

LAPORAN PROJECT AKHIR SEMESTER
MATA KULIAH SISTEM OPERASI



“PROGRAM MENENTUKAN CUACA DAERAH DI INDONESIA”

DISUSUN OLEH:

MUHAMAD HARIS HARTANTO (21083010045)

DOSEN PENGAMPU:

MOHAMMAD IDHOM, SP., S.KOM., MT.

PROGRAM STUDI SAINS DATA

FAKULTAS ILMU KOMPUTER

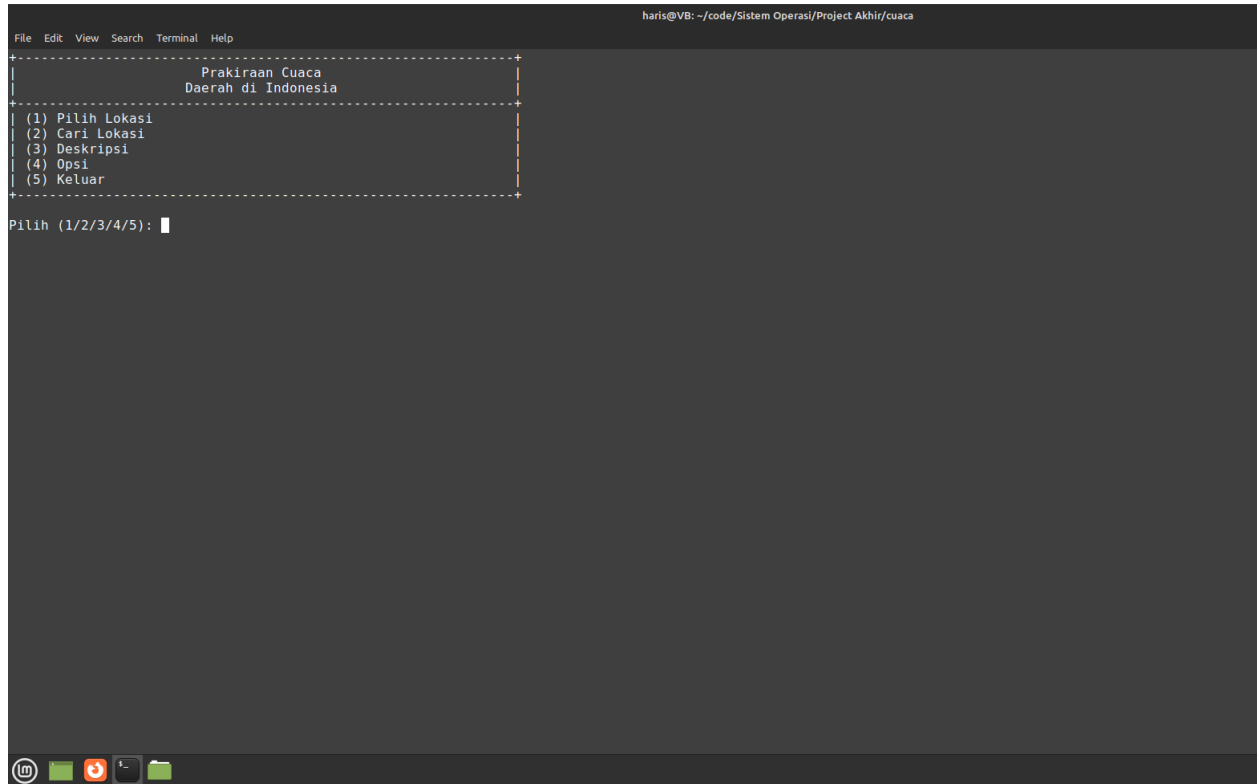
UNIVERSITAS PEMBANGUNAN NASIONAL “VETERAN” JAWA TIMUR

Jl. Rungkut Madya No.1, Gn.Anyar, Kec. Gn. Anyar, Kota SBY, Jawa Timur 60294

2022

1. Tampilan Halaman Program

1.1. Tampilan Halaman Utama



Gambar 1 Tampilan Utama Halaman Program Prakiraan Cuaca

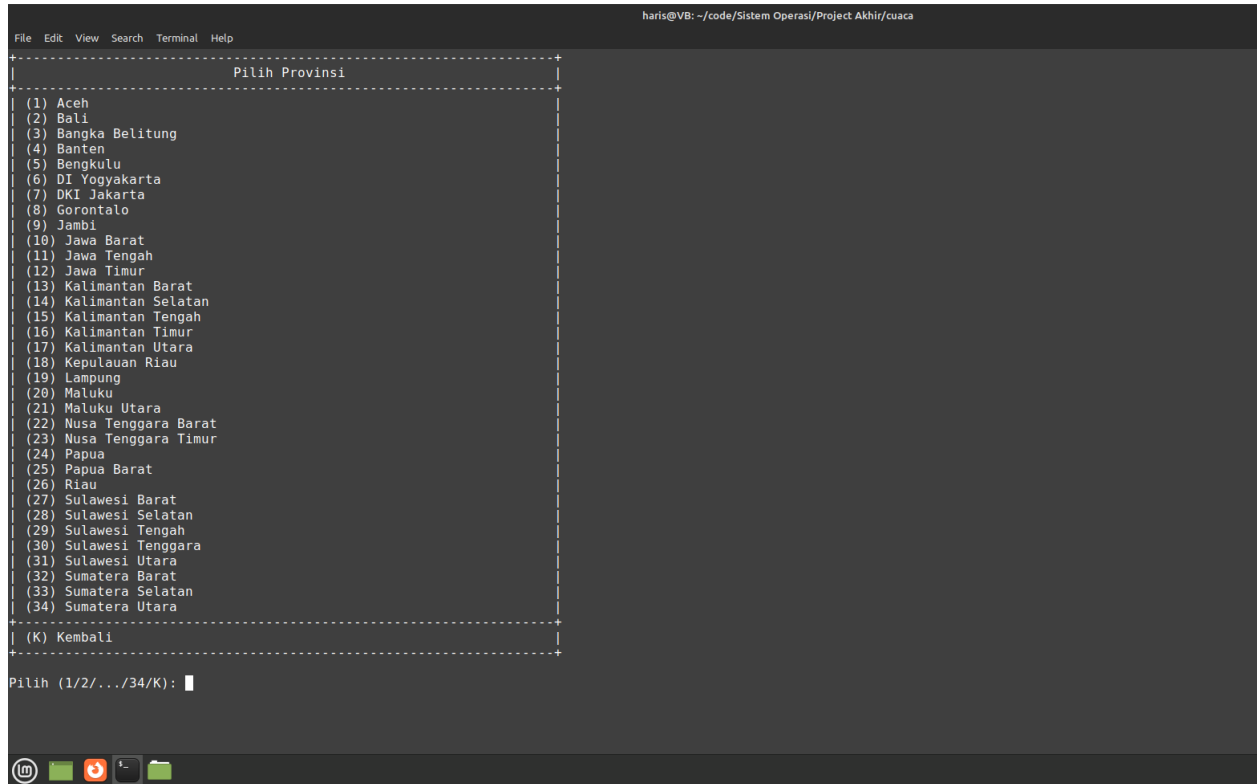
Merupakan halaman yang ditampilkan ketika program dijalankan. Diberikan lima pilihan yang dapat dipilih oleh pengguna seperti yang tertera pada gambar. Pengguna dapat melanjutkan program dengan mengisi kolom input yang disediakan dengan memasukkan nomor dari setiap pilihan yang tersedia. Apabila pengguna memasukkan nomor yang tidak sesuai dengan pilihan yang tersedia, maka akan muncul pemberitahuan berupa kalimat “Pilihan tidak valid” selama 0.5 detik dan akan ditampilkan halaman utama kembali.

Lokasi daerah yang tertera pada header mengartikan bahwa pemilihan dan pencarian lokasi mencakup lingkup wilayah tersebut. Saat pertama kali dijalankan program secara bawaan telah diatur dengan lingkup wilayah seluruh Indonesia yang menandakan dapat dilakukan pemilihan dan pencarian daerah yang berada di Indonesia sesuai dengan data yang terdapat dalam sumber data. Lingkup wilayah dapat diubah pada halaman “Opsi”.

Digunakan library PrettyTable untuk membuat tampilan tabel dan dibuat percabangan untuk setiap input yang dimasukkan oleh pengguna untuk melanjutkan atau keluar dari program.

1.2. Tampilan Halaman “Pilih Lokasi”

1.2.1. Tampilan Halaman “Pilih Provinsi”



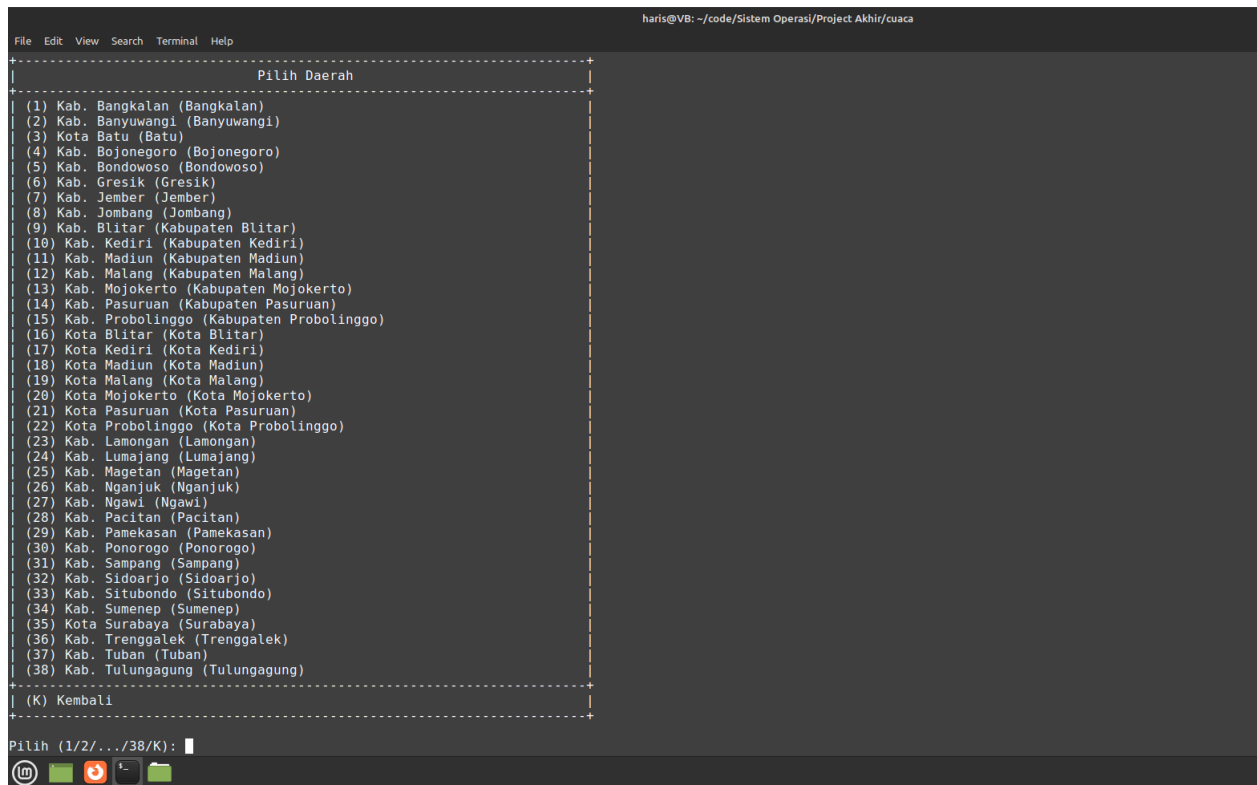
Gambar 2 Tampilan Halaman “Pilih Provinsi”

Merupakan halaman yang ditampilkan ketika pengguna memilih nomor 1 atau “Pilih Lokasi” pada halaman utama. Pada halaman ini ditampilkan daftar provinsi sesuai dengan lingkup wilayah yang ditentukan. Program secara bawaan diatur dengan lingkup wilayah seluruh Indonesia sehingga ditampilkan seluruh provinsi yang terdapat di Indonesia. Provinsi yang ditampilkan sesuai dengan data yang terdapat dalam sumber data, yaitu sebanyak 34 provinsi.

Pengguna dapat melanjutkan program dengan mengisi kolom input yang disediakan dengan memasukkan nomor dari setiap pilihan provinsi. Pengguna dapat kembali ke halaman sebelumnya (halaman utama) dengan memasukkan huruf “K” atau “k”. Apabila pengguna memasukkan nomor atau huruf yang tidak sesuai dengan pilihan yang tersedia, maka akan muncul pemberitahuan berupa kalimat “Pilihan tidak valid” selama 0.5 detik dan akan ditampilkan halaman Pilih Provinsi kembali.

Digunakan library PrettyTable untuk membuat tampilan tabel dan digunakan percabangan untuk setiap input yang dimasukkan oleh pengguna untuk melanjutkan program atau kembali ke halaman sebelumnya. Percabangan yang terdapat pada halaman ini dibuat secara dinamis menyesuaikan daftar provinsi dari lingkup wilayah yang dipilih oleh pengguna.

1.2.2. Tampilan Halaman “Pilih Daerah”



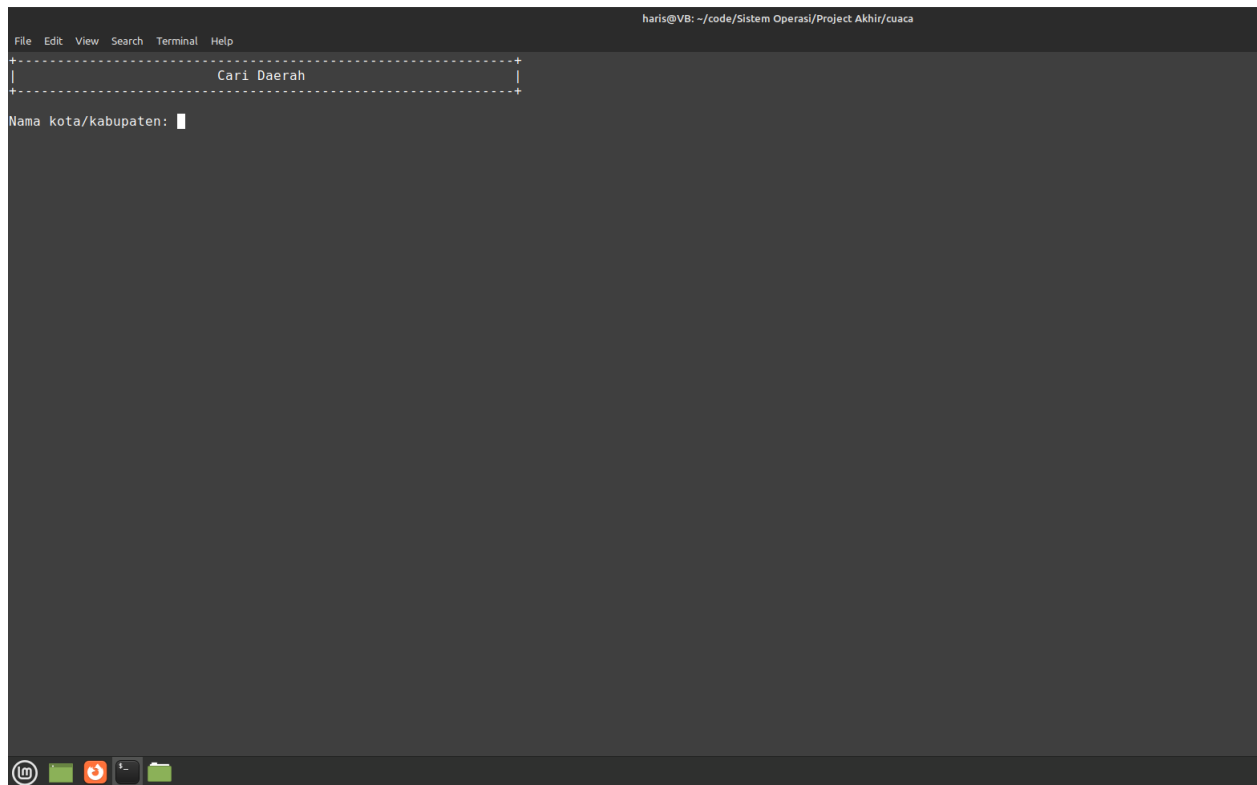
Gambar 3 Tampilan Halaman “Pilih Daerah”

Merupakan halaman yang ditampilkan setelah pengguna memilih salah satu provinsi yang tersedia pada halaman Pilih Provinsi. Pada halaman ini ditampilkan daftar kota atau kabupaten yang terdapat pada provinsi yang telah dipilih sebelumnya. Nama daerah yang tertulis dalam tanda kurung merupakan nama daerah dari ibu kota kabupaten. Berbeda dengan kota, kabupaten memiliki daerah atau wilayah yang menjadi tempat kedudukan pusat pemerintahan yang disebut dengan ibu kota kabupaten. Oleh karena itu, pada suatu kabupaten bisa memiliki nama ibu kota kabupaten yang berbeda. Tujuan dari penulisan daftar kota atau kabupaten yang disertai dengan wilayah pusat pemerintahannya adalah untuk memperjelas lokasi daerah yang akan ditampilkan data prakiraan cuacanya.

Pengguna dapat melanjutkan program dengan mengisi kolom input yang disediakan dengan memasukkan nomor dari setiap pilihan daerah. Pengguna dapat kembali ke halaman sebelumnya (halaman Pilih Provinsi) dengan memasukkan huruf “K” atau “k”. Apabila pengguna memasukkan nomor atau huruf yang tidak sesuai dengan pilihan yang tersedia, maka akan muncul pemberitahuan berupa kalimat “Pilihan tidak valid” selama 0.5 detik dan akan ditampilkan halaman Pilih Daerah kembali.

Digunakan library PrettyTable untuk membuat tampilan tabel dan digunakan percabangan untuk setiap input yang dimasukkan oleh pengguna untuk melanjutkan program atau kembali ke halaman sebelumnya. Percabangan yang terdapat pada halaman ini dibuat secara dinamis menyesuaikan daftar provinsi dari lingkup wilayah yang dipilih oleh pengguna.

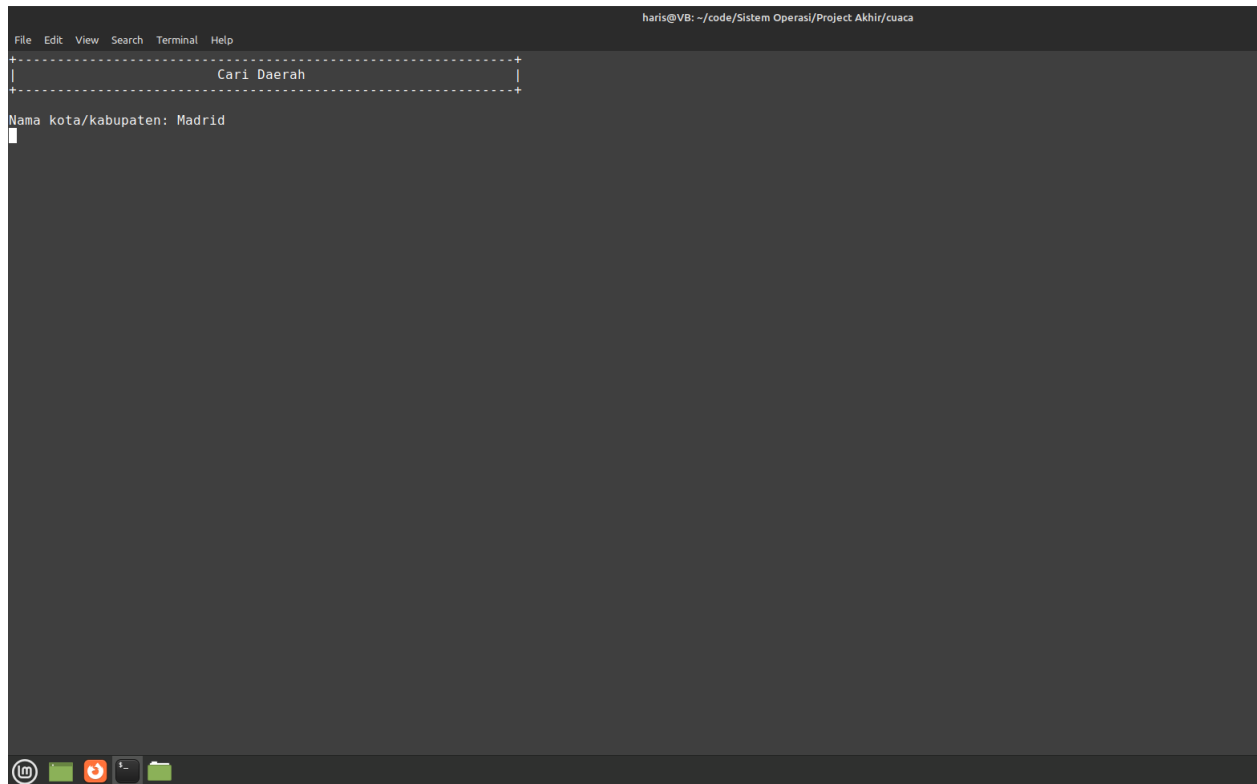
1.3. Tampilan “Cari Lokasi”



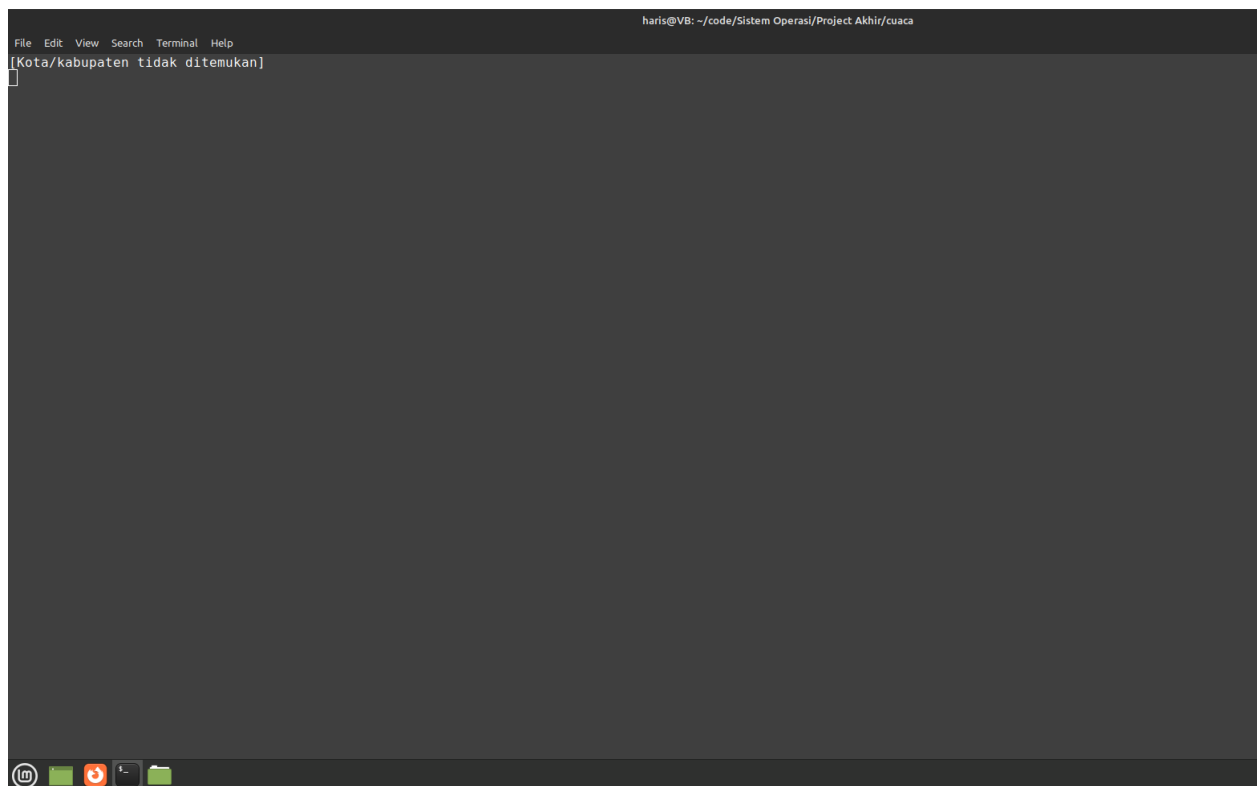
Gambar 4 Tampilan Halaman “Cari Lokasi”

Merupakan halaman yang ditampilkan ketika pengguna memilih nomor 2 atau “Cari Lokasi” pada halaman utama. Pada halaman ini diberikan kolom input yang disediakan untuk mencari suatu kota atau kabupaten yang ingin ditampilkan data prakiraan cuacanya. Pengguna dapat melanjutkan program dengan mengisi kolom input yang disediakan dengan menuliskan nama dari kota atau kabupaten (ibu kota kabupaten) yang ingin dicari.

Pencarian kota/kabupaten dilakukan dengan pencocokan hasil input pengguna dengan nama kota/kabupaten yang terdapat pada sumber data dengan melakukan perulangan for terhadap value dictionary dari list data yang berisi seluruh nama kota/kabupaten dari lingkup wilayah yang dipilih oleh pengguna. Apabila ditemukan nama kota/kabupaten yang sesuai, maka dilakukan ekstraksi ID dan nama kota/kabupaten yang merupakan key dan value dari dictionary data yang berisi seluruh ID dan nama kota/kabupaten untuk menjadi argumen pada function yang berfungsi untuk menampilkan data prakiraan cuaca pada halaman selanjutnya.



Gambar 5 Input Pengguna pada Halaman “Cari Lokasi”

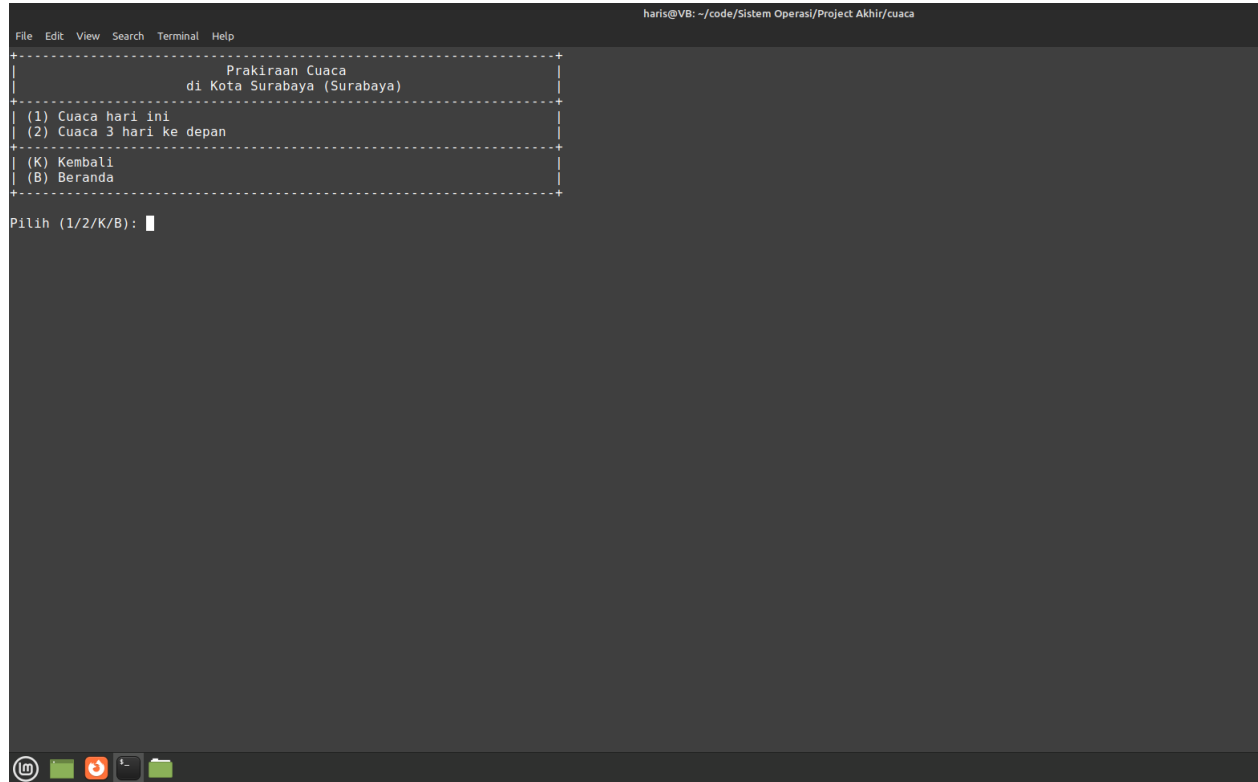


Gambar 6 Pemberitahuan Pencarian Tidak Ditemukan

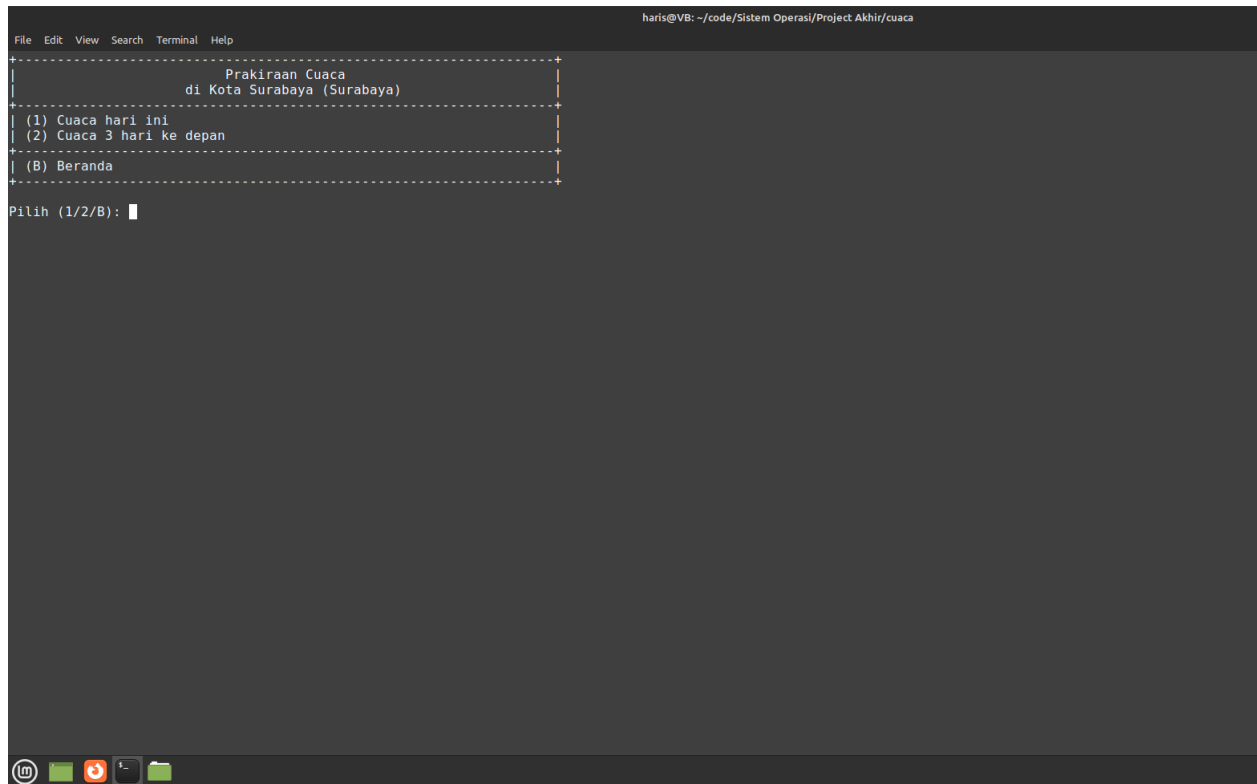
Data kota atau kabupaten yang tersedia bergantung pada opsi lingkup wilayah yang dipilih. Opsi tersebut dapat diketahui dengan melihat lokasi daerah yang tertulis pada header di halaman utama. Semisal opsi lingkup wilayah yang dipilih dan digunakan adalah Pulau Kalimantan, maka pencarian terhadap Kota Surabaya tidak akan ditemukan karena

kota tersebut berada di luar lingkup wilayah Pulau Kalimantan. Apabila dilakukan pencarian terhadap kota atau kabupaten yang berada di luar opsi lingkup wilayah yang digunakan, maka akan ditampilkan pemberitahuan berupa kalimat “Kota/kabupaten tidak ditemukan”.

1.4. Tampilan Halaman “Prakiraan Cuaca”



Gambar 7 Tampilan Halaman “Prakiraan Cuaca” dari Menu Pemilihan Kota/Kabupaten

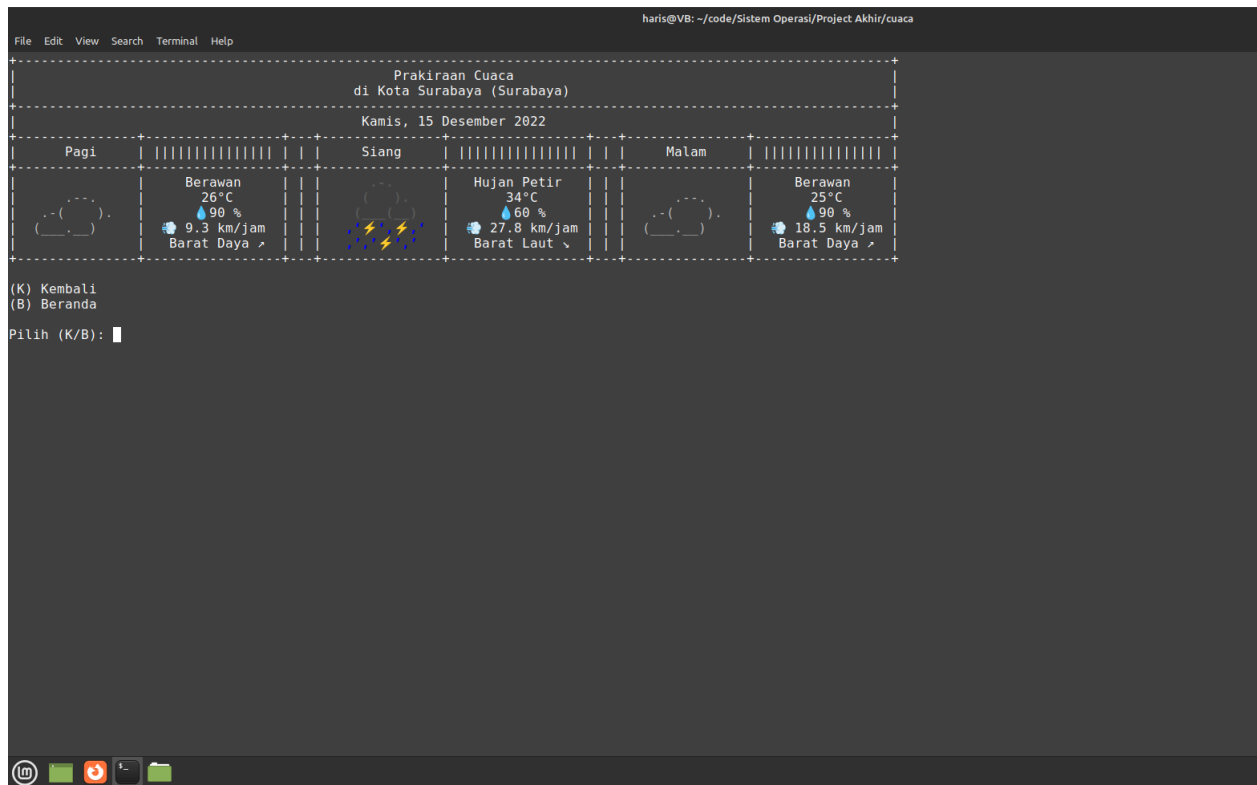


Gambar 8 Tampilan Halaman “Prakiraan Cuaca” dari Menu Pencarian Lokasi

Merupakan halaman yang ditampilkan setelah pengguna melakukan pemilihan daerah (kota/kabupaten) atau berhasil melakukan pencarian kota/kabupaten. Diberikan empat pilihan apabila pengguna sebelumnya melakukan pemilihan daerah dan diberikan tiga pilihan apabila pengguna sebelumnya melakukan pencarian kota/kabupaten. Pengguna dapat melanjutkan program dengan mengisi kolom input yang disediakan dengan memasukkan nomor dari setiap pilihan yang tersedia. Pengguna juga dapat kembali ke halaman sebelumnya (halaman Pilih Daerah) dengan memasukkan huruf “K” atau “k”. Apabila pengguna memasukkan nomor atau huruf yang tidak sesuai dengan pilihan yang tersedia, maka akan muncul pemberitahuan berupa kalimat “Pilihan tidak valid” selama 0.5 detik dan akan ditampilkan halaman Prakiraan Cuaca kembali.

Digunakan library PrettyTable untuk membuat tampilan tabel dan digunakan percabangan untuk setiap input yang dimasukkan oleh pengguna untuk melanjutkan program.

1.4.1. Tampilan Halaman “Prakiraan Cuaca Hari Ini”

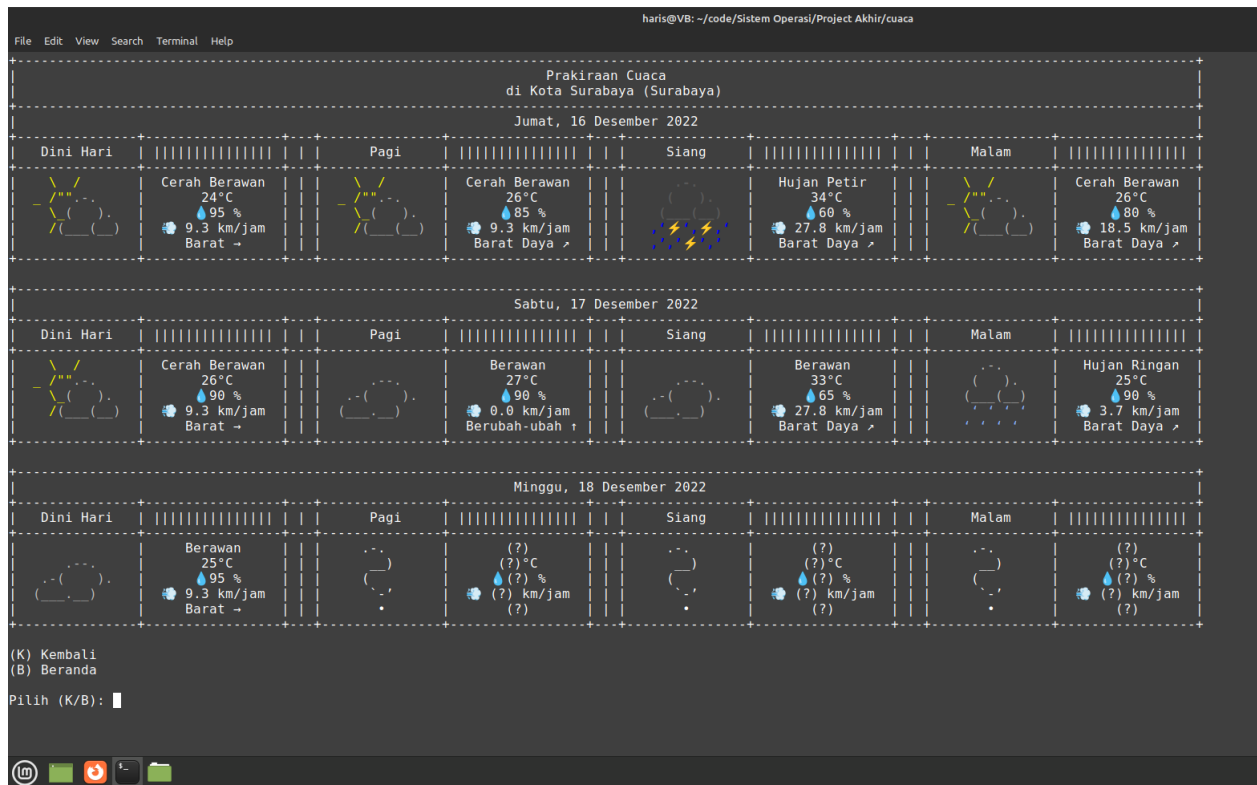


Gambar 9 Tampilan Halaman “Prakiraan Cuaca Hari Ini”

Merupakan halaman yang ditampilkan ketika pengguna memilih nomor 1 atau “Cuaca hari ini” pada halaman Prakiraan Cuaca. Pada halaman ini ditampilkan data prakiraan cuaca pada waktu pagi, siang, dan malam di hari program ini dijalankan sesuai dengan daerah yang dipilih atau dicari oleh pengguna. Pengguna dapat kembali ke halaman sebelumnya (halaman Prakiraan Cuaca) dengan memasukkan huruf “K” atau “k” dan kembali ke halaman utama dengan memasukkan huruf “B” atau “b. Apabila pengguna memasukkan huruf yang tidak sesuai dengan pilihan yang tersedia, maka akan muncul pemberitahuan berupa kalimat “Pilihan tidak valid” selama 0.5 detik dan akan ditampilkan halaman Pilih Provinsi kembali.

Digunakan library PrettyTable untuk membuat tampilan tabel dan digunakan perulangan for terhadap data prakiraan cuaca pada kota/kabupaten yang terpilih untuk mengisi sebuah list yang akan ditransformasikan menjadi wujud tabel seperti yang ditunjukkan pada Gambar 9.

1.4.2. Tampilan Halaman “Prakiraan Cuaca 3 Hari ke Depan”



Prakiraan Cuaca di Kota Surabaya (Surabaya)							
Jumat, 16 Desember 2022							
Dini Hari	Pagi	Siang	Malam	Dini Hari	Pagi	Siang	Malam
Cerah Berawan 24°C 95 % 9.3 km/jam Barat →	Cerah Berawan 26°C 85 % 9.3 km/jam Barat Daya ↗	Hujan Petir 34°C 60 % 27.8 km/jam Barat Daya ↗	Cerah Berawan 26°C 80 % 18.5 km/jam Barat Daya ↗	Cerah Berawan 26°C 90 % 9.3 km/jam Barat →	Berawan 27°C 65 % 0.0 km/jam Berubah-ubah ↑	Berawan 33°C 65 % 27.8 km/jam Barat Daya ↗	Hujan Ringan 25°C 90 % 3.7 km/jam Barat Daya ↗
Sabtu, 17 Desember 2022							
Dini Hari	Pagi	Siang	Malam	Dini Hari	Pagi	Siang	Malam
Berawan 25°C 95 % 9.3 km/jam Barat →	(?) (?)°C (?) % (?) km/jam	(?) (?)°C (?) % (?) km/jam	(?) (?)°C (?) % (?) km/jam	(?) (?)°C (?) % (?) km/jam	(?) (?)°C (?) % (?) km/jam	(?) (?)°C (?) % (?) km/jam	(?) (?)°C (?) % (?) km/jam
Minggu, 18 Desember 2022							
Dini Hari	Pagi	Siang	Malam	Dini Hari	Pagi	Siang	Malam
Berawan 25°C 95 % 9.3 km/jam Barat →	(?) (?)°C (?) % (?) km/jam	(?) (?)°C (?) % (?) km/jam	(?) (?)°C (?) % (?) km/jam	(?) (?)°C (?) % (?) km/jam	(?) (?)°C (?) % (?) km/jam	(?) (?)°C (?) % (?) km/jam	(?) (?)°C (?) % (?) km/jam

(K) Kembali
(B) Beranda
Pilih (K/B):

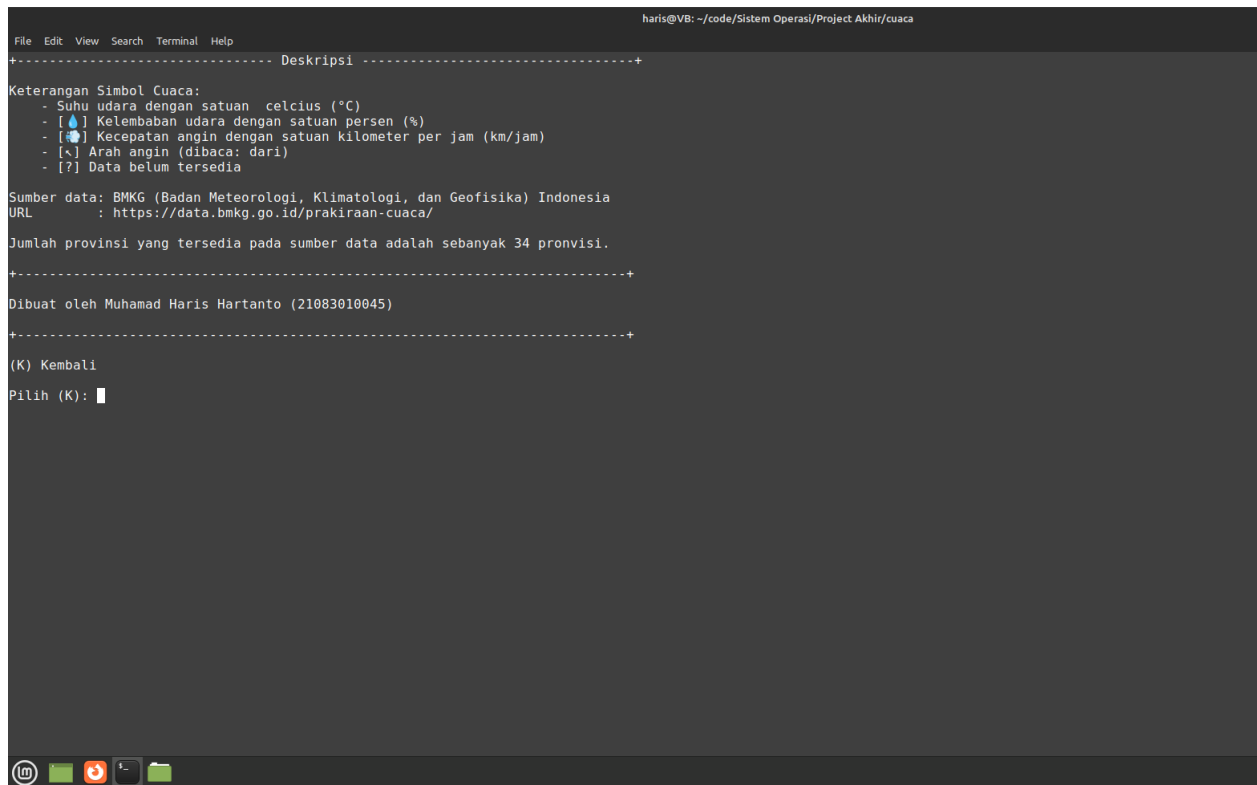
Gambar 10 Halaman “Prakiraan Cuaca 3 Hari ke Depan”

Merupakan halaman yang ditampilkan ketika pengguna memilih nomor 2 atau “Cuaca 3 hari ke depan” pada halaman Prakiraan Cuaca. Pada halaman ini ditampilkan data prakiraan cuaca pada waktu dini hari, pagi, siang, dan malam untuk tiga hari ke depan dari hari program ini dijalankan sesuai dengan daerah yang dipilih atau dicari oleh pengguna. Khusus untuk hari ketiga, data yang tersedia hanya pada waktu dini hari saja.

Pengguna dapat kembali ke halaman sebelumnya (halaman Prakiraan Cuaca) dengan memasukkan huruf “K” atau “k” dan kembali ke halaman utama dengan memasukkan huruf “B” atau “b. Apabila pengguna memasukkan huruf yang tidak sesuai dengan pilihan yang tersedia, maka akan muncul pemberitahuan berupa kalimat “Pilihan tidak valid” selama 0.5 detik dan akan ditampilkan halaman Pilih Provinsi kembali

Digunakan library PrettyTable untuk membuat tampilan tabel dan digunakan perulangan for terhadap data prakiraan cuaca pada kota/kabupaten yang terpilih untuk mengisi sebuah list yang akan ditransformasikan menjadi wujud tabel seperti yang ditunjukkan pada Gambar 10.

1.5. Tampilan Halaman “Deskripsi”



```
haris@VB: ~/code/Sistem Operasi/Project Akhir/cuaca
File Edit View Search Terminal Help
+----- Deskripsi -----+
Keterangan Simbol Cuaca:
- Suhu udara dengan satuan celcius (°C)
- [💧] Kelembaban udara dengan satuan persen (%)
- [💨] Kecepatan angin dengan satuan kilometer per jam (km/jam)
- [↻] Arah angin (dibaca: dari)
- [?] Data belum tersedia

Sumber data: BMKG (Badan Meteorologi, Klimatologi, dan Geofisika) Indonesia
URL          : https://data.bmkg.go.id/prakiraan-cuaca/

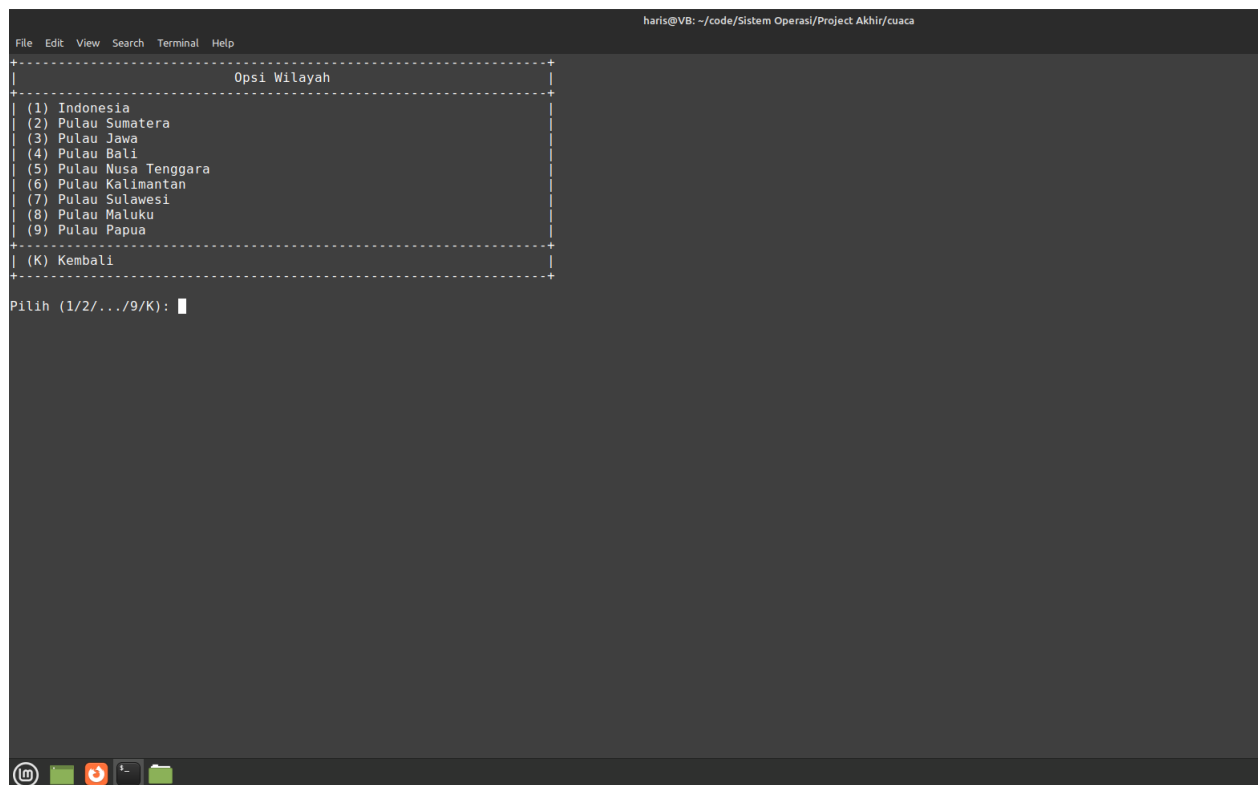
Jumlah provinsi yang tersedia pada sumber data adalah sebanyak 34 provinsi.
+-----+
Dibuat oleh Muhamad Haris Hartanto (21083010045)
+-----+

(K) Kembali
Pilih (K): █
```

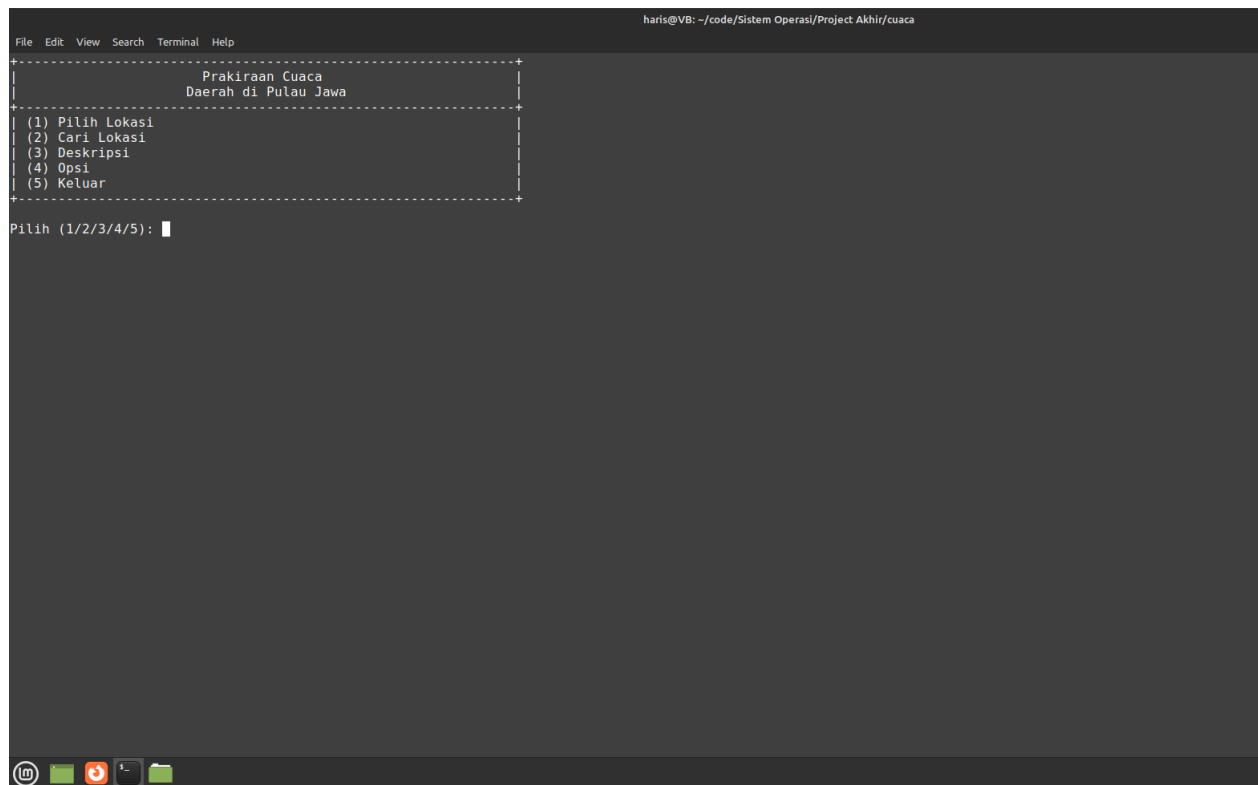
Gambar 11 Tampilan Halaman Deskripsi

Merupakan halaman yang ditampilkan ketika pengguna memilih nomor 3 atau “Deskripsi” pada halaman utama. Pada halaman ini terdapat penjelasan mengenai arti dari simbol yang terdapat pada tampilan prakiraan cuaca, informasi sumber data, dan pembuat program ini. Pengguna dapat kembali ke halaman sebelumnya (halaman utama) dengan memasukkan huruf “K” atau “k”. Apabila pengguna memasukkan nomor atau huruf yang tidak sesuai dengan pilihan yang tersedia, maka akan muncul pemberitahuan berupa kalimat “Pilihan tidak valid” selama 0.5 detik dan akan ditampilkan halaman Pilih Provinsi kembali.

1.6. Tampilan “Opsi”



Gambar 12 Tampilan Halaman Opsi



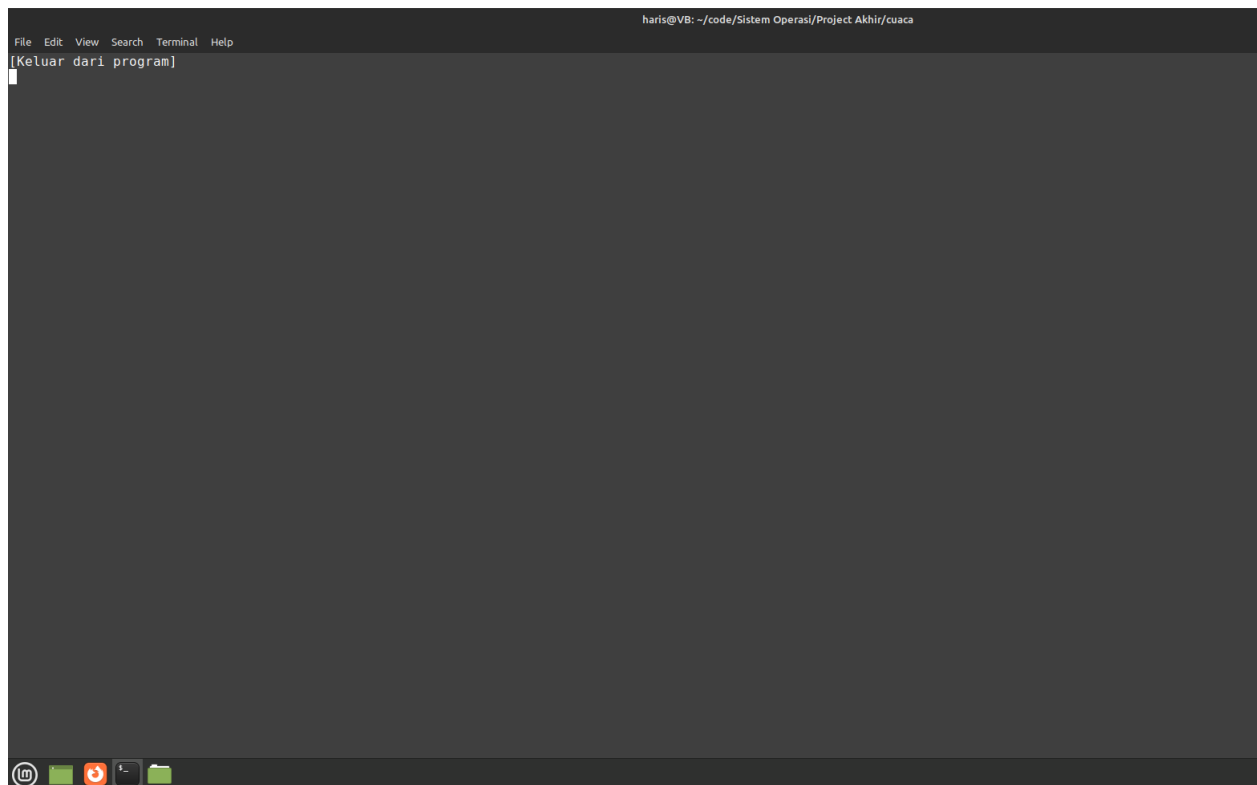
Gambar 13 Tampilan Halaman Utama Setelah Berganti Lingkup Wilayah

Merupakan halaman yang ditampilkan ketika pengguna memilih nomor 4 atau “Opsi” pada halaman utama. Pada halaman ini ditampilkan daftar lingkup wilayah yang dapat digunakan oleh pengguna. Program secara bawaan diatur

dengan lingkup wilayah seluruh Indonesia. Pengguna dapat melanjutkan program dengan mengisi kolom input yang disediakan dengan memasukkan nomor dari setiap pilihan opsi wilayah. Setelah dilakukan pemilihan program akan kembali ke halaman utama dengan judul pada header sesuai dengan opsi lingkup wilayah yang dipilih. Pengguna dapat kembali ke halaman sebelumnya (halaman utama) dengan memasukkan huruf “K” atau “k”. Apabila pengguna memasukkan nomor atau huruf yang tidak sesuai dengan pilihan yang tersedia, maka akan muncul pemberitahuan berupa kalimat “Pilihan tidak valid” selama 0.5 detik dan akan ditampilkan halaman “Opsi” kembali.

Digunakan library PrettyTable untuk membuat tampilan tabel dan dibuat percabangan untuk setiap input yang dimasukkan oleh pengguna untuk melanjutkan program atau kembali ke halaman sebelumnya. Digunakan library PrettyTable untuk membuat tampilan tabel dan digunakan perulangan for terhadap data prakiraan cuaca pada kota/kabupaten yang terpilih untuk mengisi sebuah list yang akan ditransformasikan menjadi wujud tabel seperti yang ditunjukkan pada Gambar 13. Digunakan library PrettyTable untuk membuat tampilan tabel dan digunakan percabangan untuk setiap input yang dimasukkan oleh pengguna untuk melanjutkan program.

1.7. Tampilan “Keluar”



Gambar 14 Pemberitahuan Pengguna Telah Keluar dari Program

Ditampilkan pemberitahuan “Keluar dari program” selama 0.5 detik apabila pengguna memilih nomor 5 atau “Keluar” pada halaman utama. Pilihan ini digunakan untuk keluar dari program.

2. Kode Program

Keseluruhan kode program dapat dilihat melalui repository GitHub pada laman berikut: <https://github.com/harishartanto/21083010045/tree/master/Project%20Akhir>.

2.1. File main.py

Berisikan baris kode yang memiliki fungsi untuk membuat tampilan dan alur program serta algoritma dalam proses pengolahan data untuk ditampilkan kepada pengguna.

```
import os
import time
from prettytable import PrettyTable
from datetime import datetime, timedelta
from constants import *

loc = 'Indonesia'
province_url = data_cleaning(get_province())

def home():
    header(loc)
    table = table_header()
    table.add_row(['(1) Pilih Lokasi\n(2) Cari Lokasi\n(3) Deskripsi\n(4) Opsi\n(5) Keluar'])
    print(table)

    jawab = input('\nPilih (1/2/3/4/5): ')
    os.system('clear')

    if jawab == '1':
        table.clear_rows()
        os.system('clear')
        select_province(province_url)
    elif jawab == '2':
        table.clear_rows()
        loc_search(province_url)
    elif jawab == '3':
        table.clear_rows()
        os.system('clear')
        description()
    elif jawab == '4':
        table.clear_rows()
```

```

        os.system('clear')
        options()
    elif jawab == '5':
        table.clear_rows()
        os.system('clear')
        print('Keluar dari program')
        time.sleep(0.5)
        os.system('clear')
        exit()
    else:
        table.clear_rows()
        invalid_selection()
        home()

def description():
    print('+----- Dekripsi -----')
    print('-----+\n')
    print('''Keterangan Simbol Cuaca:
- Suhu udara dengan satuan celcius (°C)
- [💧] Kelembaban udara dengan satuan persen (%)
- [🌀] Kecepatan angin dengan satuan kilometer per jam (km/jam)
- [^] Arah angin (dibaca: dari)
- [?] Data belum tersedia
''')
    print('Sumber data: BMKG (Badan Meteorologi, Klimatologi, dan Geofisika)
Indonesia\n'+
        'URL      : https://data.bmkg.go.id/prakiraan-cuaca/\n\nJumlah
provinsi yang tersedia pada sumber data adalah sebanyak 34 provinsi.')
    print('\n+ '+'-'*76+'+\n')
    print('Dibuat oleh Muhamad Haris Hartanto (21083010045)')
    print('\n+ '+'-'*76+'+\n')
    print('(K) Kembali')

    jawab = input('\nPilih (K): ').lower()
    os.system('clear')

    if jawab == 'k':
        home()
    else:

```

```

        invalid_selection()
        description()

def options():
    table = table_header('Opsi Lokasi')

    global loc
    global province_url

    options_list = []
    n = 1
    for k in loc_filter.keys():
        o = f'({n}) {k}'
        options_list.append(o)
        n += 1
    join_options = '\n'.join(options_list)

    table.add_row([join_options])
    print(table)
    table.add_row(['(K) Kembali'])
    print( "\n".join(table.get_string().splitlines()[-2:]) )

    jawab = input(f'\nPilih (1/2/.../{n-1}/K): ')
    os.system('clear')

    if jawab == 'K' or jawab == 'k':
        table.clear_rows()
        home()
    elif jawab.isnumeric():
        if int(jawab) in range(1, n):
            loc = list(loc_filter.keys())[int(jawab)-1]
            province_url = loc_selection(loc_filter, loc)
            table.clear_rows()
            home()
        else:
            table.clear_rows()
            invalid_selection()
            options()
    else:

```



```

        table.clear_rows()
        invalid_selection()
        options()

def loc_search(province_dict):
    jawab = input('Nama kota/kabupaten: ').lower()

    city = []
    for _, v in province_dict.items():
        i, _ = get_city(v)
        city.append(i)

    cid_list = {}
    for i in city:
        for k, v in i.items():
            x = re.search(r'\((.*?)\)', v)
            if (x.group(1)).lower() == jawab:
                cid_list[k] = v

    try:
        for i, v in enumerate(city):
            if list(cid_list.keys())[0] in v:
                prov_idx = i
    except IndexError:
        os.system('clear')
        print('[Kota/kabupaten tidak ditemukan]')
        time.sleep(1)
        os.system('clear')
        home()

    url = list(province_dict.values())[prov_idx]
    response = requests.get(url)
    r = response.text
    data = bs(r, 'xml')

    os.system('clear')
    if len(cid_list) == 0:
        print('[Kota/kabupaten tidak ditemukan]')
        time.sleep(1)
        os.system('clear')

```

```

        home()
    else:
        weather(list(cid_list.keys())[0], list(cid_list.values())[0], data)

def select_province(province_dict):
    table = table_header('Pilih Provinsi')

    province_list = []
    n = 1
    for k in province_dict.keys():
        p = f'({n}) {k}'
        province_list.append(p)
        n += 1
    join_province = '\n'.join(province_list)

    table.add_row([join_province])
    print(table)
    table.add_row(['(K) Kembali'])
    print( "\n".join(table.get_string().splitlines()[-2:]) )

    if n == 1:
        jawab = input('\nPilih (1/K): ')
    elif n == 2:
        jawab = input('\nPilih (1/2/K): ')
    else:
        jawab = input(f'\nPilih (1/2/.../{n-1}/K): ')
    os.system('clear')

    if jawab == 'K' or jawab == 'k':
        table.clear_rows()
        home()
    elif jawab.isnumeric():
        if int(jawab) in range(1, n):
            url = province_dict[list(province_dict.keys())[int(jawab)-1]]
            x, y = get_city(url)
            table.clear_rows()
            select_city(x, y)
        else:
            table.clear_rows()

```

```

        invalid_selection()
        select_province(province_dict)
    else:
        table.clear_rows()
        invalid_selection()
        select_province(province_dict)

def get_city(url):
    response = requests.get(url)
    r = response.text
    data = bs(r, 'xml')
    city = data.find_all('area')
    city_n = data.find_all('name', {'xml:lang': 'id_ID'})

    city_dict = {}
    city_list = []

    outside_dom_idx = []
    n = 0
    for i in city:
        if i['tags'] == '':
            k, v = i['id'], i['description']
            city_dict[k] = v
            n += 1
        else:
            n += 1
            outside_dom_idx.append(n-1)

    for i in city_n:
        city_list.append(i.text)

    city_list = [city_list[i] for i in range(len(city_list)) if i not in
outside_dom_idx]

    city_dict = {k: city_list[i]+f' ({v})' for i, (k, v) in
enumerate(city_dict.items())}

    return city_dict, data

```

```

def select_city(city_dict, data):
    table = table_header('Pilih Daerah')
    daerah_list = []
    n = 1
    for v in city_dict.values():
        d = f'({n}) {v}'
        daerah_list.append(d)
        n += 1
    join_daerah = '\n'.join(daerah_list)

    table.add_row([join_daerah])
    print(table)
    table.add_row(['(K) Kembali'])
    print( "\n".join(table.get_string().splitlines()[-2:]))

    jawab = input(f'\nPilih (1/2/.../{n-1}/K): ').lower()
    os.system('clear')

    if jawab == 'k':
        table.clear_rows()
        os.system('clear')
        select_province(province_url)
    elif jawab.isnumeric():
        if int(jawab) in range(1, n):
            key, value = list(city_dict.items())[int(jawab)-1]
            table.clear_rows()
            weather(key, value, data, city_dict)
        else:
            table.clear_rows()
            invalid_selection()
            select_city(city_dict, data)
    else:
        table.clear_rows()
        invalid_selection()
        select_city(city_dict, data)

def weather(city_id, city_n, data, city_dict=None):
    header(city_n, length=67)
    table = table_header()

```

```

hourly_list = ['0', '6', '12', '18', '24', '30', '36', '42', '48', '54',
'60', '66']
param_id = ['weather', 't', 'hu', 'ws']

weather_list = []
i = 0
for p in param_id:
    weather_list.append([])
    param_i = data.find(id=city_id).find(id=p)
    for h in hourly_list:
        h_i = param_i.find(h=h).value.string
        weather_list[i].append(h_i)
    i += 1

wind_direction = data.find(id=city_id).find(id='wd')
wind_dir_list = []
for wd in hourly_list:
    wdh = wind_direction.find(h=wd).find(unit='CARD').text
    wind_dir_list.append(wdh)

if city_dict == None:
    table.add_row(['(1) Cuaca hari ini\n(2) Cuaca 3 hari kedepan'])
    print(table)
    table.add_row(['(B) Beranda'])
    print( "\n".join(table.get_string().splitlines()[-2:]) )
else:
    table.add_row(['(1) Cuaca hari ini\n(2) Cuaca 3 hari kedepan'])
    print(table)
    table.add_row(['(K) Kembali\n(B) Beranda'])
    print( "\n".join(table.get_string().splitlines()[-3:]) )

if city_dict == None:
    jawab = input('\nPilih (1/2/B): ').lower()
else:
    jawab = input('\nPilih (1/2/K/B): ').lower()
os.system('clear')

if jawab == '1':

```

```

        table.clear_rows()
        td_weather(weather_list, wind_dir_list, city_id, city_n, data,
city_dict)
    elif jawab == '2':
        table.clear_rows()
        tm_weather(weather_list, wind_dir_list, city_id, city_n, data,
city_dict)
    elif city_dict == None and jawab == 'b':
        table.clear_rows()
        home()
    elif city_dict != None and jawab == 'k':
        table.clear_rows()
        select_city(city_dict, data)
    elif city_dict != None and jawab == 'b':
        table.clear_rows()
        home()
    else:
        table.clear_rows()
        invalid_selection()
        weather(city_id, city_n, data, city_dict)

def td_weather(weather_list, wind_dir_list, city_id, city_n, data,
city_dict=None):
    header(city_n, '', 109)
    date_time = datetime.now().strftime("%d %B %Y")
    date_time = date_time.replace(date_time[3:-5], month_id[date_time[3:-5]])
    day_name = datetime.now().strftime("%A")

    ver_grid = '|\\n|\\n|\\n|\\n|'
    td_list = []
    for i in range(0, 3):
        if i != 2:
            td_list.append([get_symbol(weather_list[0][i], weather_symbols),
f'{weather_code[weather_list[0][i]]}\\n{weather_list[1][i]}°C\\n💧{weather_list[2]
[i]} %\\n🌀 {knot_to_kmh(weather_list[3][i])}
km/jam\\n{wind_d_code[wind_dir_list[i]]}', ver_grid])
        else:
            td_list.append([get_symbol(weather_list[0][i], weather_symbols),
f'{weather_code[weather_list[0][i]]}\\n{weather_list[1][i]}°C\\n💧{weather_list[2]

```

```

[i]} %\n↵ {knot_to_kmh(weather_list[3][i])}
km/jam\n{wind_d_code[wind_dir_list[i]]}')])

table = PrettyTable()
table._validate_field_names = lambda *a, **k: None
table.title = f'{day_id[day_name]}, {date_time}'
table.field_names = ['Pagi', '|' * 15, '|', 'Siang', '|' * 15, '|', 'Malam',
'|' * 15]
table.add_row([td_list[0][0], td_list[0][1], td_list[0][2],
td_list[1][0], td_list[1][1], td_list[1][2],
td_list[2][0], td_list[2][1]])
print(table)

print('\n(K) Kembali\n(B) Beranda')

jawab = input('\nPilih (K/B): ').lower()
os.system('clear')

if jawab == 'k':
    weather(city_id, city_n, data, city_dict)
elif jawab == 'b':
    home()
else:
    invalid_selection()
    td_weather(weather_list, wind_dir_list, city_id, city_n, data,
city_dict)

def tm_weather(weather_list, wind_dir_list, city_id, city_n, data,
city_dict=None):
    header(city_n, '', 147)
    for i in range(0,3):
        for j in range(len(weather_list)):
            weather_list[j].append('(?)')

    for i in range(0,3):
        wind_dir_list.append('(?)')

ver_grid = '| \n| \n| \n| \n| '
n_list = [[3, 7], [7, 11], [11, 15]]

```

```

def tm_data(i, n):
    tm_list = []
    for i in range(i, n):
        if i != n-1:
            tm_list.append([get_symbol(weather_list[0][i],
weather_symbols),
f'{weather_code[weather_list[0][i]]}\n{weather_list[1][i]}°C\n💧{weather_list[2]
[i]} %\n☞ {knot_to_kmh(weather_list[3][i])}
km/jam\n{wind_d_code[wind_dir_list[i]]}', ver_grid])
        else:
            tm_list.append([get_symbol(weather_list[0][i],
weather_symbols),
f'{weather_code[weather_list[0][i]]}\n{weather_list[1][i]}°C\n💧{weather_list[2]
[i]} %\n☞ {knot_to_kmh(weather_list[3][i])}
km/jam\n{wind_d_code[wind_dir_list[i]]}'])
    return tm_list

def tm_table(tm_list, day_name, date_time):
    table = PrettyTable()
    table._validate_field_names = lambda *a, **k: None
    table.title = f'{day_id[day_name]}, {date_time}'
    table.field_names = ['Dini Hari', '| '*15, '|', 'Pagi', '| '*15, '|',
'Siang', '| '*15, '|', 'Malam', '| '*15]
    table.add_row([tm_list[0][0], tm_list[0][1], tm_list[0][2],
tm_list[1][0], tm_list[1][1], tm_list[1][2],
tm_list[2][0], tm_list[2][1], tm_list[2][2],
tm_list[3][0], tm_list[3][1]])

    print(table, '\n')

for (i, j), x in zip(n_list, range(1, 4)):
    tm_date_time = datetime.now() + timedelta(days=x)
    day_name = datetime.now() + timedelta(days=x)

    tm_date_time = tm_date_time.strftime("%d %B %Y")
    tm_date_time = tm_date_time.replace(tm_date_time[3:-5],
month_id[tm_date_time[3:-5]])
    day_name = day_name.strftime("%A")

```



```

        z = tm_data(i, j)
        tm_table(z, day_name, tm_date_time)

print('(K) Kembali\n(B) Beranda')

jawab = input('\nPilih (K/B): ').lower()
os.system('clear')

if jawab == 'k':
    os.system('clear')
    weather(city_id, city_n, data, city_dict)
elif jawab == 'b':
    os.system('clear')
    home()
else:
    invalid_selection()
    tm_weather(weather_list, wind_dir_list, city_id, city_n, data,
city_dict)

def header(place, place_2='Daerah', length=62):
    print('+ '+'-'*length+'+')
    print('| '+'Prakiraan Cuaca'.center(length)+'|')
    print('| '+f'{place_2} di {place}'.center(length)+'|')

def table_header(title=None):
    table = PrettyTable()
    if title == None:
        title = ''

    table.header = False
    table.title = title
    table.align = 'l'
    table._min_table_width = 75

    return table

def invalid_selection():
    os.system('clear')

```

```
print('[Pilihan tidak valid]')
time.sleep(0.5)
os.system('clear')

def loc_selection(loc_filter, key):
    province_url = data_cleaning(get_province())
    province_url = data_filtering(province_url, loc_filter[key])
    return province_url

def get_symbol(id, weather):
    symbol = '\n'.join(weather[id])
    return symbol

def knot_to_kmh(knots):
    try:
        kmh = round(float(knots) * 1.852, 1)
    except ValueError:
        kmh = knots
    return kmh

if __name__ == '__main__':
    os.system('clear')
    home()
```

2.2. File get_data.py

Berisikan baris kode yang memiliki fungsi untuk melakukan *scraping* data pada situs Data Terbuka BMKG terhadap data prakiraan cuaca untuk 34 provinsi Indonesia.

```
import re
import requests
from bs4 import BeautifulSoup as bs

URL = 'https://data.bmkg.go.id/prakiraan-cuaca/'

response = requests.get(URL)
html_page = bs(response.content, "html.parser")

all_links = html_page.find_all("a")

def get_province():
    province_url = {}
    for link in all_links:
        href = link['href']
        if href.startswith('../DataMKG/MEWS/DigitalForecast/DigitalForecast-'):
            province_url[link.text] = href
    return province_url

def data_cleaning(data):
    for key, value in list(data.items()):
        data[key] = value.replace('../', 'https://data.bmkg.go.id/')
        data[key.replace('DigitalForecast-', '').replace('.xml', '')] = data.pop(key)

    for key, value in list(data.items()):
        data[re.sub(r'([A-Z]) (?=[A-Z] [a-z]) | ([a-z]) (?=[A-Z]) ', r'\1\2 ', key)] = data.pop(key)

    del data['Indonesia']

    return data
```

```
def data_filtering(data, filter_list):
    for key, _ in list(data.items()):
        if key not in filter_list:
            del data[key]
    return data
```

2.3. File constanst.py

Berisikan baris kode yang memiliki fungsi untuk menyimpan konstanta yang diperlukan oleh program, seperti simbol cuaca, kode cuaca beserta maknanya, hingga terjemahan bahasa inggris-indonesia untuk penanggalan.

```
from get_data import *

loc_filter = {'Indonesia': list(data_cleaning(get_province()).keys()),
              'Pulau Sumatera': ['Aceh', 'Bangka Belitung', 'Bengkulu',
                                  'Jambi', 'Kepulauan Riau', 'Lampung', 'Riau', 'Sumatera Barat', 'Sumatera
Selatan', 'Sumatera Utara'],
              'Pulau Jawa': ['Banten', 'DI Yogyakarta', 'DKI Jakarta', 'Jawa
Barat', 'Jawa Tengah', 'Jawa Timur'],
              'Pulau Bali': ['Bali'],
              'Pulau Nusa Tenggara': ['Nusa Tenggara Barat', 'Nusa Tenggara
Timur'],
              'Pulau Kalimantan': ['Kalimantan Barat', 'Kalimantan Selatan',
                                    'Kalimantan Tengah', 'Kalimantan Timur', 'Kalimantan Utara'],
              'Pulau Sulawesi': ['Gorontalo', 'Sulawesi Barat', 'Sulawesi
Selatan', 'Sulawesi Tengah', 'Sulawesi Tenggara', 'Sulawesi Utara'],
              'Pulau Maluku': ['Maluku', 'Maluku Utara'],
              'Pulau Papua': ['Papua', 'Papua Barat']}]

weather_code = {'0': 'Cerah',
                 '1': 'Cerah Berawan',
                 '2': 'Cerah Berawan',
                 '3': 'Berawan',
                 '4': 'Berawan Tebal',
                 '5': 'Udara Kabur',
                 '10': 'Asap',
                 '45': 'Kabut',
                 '60': 'Hujan Ringan',
                 '61': 'Hujan Sedang',
                 '63': 'Hujan Lebat',
```

```
'80': 'Hujan Lokal',  
'95': 'Hujan Petir',  
'97': 'Hujan Petir',  
'(?)': '({})'
```

```
wind_d_code = {'N': 'Utara ↓',  
               'NNE': 'Timur Laut ↙',  
               'NE': 'Timur Laut ↙',  
               'ENE': 'Timur Laut ↙',  
               'E': 'Timur ←',  
               'ESE': 'Tenggara ↖',  
               'SE': 'Tenggara ↖',  
               'SSE': 'Tenggara ↖',  
               'S': 'Selatan ↑',  
               'SSW': 'Barat Daya ↗',  
               'SW': 'Barat Daya ↗',  
               'WSW': 'Barat Daya ↗',  
               'W': 'Barat →',  
               'WNW': 'Barat Laut ↘',  
               'NW': 'Barat Laut ↘',  
               'NNW': 'Barat Laut ↘',  
               'VARIABLE': 'Berubah-ubah ↑',  
               '({})': '({})'
```

```
day_id = {'Sunday': 'Minggu',  
          'Monday': 'Senin',  
          'Tuesday': 'Selasa',  
          'Wednesday': 'Rabu',  
          'Thursday': 'Kamis',  
          'Friday': 'Jumat',  
          'Saturday': 'Sabtu'}
```

```
month_id = {'January': 'Januari',  
            'February': 'Februari',  
            'March': 'Maret',  
            'April': 'April',  
            'May': 'Mei',  
            'June': 'Juni',  
            'July': 'Juli',
```

```

    'August': 'Agustus',
    'September': 'September',
    'October': 'Oktober',
    'November': 'November',
    'December': 'Desember'}

```

```

weather_symbols = {

```

```

    "(?):" [

```

```

        "    .-."    ",
        "    __)"    ",
        "    ("    ",
        "    `-'    ",
        "    ."    "],

```

```

    "0": [

```

```

        "\033[38;5;226m    \ \    /    \033[0m",
        "\033[38;5;226m    .-."    \033[0m",
        "\033[38;5;226m    - (    ) -    \033[0m",
        "\033[38;5;226m    `-'    \033[0m",
        "\033[38;5;226m    /    \ \    \033[0m"],

```

```

    "1": [

```

```

        "\033[38;5;226m    \ \    /\033[0m    ",
        "\033[38;5;226m    _ /\\""\033[38;5;250m.-."    \033[0m",
        "\033[38;5;226m    \ \_ \033[38;5;250m(    ).    \033[0m",
        "\033[38;5;226m    /\033[38;5;250m(__(__) \033[0m",
        "    "],

```

```

    "2": [

```

```

        "\033[38;5;226m    \ \    /\033[0m    ",
        "\033[38;5;226m    _ /\\""\033[38;5;250m.-."    \033[0m",
        "\033[38;5;226m    \ \_ \033[38;5;250m(    ).    \033[0m",
        "\033[38;5;226m    /\033[38;5;250m(__(__) \033[0m",
        "    "],

```

```

    "3": [

```

```

        "    ",
        "\033[38;5;250m    .--."    \033[0m",
        "\033[38;5;250m    .-(    ).    \033[0m",
        "\033[38;5;250m    (____.)    \033[0m",
        "    "],

```

```

    "4": [

```

```

        "    ",

```

```

"\033[38;5;240;1m    .--.    \033[0m",
"\033[38;5;240;1m  .-(    ).    \033[0m",
"\033[38;5;240;1m (_____) \033[0m",
"
    "],
"5": [
"
    ",
"\033[38;5;251m _ - _ - _ - \033[0m",
"\033[38;5;251m _ - _ - _ - \033[0m",
"\033[38;5;251m _ - _ - _ - \033[0m",
"
    "],
"10": [
"
    ",
"\033[38;5;251m _ - _ - _ - \033[0m",
"\033[38;5;251m _ - _ - _ - \033[0m",
"\033[38;5;251m _ - _ - _ - \033[0m",
"
    "],
"45": [
"
    ",
"\033[38;5;251m _ - _ - _ - \033[0m",
"\033[38;5;251m _ - _ - _ - \033[0m",
"\033[38;5;251m _ - _ - _ - \033[0m",
"
    "],
"60": [
"\033[38;5;250m    .--.    \033[0m",
"\033[38;5;250m    (    ).    \033[0m",
"\033[38;5;250m    (____) \033[0m",
"\033[38;5;111m    ' ' ' ' \033[0m",
"\033[38;5;111m    ' ' ' ' \033[0m]",
"61": [
"\033[38;5;250m    .--.    \033[0m",
"\033[38;5;250m    (    ).    \033[0m",
"\033[38;5;250m    (____) \033[0m",
"\033[38;5;111m    ,',',',', \033[0m",
"\033[38;5;111m    ' ' ' ' \033[0m]",
"63": [
"\033[38;5;240;1m    .--.    \033[0m",
"\033[38;5;240;1m    (    ).    \033[0m",
"\033[38;5;240;1m    (____) \033[0m",
"\033[38;5;21;1m    ,',',',', \033[0m",

```

```
"\033[38;5;21;1m ,',',',',', \033[0m"],
"80": [
  "\033[38;5;226m _`/\`\"`\033[38;5;250m.-. \033[0m",
  "\033[38;5;226m ,\\_\033[38;5;250m( ). \033[0m",
  "\033[38;5;226m /\033[38;5;250m(__(_) \033[0m",
  "\033[38;5;111m ' ' ' ' \033[0m",
  "\033[38;5;111m ' ' ' ' \033[0m"],
"95": [
  "\033[38;5;240;1m .-. \033[0m",
  "\033[38;5;240;1m ( ). \033[0m",
  "\033[38;5;240;1m (__(_) \033[0m",
  "\033[38;5;21;1m
, '\033[38;5;228;5m⚡\033[38;5;21;25m', \033[38;5;228;5m⚡\033[38;5;21;25m, '
\033[0m",
  "\033[38;5;21;1m ,',','\033[38;5;228;5m⚡\033[38;5;21;25m', ' \033[0m"],
"97": [
  "\033[38;5;240;1m .-. \033[0m",
  "\033[38;5;240;1m ( ). \033[0m",
  "\033[38;5;240;1m (__(_) \033[0m",
  "\033[38;5;21;1m
, '\033[38;5;228;5m⚡\033[38;5;21;25m', \033[38;5;228;5m⚡\033[38;5;21;25m, '
\033[0m",
  "\033[38;5;21;1m ,',','\033[38;5;228;5m⚡\033[38;5;21;25m', ' \033[0m"],
}
```