conn.row_factory = sqlite3.Row
    sql = "SELECT * FROM messages WHERE timestamp >= ? AND timestamp < ?"
    args = [int(start.timestamp() * 1000), int(end.timestamp() * 1000)]
    if TARGET_CHAT_ID:
        # hanya satu kumpulan
        sql += " AND chat_id = ?"
        args.append(TARGET_CHAT_ID)
    rows = conn.execute(sql + " ORDER BY timestamp", args).fetchall()
    conn.close()
    return rows

def build_transcript(rows):
    """Tukar baris database kepada teks transkrip untuk AI."""
    tz = ZoneInfo(TIMEZONE)
    lines = []
    for r in rows:
        hmmm = datetime.fromtimestamp(r["timestamp"] / 1000, tz).strftime("%H:%M")
        tag = "[PHOTO] " if r["media_path"] else ""
        if r["text"] or tag:
            lines.append(f"[{hmmm}] {r['sender_name']}: {tag}{r['text']}")
    return "\n".join(lines)

```

ai_engine.py

```

import base64
import requests
from config import GEMINI_API_KEY, GEMINI_MODEL

URL = (
    "https://generativelanguage.googleapis.com/v1beta/models/"
    f"{GEMINI_MODEL}:generateContent"
)

def _call(parts, timeout=90):
    if not GEMINI_API_KEY:
        raise RuntimeError("GEMINI_API_KEY belum diisi dalam fail .env")
    resp = requests.post(
        URL,
        headers={"x-goog-api-key": GEMINI_API_KEY, "Content-Type": "application/json"},
        json={"contents": [{"parts": parts}]},
        timeout=timeout,
    )
    resp.raise_for_status()
    return resp.json()["candidates"][0]["content"]["parts"][0]["text"].strip()

def summarize(transcript, system_prompt):
    text = f"{system_prompt}\n\nTranscript:\n{transcript}"
    return _call([{"text": text}])

def describe_photo(image_path, prompt):
    with open(image_path, "rb") as f:
        b64 = base64.b64encode(f.read()).decode("utf-8")
    return _call([
        {"text": prompt},
        {"inline_data": {"mime_type": "image/jpeg", "data": b64}},
    ], timeout=60)

```

test_summary.py

```

from datetime import datetime
from zoneinfo import ZoneInfo
from config import TIMEZONE, USE_CASE
from prompts import PROMPTS
from db_utils import get_day_messages, build_transcript
from ai_engine import summarize

today = datetime.now(ZoneInfo(TIMEZONE)).strftime("%Y-%m-%d")
transcript = build_transcript(get_day_messages(today))
print("--- TRANSKRIP ---")
print(transcript or "(kosong)")
if transcript:
    print("\n--- RINGKASAN AI ---")
    print(summarize(transcript, PROMPTS[USE_CASE]["summary"]))

```

Anatomi sebuah prompt

Prompt dalam `prompts.py` ditulis dalam English kerana AI lazimnya mengikut arahan English dengan lebih tepat, tetapi ia mengarahkan output dalam Bahasa Melayu. Setiap prompt mempunyai empat bahagian:

- **Peranan:** "You are a Senior Site Project Manager" menetapkan sudut pandang AI.
- **Tugas:** laporan apa yang perlu ditulis, dan dalam bahasa apa.
- **Struktur:** senarai seksyen tetap supaya laporan konsisten setiap hari.
- **Peraturan:** jangan reka maklumat, tulis "Tiada maklumat" jika kosong, abaikan sembang yang tidak berkaitan, dan format `##` untuk tajuk supaya mudah diproses pada Milestone 4.

1. Pastikan collector sudah menyimpan beberapa mesej *hari ini*, kemudian jalankan:

terminal

```
python test_summary.py
```

CHECKPOINT

Anda melihat transkrip mesej hari ini, diikuti ringkasan AI dalam Bahasa Melayu dengan tajuk bermula `##` .
Jika transkrip kosong, semak `TIMEZONE` dan `TARGET_CHAT_ID`.

Milestone 4: Jana Laporan PDF

Matlamat: Ringkasan AI dan foto digabungkan menjadi satu fail PDF.

python-docx menyusun dokumen Word (tajuk, ringkasan, galeri foto dua lajur), kemudian **LibreOffice** dalam mod headless menukarnya kepada PDF. Pendekatan ini mudah kerana anda boleh membuka fail DOCX yang sama dan menyuntingnya di Word jika perlu.

report_compiler.py

```
import os
import subprocess
import docx
from docx.shared import Inches, Pt, RGBColor

def add_markdown(doc, text):
    """Papar teks AI: '## ' = tajuk, '- ' = bullet, selebihnya perenggan."""
    for line in text.splitlines():
        line = line.strip().replace("***", "")
        if not line:
            continue
        if line.startswith("## "):
            doc.add_heading(line[3:], level=2)
        elif line.startswith("- "):
            doc.add_paragraph(line[2:], style="List Bullet")
        else:
            doc.add_paragraph(line)

def create_pdf_report(date_str, title, summary_text, photos, output_dir="reports"):
    os.makedirs(output_dir, exist_ok=True)
    doc = docx.Document()

    # 1. Tajuk
    p = doc.add_paragraph()
    r = p.add_run(f"{title}\nTarikh: {date_str}")
    r.bold = True
    r.font.size = Pt(16)
    r.font.color.rgb = RGBColor(15, 23, 42)

    # 2. Ringkasan AI
    add_markdown(doc, summary_text)

    # 3. Galeri foto (2 lajur) dengan kapsyen
    if photos:
        doc.add_heading("Galeri Foto", level=1)
        table = doc.add_table(rows=0, cols=2)
        for i in range(0, len(photos), 2):
            cells = table.add_row().cells
            for j in range(2):
                if i + j >= len(photos):
                    continue
                img_path, caption = photos[i + j]
                para = cells[j].paragraphs[0]
                if os.path.exists(img_path):
                    para.add_run().add_picture(img_path, width=Inches(3.0))
                para.add_run(f"\n{caption}").font.size = Pt(8.5)

    # 4. Simpan DOCX, tukar kepada PDF
    docx_path = os.path.join(output_dir, f"Report_{date_str}.docx")
    doc.save(docx_path)
    subprocess.run(
        ["libreoffice", "--headless", "--convert-to", "pdf",
         "--outdir", output_dir, docx_path],
        check=True, timeout=180,
    )
    return os.path.join(output_dir, f"Report_{date_str}.pdf")
```

main.py (versi Milestone 4)

```

import sys
from datetime import datetime
from zoneinfo import ZoneInfo

from config import TIMEZONE, USE_CASE, MAX_PHOTOS
from prompts import PROMPTS
from db_utils import get_day_messages, build_transcript
from ai_engine import summarize, describe_photo
from report_compiler import create_pdf_report

def main():
    # Tarikh: guna argumen (python main.py 2026-10-06) atau hari ini
    if len(sys.argv) > 1:
        date_str = sys.argv[1]
    else:
        date_str = datetime.now(ZoneInfo(TIMEZONE)).strftime("%Y-%m-%d")

    rows = get_day_messages(date_str)
    if not rows:
        print(f"Tiada mesej untuk {date_str}. Laporan tidak dijana.")
        return

    prompt = PROMPTS[USE_CASE]
    summary = summarize(build_transcript(rows), prompt["summary"])

    photos = []
    for r in [r for r in rows if r["media_path"]][:MAX_PHOTOS]:
        try:
            caption = describe_photo(r["media_path"], prompt["photo"])
        except Exception as e:
            print("Gagal huraikan foto:", e)
            caption = "(huraian tidak tersedia)"
        photos.append((r["media_path"], caption))

    pdf_path = create_pdf_report(date_str, prompt["title"], summary, photos)
    print("PDF dijana:", pdf_path)

if __name__ == "__main__":
    main()

```

Bagaimana ia berfungsi

- **add_markdown** menukar **##** kepada tajuk dan **-** kepada bullet; teks lain menjadi perenggan.
- Setiap foto (sehingga **MAX_PHOTOS**, lalai 12) dihantar kepada Gemini untuk dihurai dalam satu ayat, dan huraian itu menjadi kapsyen. Had foto menjimatkan kos dan masa.
- Jika tiada mesej pada tarikh itu, laporan tidak dijana.

1. Jalankan untuk hari ini, atau untuk tarikh tertentu:

terminal

```

python main.py
python main.py 2026-10-05

```

2. Jika anda menggunakan VPS, muat turun PDF ke komputer anda (jalankan di komputer anda):

terminal (komputer anda)

```
scp namapengguna@IP-VPS:~/whatsapp-reporter/reports/Report_2026-10-05.pdf .
```

CHECKPOINT

Terminal memaparkan **PDF dijana: reports/Report_TARIKH.pdf**. Apabila dibuka, PDF mengandungi tajuk, ringkasan seksyen demi seksyen, dan galeri foto dengan kapsyen.

TIP

Larian pertama LibreOffice boleh mengambil 10 hingga 30 saat. Jika penukaran gagal tanpa mesej jelas, hentikan proses lama dengan **kill soffice** dan cuba lagi.

Milestone 5: Hantar dan Jadualkan

Matlamat: PDF dihantar ke Telegram dan dijana secara automatik setiap petang.

Langkah 1: Cipta bot Telegram

1. Dalam Telegram, cari **@BotFather**, hantar **/newbot** dan ikut arahan. Salin **token** yang diberi.
2. Hantar satu mesej kepada bot baru anda (atau masukkan ke dalam kumpulan Telegram).
3. Buka alamat ini di pelayar (gantikan TOKEN) dan cari nilai **"chat":{"id":...}**. Itulah **chat id** anda:

pelayar

```
https://api.telegram.org/botTOKEN/getUpdates
```

4. Tambah dua baris ke dalam **.env**:

.env (tambahan)

```
TELEGRAM_BOT_TOKEN=token_daripada_botfather
TELEGRAM_CHAT_ID=chat_id_anda
```

Langkah 2: Tambah modul penghantaran

telegram_sender.py

```
import requests

def send_pdf(bot_token, chat_id, pdf_path, caption=""):
    url = f"https://api.telegram.org/bot{bot_token}/sendDocument"
    with open(pdf_path, "rb") as f:
        resp = requests.post(
            url,
            data={"chat_id": chat_id, "caption": caption},
            files={"document": f},
            timeout=120,
        )
    resp.raise_for_status()
    return True
```

Gantikan **main.py** dengan versi akhir. Tambahkan: import Telegram, dan blok di hujung yang menghantar PDF jika token dan chat id diisi.

main.py (versi akhir)

```

import sys
from datetime import datetime
from zoneinfo import ZoneInfo

from config import (TIMEZONE, USE_CASE, MAX_PHOTOS,
                    TELEGRAM_BOT_TOKEN, TELEGRAM_CHAT_ID)
from prompts import PROMPTS
from db_utils import get_day_messages, build_transcript
from ai_engine import summarize, describe_photo
from report_compiler import create_pdf_report
from telegram_sender import send_pdf

def main():
    # Tarikh: guna argumen (python main.py 2026-10-06) atau hari ini
    if len(sys.argv) > 1:
        date_str = sys.argv[1]
    else:
        date_str = datetime.now(ZoneInfo(TIMEZONE)).strftime("%Y-%m-%d")

    rows = get_day_messages(date_str)
    if not rows:
        print(f"Tiada mesej untuk {date_str}. Laporan tidak dijana.")
        return

    prompt = PROMPTS[USE_CASE]
    summary = summarize(build_transcript(rows), prompt["summary"])

    photos = []
    for r in [r for r in rows if r["media_path"]][:MAX_PHOTOS]:
        try:
            caption = describe_photo(r["media_path"], prompt["photo"])
        except Exception as e:
            print("Gagal huraikan foto:", e)
            caption = "(huraian tidak tersedia)"
        photos.append((r["media_path"], caption))

    pdf_path = create_pdf_report(date_str, prompt["title"], summary, photos)
    print("PDF dijana:", pdf_path)

    if TELEGRAM_BOT_TOKEN and TELEGRAM_CHAT_ID:
        send_pdf(TELEGRAM_BOT_TOKEN, TELEGRAM_CHAT_ID, pdf_path,
                f"{prompt['title']} - {date_str}")
        print("Dihantar ke Telegram.")

if __name__ == "__main__":
    main()

```

terminal

```
python main.py
```

CHECKPOINT

PDF tiba dalam sembang Telegram anda dengan kapsyen tajuk dan tarikh.

Langkah 3: Jadualkan dengan cron

cron ialah penjadual tugas dalam Linux. Pertama, pastikan zon masa pelayan anda betul kerana VPS biasanya menggunakan UTC:

terminal

```
timedatectl  
sudo timedatectl set-timezone Asia/Kuching  
crontab -e
```

Tambah baris ini pada hujung fail crontab:

crontab

```
0 17 * * 1-6 cd ~/whatsapp-reporter && .venv/bin/python main.py >> cron.log 2>&1
```

Bacaannya: minit **0**, jam **17**, setiap hari bulan, setiap bulan, hari **1-6** (Isnin hingga Sabtu). Output dan ralat direkodkan dalam **cron.log**.

AMARAN

cron hanya menjana laporan. Skrip **collector.js** mesti sentiasa berjalan untuk mengumpul mesej. Bahagian 4 menerangkan cara menjadikannya berjalan 24 jam.

Bahagian 3: Jadikan Ia Milik Anda

Menyesuaikan Prompt

Prompt ialah bahagian yang paling berkuasa untuk diubah. Dua set tambahan di bawah boleh ditambah ke dalam **PROMPTS** dalam **prompts.py**, kemudian tukar **USE_CASE** dalam **.env**.

prompts.py (tambahan)

```
"shop": {
    "title": "LAPORAN HARIAN KEDAI",
    "summary": (
        "You are a retail operations assistant. Based on today's WhatsApp group "
        "transcript from the shop staff, write a Daily Shop Report in Bahasa Melayu "
        "with these sections: RINGKASAN JUALAN; PRODUK LARIS; STOK HAMPIR HABIS; "
        "MASALAH & ADUAN PELANGGAN; TINDAKAN ESOK. " + FORMAT_RULES
    ),
    "photo": (
        "Describe this shop photo in 1 sentence in Bahasa Melayu: what product or "
        "area is shown and anything that needs attention (empty shelf, damage, etc.)."
    ),
},
"trip": {
    "title": "DIARI PERJALANAN",
    "summary": (
        "You are a friendly travel diary writer. Based on today's WhatsApp group "
        "transcript, write a short Travel Diary entry in Bahasa Melayu with these "
        "sections: RINGKASAN HARI INI; TEMPAT YANG DILAWATI; MAKANAN & PENGINAPAN; "
        "MOMEN TERBAIK; PERBELANJAAN. " + FORMAT_RULES
    ),
    "photo": (
        "Describe this travel photo in 1 short sentence in Bahasa Melayu: the place "
        "or activity shown. If you are unsure about the location, say so."
    ),
},
},
```

Menulis prompt anda sendiri

Contohnya untuk kumpulan kelas atau sukarelawan, tulis dahulu apa yang pembaca laporan ingin tahu, kemudian tukarnya kepada seksyen. Pola yang berkesan:

- Mulakan dengan peranan yang jelas dan jenis dokumen yang dihasilkan.
- Senaraikan seksyen yang tepat; AI akan mengikutnya setiap hari.
- Sentiasa sertakan peraturan: jangan reka maklumat, tulis "Tiada maklumat" jika kosong.
- Gunakan **FORMAT_RULES** yang sama supaya output serasi dengan penjana PDF.
- Uji pada tarikh lama dengan **python main.py TARIKH**, ubah prompt, dan ulang sehingga puas hati.

Menukar Reka Letak Laporan

Semua perubahan reka letak dibuat dalam **report_compiler.py**. Beberapa contoh:

Perubahan	Cara
Tajuk laporan	Ubah nilai <code>title</code> dalam prompt yang berkaitan.
Saiz foto	Ubah <code>Inches(3.0)</code> . Gunakan <code>2.2</code> untuk foto lebih kecil.
Bilangan foto	Ubah <code>MAX_PHOTOS</code> dalam <code>.env</code> .
Bahasa output	Ubah arahan bahasa dalam prompt (contohnya "in English").
Fon	<code>doc.styles["Normal"].font.name = "Liberation Sans"</code> (gunakan fon yang dipasang pada pelayan).
Logo	Tambah <code>doc.add_picture("logo.png", width=Inches(1.2))</code> sebelum tajuk.

Untuk reka letak yang lebih canggih, sediakan fail templat `.docx` dengan header, footer dan gaya syarikat anda, kemudian buka dengan `docx.Document("templat.docx")` sebagai ganti `docx.Document()`.

Idea Pengembangan

Idea	Pendekatan ringkas
Beberapa kumpulan	Lakukan <code>main.py</code> untuk setiap <code>TARGET_CHAT_ID</code> dan hasilkan satu PDF bagi setiap kumpulan.
Laporan mingguan	Kumpul 7 ringkasan harian dan minta AI meringkaskannya semula.
Nota suara	Simpan audio dalam collector dan hantar kepada Gemini untuk transkripsi.
Dokumen dan video	Tambah pengendali untuk <code>documentMessage</code> dan <code>videoMessage</code> dalam collector.
Hantar melalui e-mel	Gunakan <code>smtplib</code> sebagai ganti atau tambahan kepada Telegram.
Simpan ke Google Drive	Muat naik PDF selepas dijana untuk arkib automatik.
Papan pemuka web	Bina halaman kecil yang membaca <code>chats.db</code> dan memaparkan laporan lama.

Bahagian 4: Pastikan Ia Terus Berjalan

Operasi Sentiasa Hidup

Collector mesti berjalan sepanjang masa. Untuk percubaan, laptop yang dihidupkan memadai. Untuk kegunaan sebenar, gunakan VPS dan **systemd**, pengurus perkhidmatan Linux yang memulakan semula skrip secara automatik jika ia berhenti atau pelayan dihidupkan semula.

1. Imbas QR dahulu secara manual di VPS: jalankan `node collector.js`, imbas QR, tunggu "Berjaya disambungkan!", kemudian tekan Ctrl+C.
2. Jalankan `whoami` dan `which node` untuk mengetahui nama pengguna dan laluan Node.js anda.
3. Cipta fail perkhidmatan dengan `sudo nano /etc/systemd/system/whatsapp-collector.service`. Gantikan `NAMAPENGGUNA` dengan hasil `whoami`:

whatsapp-collector.service

```
[Unit]
Description=WhatsApp Collector
After=network-online.target
Wants=network-online.target

[Service]
Type=simple
User=NAMAPENGGUNA
WorkingDirectory=/home/NAMAPENGGUNA/whatsapp-reporter
ExecStart=/usr/bin/node collector.js
Restart=always
RestartSec=10

[Install]
WantedBy=multi-user.target
```

4. Aktifkan dan semak:

terminal

```
sudo systemctl daemon-reload
sudo systemctl enable --now whatsapp-collector
sudo systemctl status whatsapp-collector
journalctl -u whatsapp-collector -f
```

CHECKPOINT

Status memaparkan **active (running)**. Mesej baru dalam kumpulan muncul dalam log `journalctl`. Cuba `sudo reboot`; selepas pelayan hidup semula, collector berjalan sendiri.

AMARAN

Jika sesi WhatsApp tamat (logged out), hentikan perkhidmatan (`sudo systemctl stop whatsapp-collector`), padam `auth_info`, imbas QR secara manual semula, dan mulakan semula perkhidmatan.

Sandaran dan pembersihan

terminal

```
# sandaran mingguan (simpan di tempat lain)
tar czf backup.tgz auth_info chats.db .env

# padam foto lebih 90 hari
find ~/whatsapp-reporter/media -mindepth 1 -maxdepth 1 -type d -mtime +90 -exec rm -rf {} +
```

Penyelesaian Masalah

Masalah	Punca biasa	Penyelesaian
QR tidak muncul atau pecah	Terminal terlalu sempit	Lebarkan terminal atau kecilkan fon.
Terputus sejeurus selepas imbas	Sesi auth_info rosak	Padam auth_info , imbas semula.
Logged out	Peranti dinyahpautkan di telefon, atau sesi tamat	Padam auth_info dan imbas QR semula.
Mesej tidak disimpan	Nombor belum dalam kumpulan, atau collector tidak berjalan	Semak log; pastikan ujian dihantar selepas bot masuk kumpulan.
Laporan "Tiada mesej"	Zon masa atau TARGET_CHAT_ID salah	Semak TIMEZONE , date pada pelayan, dan ID kumpulan.
Ralat 403 atau 429 daripada Gemini	Kunci salah atau kuota habis	Semak GEMINI_API_KEY dan had penggunaan akaun.
Ralat 404 untuk model	Nama model sudah berubah	Kemas kini GEMINI_MODEL .
PDF tidak dijana	LibreOffice tiada atau proses tersangkut	libreoffice --version ; jalankan pkill soffice dan cuba lagi.
Telegram: chat not found	Bot belum dihubungi atau chat id salah	Hantar mesej kepada bot, semak semula getUpdates .
better-sqlite3 gagal dipasang	Alat binaan tiada	sudo apt install build-essential python3 kemudian npm install semula.
cron tidak berjalan	Laluan atau zon masa salah	Semak crontab -l dan tail -50 cron.log .

Keselamatan dan Privasi

- **Rahsia dalam .env:** jalankan **chmod 600 .env**, jangan muat naik ke GitHub, dan pastikan **.gitignore** mengandungi **.env**.
- **Folder auth_info:** sesiapa yang memilikinya boleh mengawal sesi WhatsApp bot. Jangan kongsi dan sertakan dalam sandaran yang selamat.
- **Persetujuan:** beritahu ahli kumpulan bahawa mesej direkodkan dan diringkaskan oleh AI. Di Malaysia, pemprosesan data peribadi tertakluk kepada Akta Perlindungan Data Peribadi 2010 (PDPA); rujuk dasar organisasi atau nasihat pakar jika perlu.
- **Data dihantar ke Google:** teks dan foto dihantar kepada Gemini API. Semak terma penggunaan data bagi tier percuma dan berbayar, dan jangan gunakan sistem ini untuk maklumat yang sangat sensitif.
- **Simpanan data:** tetapkan tempoh simpanan (contohnya 90 hari) dan padam foto serta mesej lama.

- **Pelayan:** aktifkan firewall asas (`sudo ufw allow OpenSSH` kemudian `sudo ufw enable`). Bot ini tidak memerlukan sebarang port masuk selain SSH.
- **Risiko sekatan nombor:** gunakan nombor khas, hanya mendengar, dan jangan sekali-kali menghantar mesej pukal menggunakan Baileys.

Langkah Seterusnya

Anda kini mempunyai kerangka lengkap: mesej masuk, disimpan, diringkaskan, dijadikan dokumen dan dihantar. Beberapa cadangan: tulis prompt untuk kes anda sendiri, kumpulkan laporan seminggu untuk melihat kualiti, tambah sokongan nota suara, kemudian bina papan pemuka. Selamat mencuba.

Lampiran

A. Glosari

Istilah	Maksud
API key	Kata laluan khas yang membenarkan program anda menggunakan perkhidmatan seperti Gemini.
Baileys	Pustaka Node.js tidak rasmi yang menyambung ke WhatsApp seperti WhatsApp Web.
cron	Penjadual tugas Linux yang menjalankan arahan pada masa tertentu.
Daemon / systemd	Program yang berjalan di latar belakang; systemd mengurus dan memulakannya semula.
<code>.env</code>	Fail teks yang menyimpan tetapan dan rahsia di luar kod.
Prompt	Arahan yang diberikan kepada AI untuk memberitahu apa dan bagaimana ia perlu menulis.
QR code	Kod grafik yang diimbas dengan telefon untuk memautkan peranti ke WhatsApp.
SQLite	Pangkalan data ringkas yang disimpan dalam satu fail (<code>chats.db</code>).
SSH	Cara selamat untuk mengawal komputer jauh (VPS) melalui terminal.
venv	Persekitaran Python berasingan supaya pakej projek tidak bercampur dengan sistem.
VPS	Pelayan maya yang disewa dan berjalan 24 jam di pusat data.

B. Rujukan Pantas Arahan

Tugas	Arahan
Mulakan collector secara manual	<code>cd ~/whatsapp-reporter && node collector.js</code>
Status perkhidmatan	<code>sudo systemctl status whatsapp-collector</code>
Lihat log langsung	<code>journalctl -u whatsapp-collector -f</code>
Mulakan semula perkhidmatan	<code>sudo systemctl restart whatsapp-collector</code>
Aktifkan persekitaran Python	<code>source .venv/bin/activate</code>
Jana laporan hari ini	<code>python main.py</code>
Jana laporan tarikh lain	<code>python main.py 2026-10-05</code>
Lihat 5 mesej terkini	<code>sqlite3 chats.db "SELECT sender_name, text FROM messages ORDER BY timestamp DESC LIMIT 5;"</code>
Lihat jadual cron	<code>crontab -l</code>
Lihat log cron	<code>tail -50 cron.log</code>
Sandaran	<code>tar czf backup.tgz auth_info chats.db .env</code>

C. Struktur Fail dan Nota Versi

```
whatsapp-reporter/
whatsapp-reporter/
  collector.js      # Milestone 1-2: terima dan simpan mesej
  config.py         # tetapan daripada .env
  prompts.py        # prompt AI mengikut kes penggunaan
  db_utils.py       # baca mesej mengikut tarikh
  ai_engine.py      # panggilan Gemini (teks dan foto)
  report_compiler.py # jana DOCX dan tukar kepada PDF
  telegram_sender.py # hantar PDF ke Telegram
  main.py           # penyelaras utama (dijalankan oleh cron)
  test_summary.py   # ujian ringkasan
.env chats.db auth_info/ media/ reports/ cron.log
```

Nota ketepatan

Panduan ini disediakan pada Oktober 2026. Bahagian Python (pangkalan data, penjanaan DOCX dan penukaran PDF) diuji dengan data contoh. Pemasangan dan permulaan skrip Node.js turut disemak, tetapi sambungan sebenar ke WhatsApp, Gemini dan Telegram tidak dapat diuji semasa penyediaan. Perkhidmatan luaran ini kerap berubah, jadi jika sesuatu arahan tidak berfungsi, semak dokumentasi rasmi.

Komponen	Versi rujukan
Node.js	22 LTS
Baileys	7.0.0-rc14 (jika versi terkini mengubah API: <code>npm install @whiskeysockets/baileys@7.0.0-rc14</code>)
Python	3.12 (Ubuntu 24.04)
python-docx	1.2
LibreOffice	daripada repositori Ubuntu