repositori github: https://github.com/DandyYahmin/uas-eit

Crawl data dari twitter @TMCPoldaMetro

```
twitter_auth_token = 'b33af24ed78827356029e26971c6159a4ea5cb5d'

# Import required Python package
!pip install pandas

# Install Node.js (because tweet-harvest built using Node.js)
!sudo apt-get update
!sudo apt-get install -y ca-certificates curl gnupg
!sudo mkdir -p /etc/apt/keyrings
!curl -fsSL https://deb.nodesource.com/gpgkey/nodesource-repo.gpg.key
| sudo gpg --dearmor -o /etc/apt/keyrings/nodesource.gpg

!NODE_MAJOR=20 && echo "deb
[signed-by=/etc/apt/keyrings/nodesource.gpg]
https://deb.nodesource.com/node_$NODE_MAJOR.x nodistro main" | sudo
tee /etc/apt/sources.list.d/nodesource.list

!sudo apt-get update
!sudo apt-get install nodejs -y

!node -v
```

filter berdasarkan kata kunci "Situasi arus lalu lintas di"

```
from datetime import datetime, timedelta

# Generate date range: today and 365 days ago
until_date = datetime.now().date()
since_date = until_date - timedelta(days=365)

# Define target
username = "TMCPoldaMetro"
keyword = '"Situasi arus lalu lintas di"'
filename = 'tmc_polda.csv'

search_keyword = f'from:{username} {keyword} since:{since_date} until:
{until_date} lang:id'
limit = 800  # adjust this depending on how much data you want

# Run tweet-harvest
!npx -y tweet-harvest@2.6.1 -o "{filename}" -s "{search_keyword}" --
tab "LATEST" -l {limit} --token {twitter_auth_token}
```

filter data yang memiliki pola "HH.MM Situasi arus lalu lintas di"

```
import pandas as pd
import re

# 1. Memuat data
df = pd.read_csv('tweets-data/tmc_polda.csv')

# 2. Definisikan pola regex yang akan dicari
# r'\d{2}:\d{2}' mencari pola waktu (jam:menit (HH:MM))
# diikuti dengan spasi dan teks "Situasi arus lalu lintas"
pattern = r'\d{2}.\d{2} Situasi arus lalu lintas di'

# 3. Filter DataFrame
# - df['full_text'].str.contains() digunakan untuk mencari string/pola
dalam Series/kolom.
# - regex=True memberitahu pandas bahwa 'pattern' adalah sebuah
regular expression.
# - na=False akan memperlakukan nilai NaN (jika ada) sebagai False
(tidak cocok).
mask = df['full_text'].str.contains(pattern, regex=True, na=False)

# 4. Mengambil kolom 'created_at' dan 'full_text' dari data yang sudah
terfilter
# Kita ambil kedua kolom yang kita butuhkan.
# Gunakan .copy() untuk menghindari SettingWithCopyWarning saat kita
memodifikasi DataFrame nanti.
lalu_lintas = df.loc[mask, ['created_at', 'full_text']].copy()

# 5. Konversi kolom 'created_at' ke format datetime, ubah timezone,
dan format ulang
#    a. pd.to_datetime(): Mengubah string tanggal menjadi objek
datetime. Pandas cerdas
#       dan akan secara otomatis mengenali format "+0000" sebagai UTC.
#    b. .dt.tz_convert('Asia/Jakarta'): Mengonversi timezone dari UTC
ke WIB (UTC+7).
#    c. .dt.strftime('%d %b %Y'): Memformat ulang objek datetime
menjadi string dengan format
#       Hari(angka) Bulan(singkat) Tahun(lengkap). Contoh: "23 Jun
2025"
lalu_lintas['created_at'] =
pd.to_datetime(lalu_lintas['created_at']).dt.tz_convert('Asia/Jakarta'
).dt.strftime('%d %b %Y')

# 6. Mengganti nama kolom
lalu_lintas = lalu_lintas.rename(columns={
    'created_at': 'date',
    'full_text': 'text'
```

```
})

# 7. Menyimpan DataFrame yang sudah berisi dua kolom ke file csv baru
lalu_lintas.to_csv('lalu_lintas.csv', index=False)
```

Ekstraksi menggunakan metode In-NER

```
import pandas as pd
import re
import json

# Memecah sebuah kalimat (teks) menjadi daftar kata-kata atau "token".
def custom_tokenizer(text):
    if not isinstance(text, str): return []
    # Menghapus url dari text
    urls = re.findall(r'https://t.co/\S+', text)
    text_no_urls = re.sub(r'https://t.co/\S+', '', text)
    tokens = [token for token in re.split(r'([,.()])?\s+',
text_no_urls) if token]
    return tokens

# Membersihkan string lokasi yang sudah diekstraksi agar formatnya
konsisten.
def clean_location(loc_str):
    if not loc_str: return None
    # Menghapus kata depan seperti di, dari, ke, menuju.
    loc_str = re.sub(r'^(di|dari|ke|menuju)\s+', '', loc_str.strip())
    # Menghapus spasi atau tanda baca yang tidak perlu di awal dan
akhir string.
    loc_str = loc_str.strip(' ,.()')
    # Mengubah string menjadi format Title Case (setiap kata diawali
huruf kapital), kecuali jika string tersebut sudah mengandung "Jl."
    return loc_str if 'Jl.' in loc_str else loc_str.title()

# Pola untuk mengenali waktu
TIME_PATTERN = re.compile(r'^\d{2}\.\d{2}$')

# Kata-kata yang biasanya mengawali sebuah frasa lokasi
LOC_STARTERS = {'di', 'dari', 'depan', 'kawasan', 'on', 'exit'}

# Kata-kata kunci yang menandakan lokasi
LOC_KEYWORDS = {
    'arteri', 'bundaran', 'csw', 'exit', 'gbk', 'gerbang', 'graha',
'gt', 'interchange',
    'jakbar', 'jakpus', 'jaksel', 'jaktim', 'jakut', 'jembatan', 'jl',
'jlnt', 'junction',
    'km', 'kolong', 'lampu', 'light', 'merah', 'monas', 'mrt', 'off',
'pasar', 'pgc',
```

```
    'pintu', 'pospol', 'ramp', 'ruas', 'semanggi', 'simpang', 'susun',
'terminal', 'tl',
    'tmii', 'tugu', 'tol', 'traffic', 'underpass', 'wisata'
}

# Kata-kata yang menggambarkan kondisi lalu lintas
STATUS_KEYWORDS = {'cukup', 'cenderung', 'lancar', 'mengalir',
'normal', 'padat', 'ramai'}

# Kata kunci arah
DIRECTION_KEYWORD = {'arah', 'mengarah', 'menuju'}

# Kata kunci dua arah
BIDIRECTIONAL_KEYWORDS = {'maupun', 'sebaliknya', 'dan'}

# Kata kunci penyebab hambatan
OBSTACLE_CAUSE_KEYWORDS = {'karena', 'imbas', 'dikarenakan'}

# Kata kunci cuaca
WEATHER = {'hujan','berawan','gerimis','cerah'}

'''
Memberikan "tag" atau label pada setiap token yang dihasilkan oleh
custom_tokenizer.
Label ini mengikuti skema BIO (Beginning, Inside, Outside).
'''
# B- (Beginning): Menandai awal dari sebuah entitas.
# I- (Inside): Menandai bagian dalam atau kelanjutan dari sebuah
entitas.
# O (Outside): Menandai token yang tidak termasuk dalam kategori
manapun.
'''
menggunakan variabel state (is_location, is_status) untuk melacak
konteks.
Jika token sebelumnya diberi label B-LOC, token berikutnya kemungkinan
besar adalah I-LOC
(kecuali jika token tersebut adalah kata pemutus seperti 'dan' atau
'terpantau').
'''
def label_tokens(text):
    tokens = custom_tokenizer(text)
    labels = []
    is_location, is_status = False, False

    for i, tok in enumerate(tokens):
        lower_tok = tok.lower()
        if i == 0: is_location = is_status = False

        if tok.startswith('https://t.co/'):
             labels.append("O")
```

```
            is_location = is_status = False
        elif TIME_PATTERN.match(tok):
            labels.append("B-TIME")
            is_location = is_status = False
        elif lower_tok in BIDIRECTIONAL_KEYWORDS:
            labels.append("B-BIDIR")
            is_location = False
        elif lower_tok in DIRECTION_KEYWORD:
            labels.append("B-DIR")
            is_location = False
        elif lower_tok in OBSTACLE_CAUSE_KEYWORDS:
            labels.append("B-OBSTACLE-CAUSE")
            is_location = is_status = False
        elif lower_tok in LOC_STARTERS or any(kw in lower_tok for kw
in LOC_KEYWORDS):
            labels.append("B-LOC" if not is_location else "I-LOC")
            is_location = True; is_status = False
        elif lower_tok in STATUS_KEYWORDS:
            labels.append("B-STATUS" if not is_status else "I-STATUS")
            is_status = True; is_location = False
        elif is_location:
            if lower_tok in ['terpantau', 'sedangkan', 'diimbau',
'dan', 'yang']:
                labels.append("O"); is_location = False
            else:
                labels.append("I-LOC")
        elif lower_tok in WEATHER:
            labels.append("B-WEATHER")
            is_location = is_status = False
        else:
            labels.append("O")
            is_location = is_status = False
    return tokens, labels

'''
Mengambil hasil dari label_tokens (daftar token dan labelnya)
dan mengubahnya menjadi format data yang rapi (dictionary).
'''
# Fungsi ini mencari token dengan label B- (Beginning) untuk menemukan
awal dari suatu informasi.
'''
Ketika B- ditemukan, ia akan terus maju untuk mengumpulkan semua token
I- yang mengikutinya,
lalu menggabungkannya menjadi satu frasa utuh (misalnya, [B-LOC:
'Jl.', I-LOC: 'Gatot', I-LOC: 'Subroto']
menjadi "Jl. Gatot Subroto").
'''
# Setelah semua potongan informasi (waktu, status, lokasi, dll.)
terkumpul, fungsi ini mulai membangun record data.
'''
```

```
Logika utamanya adalah menangani berbagai skenario lokasi:
Satu Lokasi: Hanya ada satu lokasi yang disebutkan.
Dua Arah (maupun): "dari A maupun ke B" akan dipecah menjadi dua
record: (dari: A, ke: B).
Bolak-balik (sebaliknya): "A ke B dan arah sebaliknya" akan dipecah
menjadi dua record: (dari: A, ke: B) dan (dari: B, ke: A).
Standar (Dari-Ke): Jika ada dua lokasi, lokasi pertama dianggap from
dan yang kedua to.
'''
def extract_from_tagged(tokens, labels, date_from_csv,
base_from=None):
    time, status, obstacle, weather = None, None, None, None
    locations, bidir_flags = [], []

    i = 0
    while i < len(tokens):
        tag = labels[i]
        if tag.startswith("B-"):
            chunk_tokens = []; chunk_type = tag[2:]
            j = i
            while j < len(tokens) and (labels[j] == f"B-{chunk_type}"
or labels[j] == f"I-{chunk_type}"):
                chunk_tokens.append(tokens[j]); j += 1
            full_chunk = " ".join(chunk_tokens)
            if chunk_type == "TIME": time = full_chunk.replace('.',
':')
            elif chunk_type == "STATUS": status = full_chunk
            elif chunk_type == "LOC":
locations.append(clean_location(full_chunk))
            elif chunk_type == "BIDIR":
bidir_flags.append(tokens[i].lower())
            elif chunk_type == "OBSTACLE-CAUSE":
                obs_tokens = [tokens[k] for k in range(j, len(tokens))
if tokens[k].lower() not in ['diimbau', 'agar']]
                if obs_tokens: obstacle = ' '.join(obs_tokens).strip("
.")
            elif chunk_type == "WEATHER": weather = full_chunk
            i = j
        else: i += 1

    results = []

    if base_from and not locations: return []
    if base_from and not any(re.search(r'di |dari ', loc, re.I) for
loc in locations):
        locations.insert(0, base_from)

    if not time or not status or not locations: return []

    base_record = {'time': time, 'date': date_from_csv, 'from': None,
```

```
'to': None,
                    'status': status, 'obstacle': obstacle, 'weather':
weather}

    if len(locations) == 1:
        results.append({**base_record, 'from': locations[0], 'to':
None})
    elif 'maupun' in bidir_flags and len(locations) > 1:
        origin = locations[0]
        for dest in locations[1:]:
            results.append({**base_record, 'from': origin, 'to':
dest})
    elif 'sebaliknya' in bidir_flags and len(locations) > 1:
        loc1, loc2 = locations[0], locations[1]
        results.append({**base_record, 'from': loc1, 'to': loc2})
        results.append({**base_record, 'from': loc2, 'to': loc1})
    elif len(locations) >= 2:
        results.append({**base_record, 'from': locations[0], 'to':
locations[1]})

    return results

# Membaca file lalu_lintas.csv ke dalam sebuah DataFrame pandas.
df_input = pd.read_csv('lalu_lintas.csv')
all_extracted_data = []
base_origin_for_sedangkan = None

# Melakukan loop untuk setiap baris dalam file CSV.
for index, row in df_input.iterrows():
    if not isinstance(row['text'], str): continue

    '''
    Menangani "sedangkan" kalimat seperti "Tol Cikampek ramai,
sedangkan Tol Jagorawi lancar" mengandung dua laporan terpisah.
    re.split(r'\s+sedangkan\s+', ...) memecah kalimat tersebut menjadi
dua bagian, yang kemudian diproses secara terpisah.
    '''
    parts = re.split(r'\s+sedangkan\s+', row['text'],
flags=re.IGNORECASE)


    for i, part in enumerate(parts):
        tokens, labels = label_tokens(part)

        '''
        digunakan untuk menangani kasus seperti "Tol Cikampek arah
Jakarta lancar, sedangkan arah sebaliknya padat".
        Lokasi "Tol Cikampek" dari bagian pertama kalimat akan
digunakan kembali sebagai konteks untuk bagian kedua ("arah
sebaliknya").
```

```
        '''
        contextual_origin = base_origin_for_sedangkan if i > 0 else
None
        extracted_records = extract_from_tagged(tokens, labels,
row['date'], contextual_origin)

        if extracted_records:
            # Setiap record data yang berhasil diekstraksi akan
ditambahkan ke dalam list
            all_extracted_data.extend(extracted_records)
            if i == 0 and extracted_records[0]['from']:
                 base_origin_for_sedangkan = extracted_records[0]
['from']
    base_origin_for_sedangkan = None

# Sebelum disimpan, data disaring. Hanya record yang memiliki semua
kunci wajib (time, date, from, to, status) yang akan disimpan
MANDATORY_KEYS = ['time', 'date', 'from', 'to', 'status']
final_data = [
    rec for rec in all_extracted_data if all(rec.get(key) for key in
MANDATORY_KEYS)
]

# Menyimpan hasil ekstraksi ke JSON
output_filename = 'traffic_data_inner.json'
with open(output_filename, 'w', encoding='utf-8') as f:
    json.dump(final_data, f, ensure_ascii=False, indent=4)
```

Ekstraksi menggunakan metode regex

```
import csv
import re
import json
from datetime import datetime

# --- 1. DEFINISI ENTITAS DAN KATA KUNCI (DENGAN LOKASI YANG
DIPERBANYAK) ---

# Tabel Bantu Lokasi yang sudah diperkaya secara signifikan
LOKASI_JAKARTA = [
    # Jakarta Pusat
    {'nama_jalan': 'sudirman', 'keywords': ['sudirman', 'jl jend
sudirman', 'jendral sudirman', 'fx sudirman'], 'wilayah': 'jakarta
pusat'},
    {'nama_jalan': 'thamrin', 'keywords': ['thamrin', 'mh thamrin',
'jl mh thamrin'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'bundaran hi', 'keywords': ['bundaran hi',
'bunderan hi', 'hotel indonesia', 'plaza gi'], 'wilayah': 'jakarta
```

```
pusat'},
    {'nama_jalan': 'monas', 'keywords': ['monas', 'silang monas',
'monas timur', 'kawasan monas'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'patung kuda', 'keywords': ['patung kuda',
'bundaran patung kuda', 'monas barat daya'], 'wilayah': 'jakarta
pusat'},
    {'nama_jalan': 'gbk', 'keywords': ['gbk', 'gelora bung karno',
'pintu 10 gbk', 'gbk senayan'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'senayan', 'keywords': ['senayan', 'bundaran
senayan', 'senayan city', 'plaza senayan', 'lapangan tembak senayan'],
'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'simpang lima senen', 'keywords': ['simpang lima
senen', 'senen', 'stasiun senen', 'ps senen'], 'wilayah': 'jakarta
pusat'},
    {'nama_jalan': 'cempaka putih', 'keywords': ['cempaka putih'],
'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'salemba', 'keywords': ['salemba', 'salemba raya',
'tl carolus'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'tugu tani', 'keywords': ['tugu tani'], 'wilayah':
'jakarta pusat'},
    {'nama_jalan': 'gambir', 'keywords': ['gambir'], 'wilayah':
'jakarta pusat'},
    {'nama_jalan': 'kwitang', 'keywords': ['kwitang'], 'wilayah':
'jakarta pusat'},
    {'nama_jalan': 'harmoni', 'keywords': ['harmoni'], 'wilayah':
'jakarta pusat'},
    {'nama_jalan': 'kemayoran', 'keywords': ['kemayoran', 'jiexpo
kemayoran'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'palmerah', 'keywords': ['palmerah'], 'wilayah':
'jakarta pusat'},
    {'nama_jalan': 'parman', 'keywords': ['parman', 's parman', 'jl s
parman', 'letjend s parman'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'pejompongan', 'keywords': ['pejompongan',
'penjernihan'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'suryopranoto', 'keywords': ['suryopranoto', 'jl
suryopranoto'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'gunung sahari', 'keywords': ['gunung sahari', 'jl
gunung sahari'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'pintu besi', 'keywords': ['pintu besi'],
'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'kramat raya', 'keywords': ['kramat raya', 'jl
kramat raya'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'kramat bunder', 'keywords': ['kramat bunder', 'jl
kramat bunder'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'letjen suprapto', 'keywords': ['letjen suprapto',
'jl letjen suprapto'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'asia afrika', 'keywords': ['asia afrika', 'jl asia
afrika'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'gerbang pemuda', 'keywords': ['gerbang pemuda',
```

```
'jl gerbang pemuda'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'biak', 'keywords': ['biak', 'tl biak'], 'wilayah':
'jakarta pusat'},
    {'nama_jalan': 'dukuh atas', 'keywords': ['dukuh atas'],
'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'tanah abang', 'keywords': ['tanah abang'],
'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'sarinah', 'keywords': ['sarinah'], 'wilayah':
'jakarta pusat'},
    {'nama_jalan': 'kebon sirih', 'keywords': ['kebon sirih'],
'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'wahidin', 'keywords': ['wahidin', 'jl dr
wahidin'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'karet bivak', 'keywords': ['karet bivak'],
'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'dpr/mpr', 'keywords': ['dpr ri', 'dpr/mpr ri',
'gedung dpr/mpr'], 'wilayah': 'jakarta pusat'},
    {'nama_jalan': 'lapangan banteng', 'keywords': ['lapangan
banteng'], 'wilayah': 'jakarta pusat'},

    # Jakarta Selatan
    {'nama_jalan': 'gatot subroto', 'keywords': ['gatot subroto',
'gatsu', 'jl gatot subroto'], 'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'kuningan', 'keywords': ['kuningan', 'gt kuningan',
'mall kuningan city', 'rasuna said', 'jl rasuna said', 'hr rasuna
said'], 'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'pancoran', 'keywords': ['pancoran'], 'wilayah':
'jakarta selatan'},
    {'nama_jalan': 'semanggi', 'keywords': ['semanggi', 'plaza
semanggi', 'jembatan semanggi', 'gt semanggi', 'off ramp semanggi'],
'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'tendean', 'keywords': ['tendean'], 'wilayah':
'jakarta selatan'},
    {'nama_jalan': 'fatmawati', 'keywords': ['fatmawati', 'on ramp
fatmawati', 'mrt fatmawati'], 'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'lebak bulus', 'keywords': ['lebak bulus'],
'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'cilandak', 'keywords': ['cilandak', 'gt
cilandak'], 'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'pasar minggu', 'keywords': ['pasar minggu', 'ps
minggu'], 'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'kalibata', 'keywords': ['kalibata', 'tl
kalibata'], 'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'ragunan', 'keywords': ['ragunan', 'taman
margasatwa ragunan'], 'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'blok m', 'keywords': ['blok m'], 'wilayah':
'jakarta selatan'},
    {'nama_jalan': 'mt haryono', 'keywords': ['mt haryono', 'jl mt
haryono'], 'wilayah': 'jakarta selatan'},
```

```
    {'nama_jalan': 'tb simatupang', 'keywords': ['tb simatupang', 'jl
tb simatupang'], 'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'pondok indah', 'keywords': ['pondok indah', 'pim',
'underpass pim'], 'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'bintaro', 'keywords': ['bintaro', 'hankam
bintaro'], 'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'antasari', 'keywords': ['antasari', 'jlnt
antasari'], 'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'prapanca', 'keywords': ['prapanca raya'],
'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'satrio', 'keywords': ['jl dr satrio', 'jl satrio',
'casablanca', 'jlnt casablanca'], 'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'mampang', 'keywords': ['mampang', 'mampang
prapatan'], 'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'tebet', 'keywords': ['tebet', 'off ramp tebet'],
'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'cipete', 'keywords': ['cipete', 'off ramp
cipete'], 'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'ampera', 'keywords': ['ampera', 'gt ampera'],
'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'pakubuwono', 'keywords': ['pakubuwono'],
'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'iskandarsyah', 'keywords': ['iskandarsyah raya'],
'wilayah': 'jakarta selatan'},
    {'nama_jalan': 'melawai', 'keywords': ['melawai'], 'wilayah':
'jakarta selatan'},
    {'nama_jalan': 'panglima polim', 'keywords': ['panglima polim'],
'wilayah': 'jakarta selatan'},

    # Jakarta Timur
    {'nama_jalan': 'cawang', 'keywords': ['cawang', 'cawang kompor',
'uki cawang', 'cawang interchange'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'ahmad yani', 'keywords': ['ahmad yani', 'jl ahmad
yani'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'rawamangun', 'keywords': ['rawamangun', 'mega
rawamangun'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'panjaitan', 'keywords': ['panjaitan', 'jl di
panjaitan'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'matraman', 'keywords': ['matraman'], 'wilayah':
'jakarta timur'},
    {'nama_jalan': 'jatinegara', 'keywords': ['jatinegara', 'kodim
lama jatinegara'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'kampung rambutan', 'keywords': ['kampung
rambutan', 'kp rambutan'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'pasar rebo', 'keywords': ['pasar rebo', 'ps
rebo'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'tmii', 'keywords': ['tmii', 'taman mini', 'pintu 1
tmii'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'pgc', 'keywords': ['pgc', 'pgc cililitan'],
```

```
'wilayah': 'jakarta timur'},
    {'nama_jalan': 'cililitan', 'keywords': ['cililitan'], 'wilayah':
'jakarta timur'},
    {'nama_jalan': 'condet', 'keywords': ['condet'], 'wilayah':
'jakarta timur'},
    {'nama_jalan': 'cakung', 'keywords': ['cakung', 'kolong cakung',
'pospol cakung'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'pulogadung', 'keywords': ['pulogadung', 'pulo
gadung'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'bekasi raya', 'keywords': ['jl raya bekasi',
'bekasi raya'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'halim', 'keywords': ['halim', 'halim baru', 'halim
lama'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'dewi sartika', 'keywords': ['dewi sartika', 'jl
dewi sartika'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'bogor raya', 'keywords': ['bogor raya', 'jl raya
bogor'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'arion', 'keywords': ['arion', 'lampu merah
arion'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'pramuka', 'keywords': ['jl pramuka raya'],
'wilayah': 'jakarta timur'},
    {'nama_jalan': 'bambu apus', 'keywords': ['bambu apus'],
'wilayah': 'jakarta timur'},
    {'nama_jalan': 'cibubur', 'keywords': ['cibubur', 'cibubur
junction'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'tamini square', 'keywords': ['tamini square',
'lampu merah garuda', 'tmi square'], 'wilayah': 'jakarta timur'},
    {'nama_jalan': 'cijantung', 'keywords': ['cijantung'], 'wilayah':
'jakarta timur'},
    {'nama_jalan': 'pinang ranti', 'keywords': ['pinang ranti'],
'wilayah': 'jakarta timur'},
    {'nama_jalan': 'otista', 'keywords': ['otista raya'], 'wilayah':
'jakarta timur'},
    {'nama_jalan': 'utan kayu', 'keywords': ['utan kayu'], 'wilayah':
'jakarta timur'},
    {'nama_jalan': 'caglak', 'keywords': ['caglak', 'tl caglak'],
'wilayah': 'jakarta timur'},
    {'nama_jalan': 'rindam', 'keywords': ['rindam'], 'wilayah':
'jakarta timur'},
    {'nama_jalan': 'kiwi', 'keywords': ['kiwi', 'tl kiwi'], 'wilayah':
'jakarta timur'},

    # Jakarta Barat
    {'nama_jalan': 'daan mogot', 'keywords': ['daan mogot', 'jl daan
mogot', 'daanmogot baru'], 'wilayah': 'jakarta barat'},
    {'nama_jalan': 'slipi', 'keywords': ['slipi', 'lampu merah slipi',
'tl slipi'], 'wilayah': 'jakarta barat'},
    {'nama_jalan': 'tomang', 'keywords': ['tomang', 'underpass
tomang'], 'wilayah': 'jakarta barat'},
```

```
    {'nama_jalan': 'grogol', 'keywords': ['grogol', 'central park',
'rs darmais'], 'wilayah': 'jakarta barat'},
    {'nama_jalan': 'cengkareng', 'keywords': ['cengkareng', 'tl
cengkareng'], 'wilayah': 'jakarta barat'},
    {'nama_jalan': 'kalideres', 'keywords': ['kalideres'], 'wilayah':
'jakarta barat'},
    {'nama_jalan': 'pesing', 'keywords': ['pesing'], 'wilayah':
'jakarta barat'},
    {'nama_jalan': 'kota', 'keywords': ['kota', 'stasiun kota', 'kota
tua'], 'wilayah': 'jakarta barat'},
    {'nama_jalan': 'asemka', 'keywords': ['asemka', 'tl asemka'],
'wilayah': 'jakarta barat'},
    {'nama_jalan': 'jembatan lima', 'keywords': ['jembatan lima'],
'wilayah': 'jakarta barat'},
    {'nama_jalan': 'jalan panjang', 'keywords': ['jl panjang', 'jalan
panjang'], 'wilayah': 'jakarta barat'},
    {'nama_jalan': 'puri kembangan', 'keywords': ['puri kembangan',
'ringroad puri'], 'wilayah': 'jakarta barat'},
    {'nama_jalan': 'kedoya', 'keywords': ['kedoya'], 'wilayah':
'jakarta barat'},
    {'nama_jalan': 'latumenten', 'keywords': ['latumenten'],
'wilayah': 'jakarta barat'},
    {'nama_jalan': 'meruya', 'keywords': ['meruya', 'gt meruya'],
'wilayah': 'jakarta barat'},
    {'nama_jalan': 'roxy', 'keywords': ['roxy'], 'wilayah': 'jakarta
barat'},

    # Jakarta Utara
    {'nama_jalan': 'priok', 'keywords': ['priok', 'tanjung priok', 'tj
priok', 'pelabuhan tj priok'], 'wilayah': 'jakarta utara'},
    {'nama_jalan': 'ancol', 'keywords': ['ancol', 'taman impian jaya
ancol', 'bintang mas ancol'], 'wilayah': 'jakarta utara'},
    {'nama_jalan': 'pluit', 'keywords': ['pluit', 'emporium pluit',
'gt pluit'], 'wilayah': 'jakarta utara'},
    {'nama_jalan': 'kelapa gading', 'keywords': ['kelapa gading',
'moi', 'mall of indonesia'], 'wilayah': 'jakarta utara'},
    {'nama_jalan': 'sunter', 'keywords': ['sunter'], 'wilayah':
'jakarta utara'},
    {'nama_jalan': 'cilincing', 'keywords': ['cilincing', 'jl cakung
cilincing'], 'wilayah': 'jakarta utara'},
    {'nama_jalan': 'koja', 'keywords': ['koja', 'tl jaya koja'],
'wilayah': 'jakarta utara'},
    {'nama_jalan': 'marunda', 'keywords': ['marunda'], 'wilayah':
'jakarta utara'},
    {'nama_jalan': 'pademangan', 'keywords': ['pademangan'],
'wilayah': 'jakarta utara'},
    {'nama_jalan': 'muara angke', 'keywords': ['muara angke'],
'wilayah': 'jakarta utara'},
    {'nama_jalan': 'jembatan tiga', 'keywords': ['jembatan tiga'],
```

```
'wilayah': 'jakarta utara'},
    {'nama_jalan': 'perintis', 'keywords': ['perintis', 'tl
perintis'], 'wilayah': 'jakarta utara'},
    {'nama_jalan': 'kebon baru', 'keywords': ['kebon baru', 'tl kebon
baru'], 'wilayah': 'jakarta utara'},

    # Ruas Tol
    {'nama_jalan': 'tol dalam kota', 'keywords': ['tol dalam kota',
'dalkot', 'ruas tol dalam kota'], 'wilayah': 'lintas wilayah'},
    {'nama_jalan': 'tol jakarta-cikampek', 'keywords': ['tol jakarta-
cikampek', 'tol japek', 'ruas tol japek'], 'wilayah': 'lintas
wilayah'},
    {'nama_jalan': 'tol jagorawi', 'keywords': ['tol jagorawi', 'ruas
tol jagorawi'], 'wilayah': 'lintas wilayah'},
    {'nama_jalan': 'tol jakarta-tangerang', 'keywords': ['tol jakarta-
tangerang', 'ruas tol jakarta-tangerang', 'tol janger'], 'wilayah':
'lintas wilayah'},
    {'nama_jalan': 'tol jorr', 'keywords': ['tol jorr', 'ruas tol
jorr', 'jorr w2s'], 'wilayah': 'lintas wilayah'},
    {'nama_jalan': 'tol sedyatmo', 'keywords': ['tol sedyatmo', 'ruas
tol sedyatmo', 'tol bandara'], 'wilayah': 'lintas wilayah'},
    {'nama_jalan': 'tol cawang-priok', 'keywords': ['tol cawang -
tanjung priok', 'tol wiyoto wiyono'], 'wilayah': 'lintas wilayah'},
    {'nama_jalan': 'layang mbz', 'keywords': ['layang mbz'],
'wilayah': 'lintas wilayah'},
    {'nama_jalan': 'cikunir', 'keywords': ['cikunir', 'off ramp
cikunir', 'gt cikunir', 'km 09+750'], 'wilayah': 'bekasi'},

    # Lainnya
    {'nama_jalan': 'bandara soetta', 'keywords': ['bandara soetta',
'bandara soekarno hatta'], 'wilayah': 'tangerang'},
    {'nama_jalan': 'bekasi', 'keywords': ['bekasi'], 'wilayah':
'bekasi'},
    {'nama_jalan': 'depok', 'keywords': ['depok'], 'wilayah':
'depok'},
    {'nama_jalan': 'tangerang', 'keywords': ['tangerang'], 'wilayah':
'tangerang'},
    {'nama_jalan': 'jatiasih', 'keywords': ['jatiasih'], 'wilayah':
'bekasi'},
]


# Urutkan berdasarkan panjang keyword, dari terpanjang ke terpendek
ALL_LOC_KEYWORDS = []
for loc in LOKASI_JAKARTA:
    for keyword in loc['keywords']:
        ALL_LOC_KEYWORDS.append(keyword)
ALL_LOC_KEYWORDS = sorted(list(set(ALL_LOC_KEYWORDS)), key=len,
reverse=True)
```

```
STATUS_KEYWORDS = {
    'padat merayap': ['padat merayap'],
    'ramai cenderung padat': ['ramai cenderung padat'],
    'cukup padat': ['cukup padat', 'agak padat'],
    'padat': ['padat'],
    'tersendat': ['tersendat', 'terhambat', 'sedikit terhambat'],
    'ramai lancar': ['ramai lancar', 'ramai lancer'],
    'lancar': ['lancar'],
    'kondusif': ['kondusif'],
}

OBSTACLE_KEYWORDS = {
    'proyek': ['proyek', 'pengerjaan', 'betonisasi jalan',
'pembetonan', 'perbaikan jalan', 'perbaikan', 'pengaspalan',
'pembangunan lrt', 'galian'],
    'kecelakaan': ['kecelakaan', 'laka lantas', 'gangguan ban', 'ban
pecah'],
    'kendaraan gangguan': ['gangguan', 'mogok', 'kendala'],
    'demonstrasi': ['penyampaian pendapat', 'aksi masyarakat',
'aliansi mahasiswa'],
    'kegiatan': ['kegiatan', 'fun run', 'konser', 'pertandingan sepak
bola', 'hbkb'],
    'balap liar': ['balap liar'],
    'banjir': ['banjir', 'genangan'],
    'rekayasa lalin': ['rekayasa lalu lintas', 'buka tutup',
'pengalihan arus', 'penutupan arus'],
    'antrian': ['antrian kendaraan', 'antrian'],
    'volume kendaraan': ['volume kendaraan'],
}

WEATHER_KEYWORDS = {
    'hujan': ['hujan'],
    'gerimis': ['gerimis'],
    'berawan': ['berawan'],
    'cerah': ['cerah'],
}

BIDIRECTIONAL_KEYWORDS = [
    'arah sebaliknya', 'kedua arah', 'dan sebaliknya', 'maupun arah
sebaliknya',
    'arah berlawanan', 'maupun arah', 'arah ... maupun arah',
'dari ... menuju ... maupun'
]

# --- 2. FUNGSI-FUNGSI EKSTRAKSI ---

def find_keywords(text, keyword_map):
    for canonical, variations in keyword_map.items():
        for var in variations:
```

```
            if var in text:
                return canonical
    return None

def ner_locations(text, location_keywords):
    found_locations = []
    temp_text = " " + text + " "
    for loc in location_keywords:
        if f" {loc} " in temp_text:
            found_locations.append(loc)
            temp_text = temp_text.replace(f" {loc} ", " <LOC_FOUND> ",
1)
    # Urutkan berdasarkan posisi kemunculan di teks asli
    found_locations.sort(key=lambda x: text.find(x))
    # Kembalikan daftar unik sambil mempertahankan urutan
    return list(dict.fromkeys(found_locations))

def determine_from_to(text, locations):
    if len(locations) < 2:
        return []
    pairs = []

    # Pola: di [loc1] arah [loc2] maupun arah [loc3]
    maupun_match = re.search(r'di (.*?) arah (.*?) maupun arah (.*)',
text)
    if maupun_match and len(locations) > 0:
        source = locations[0]
        for dest in locations[1:]:
            if dest in maupun_match.group(0):
                pairs.append((source, dest))
        if pairs: return pairs # Langsung kembalikan jika pola ini
cocok

    # Pola: [loc1] ... arah/menuju/ke ... [loc2]
    for i in range(len(locations) - 1):
        loc1_re = re.escape(locations[i])
        loc2_re = re.escape(locations[i+1])
        pattern = re.compile(f"{loc1_re}.*?(?:arah|menuju|ke|menuju
arah){{1,2}}.*{loc2_re}")
        if pattern.search(text):
            pairs.append((locations[i], locations[i+1]))

    # Pola: dari [loc1] menuju [loc2]
    for i in range(len(locations) - 1):
        loc1_re = re.escape(locations[i])
        loc2_re = re.escape(locations[i+1])
        pattern = re.compile(f"dari {loc1_re} (?:menuju|ke)
{loc2_re}")
        if pattern.search(text):
            pairs.append((locations[i], locations[i+1]))
```

```
    # Jika tidak ada pola jelas, asumsikan urutan kemunculan
    if not pairs and len(locations) >= 2:
        pairs.append((locations[0], locations[1]))

    return list(dict.fromkeys(pairs))

# --- 3. PROSES UTAMA ---

def process_traffic_data(csv_filepath):
    """
    Memproses file CSV lalu lintas, mengekstrak tanggal dari kolom
'date'
    dan waktu dari kolom 'text'.
    """
    all_extracted_data = []
    try:
        with open(csv_filepath, mode='r', encoding='utf-8') as
csvfile:
            reader = csv.DictReader(csvfile)
            for row in reader:
                # Ambil data dari kolom 'date' dan 'text'
                date_from_csv = row.get('date', '').strip()
                text_from_csv = row.get('text', '').strip()

                if not date_from_csv or not text_from_csv:
                    continue

                # Ekstrak waktu dari awal kolom 'text' (misal:
"19.42 ...")
                time_match = re.search(r'^(\d{2}[.:]\d{2})',
text_from_csv)
                if not time_match:
                    continue  # Lewati baris jika tidak ada format
waktu di awal

                time = time_match.group(1).replace('.', ':')
                date = date_from_csv

                # Bersihkan teks dari stempel waktu untuk diproses
lebih lanjut
                cleaned_text = re.sub(r'^\d{2}[.:]\d{2}\s*', '',
text_from_csv).lower()

                # Ekstrak entitas lainnya menggunakan teks yang sudah
bersih
                status = find_keywords(cleaned_text, STATUS_KEYWORDS)
                obstacle = find_keywords(cleaned_text,
OBSTACLE_KEYWORDS)
                weather = find_keywords(cleaned_text,
```

```
WEATHER_KEYWORDS)
                locations_found = ner_locations(cleaned_text,
ALL_LOC_KEYWORDS)
                from_to_pairs = determine_from_to(cleaned_text,
locations_found)

                if not from_to_pairs:
                    continue

                is_bidirectional = any(keyword in cleaned_text for
keyword in BIDIRECTIONAL_KEYWORDS)

                for from_loc, to_loc in from_to_pairs:
                    if not all([time, date, from_loc, to_loc,
status]):
                        continue

                    data_entry = {
                        "time": time,
                        "date": date,
                        "from": from_loc.title(),
                        "to": to_loc.title(),
                        "status": status.title(),
                        "obstacle": obstacle.title() if obstacle else
None,
                        "weather": weather.title() if weather else
None
                    }
                    all_extracted_data.append(data_entry)

                    if is_bidirectional:
                        reversed_entry = data_entry.copy()
                        reversed_entry["from"] = to_loc.title()
                        reversed_entry["to"] = from_loc.title()
                        all_extracted_data.append(reversed_entry)

    except FileNotFoundError:
        print(f"Error: File tidak ditemukan di '{csv_filepath}'")
        return None
    return all_extracted_data

if __name__ == "__main__":
    csv_file = 'lalu_lintas.csv'
    output_json_file = 'traffic_data_regex.json'

    extracted_data = process_traffic_data(csv_file)

    if extracted_data:
        # Menghapus duplikat entri sebelum menyimpan
        unique_data = [dict(t) for t in {tuple(d.items()) for d in
extracted_data}]
```

```
        print(f"Berhasil mengekstrak {len(unique_data)} data lalu
lintas unik.")

        try:
            with open(output_json_file, 'w', encoding='utf-8') as f:
                json.dump(unique_data, f, indent=2,
ensure_ascii=False)

            print(f"Hasil ekstraksi telah berhasil disimpan ke file:
'{output_json_file}'")

        except IOError as e:
            print(f"Gagal menyimpan file: {e}")

    else:
        print("Tidak ada data yang dapat diekstrak atau file CSV
kosong/tidak ditemukan.")
```

```
Berhasil mengekstrak 677 data lalu lintas unik.
Hasil ekstraksi telah berhasil disimpan ke file:
'traffic_data_regex.json'
```