****

**SISTEM MANAJEMEN SERVICE BENGKEL RESMI HONDA**

MUHAMMAD ALI PRATAMA PUTRA

5220411416

**DAFTAR ISI**

3

3

4

5

7

9

9

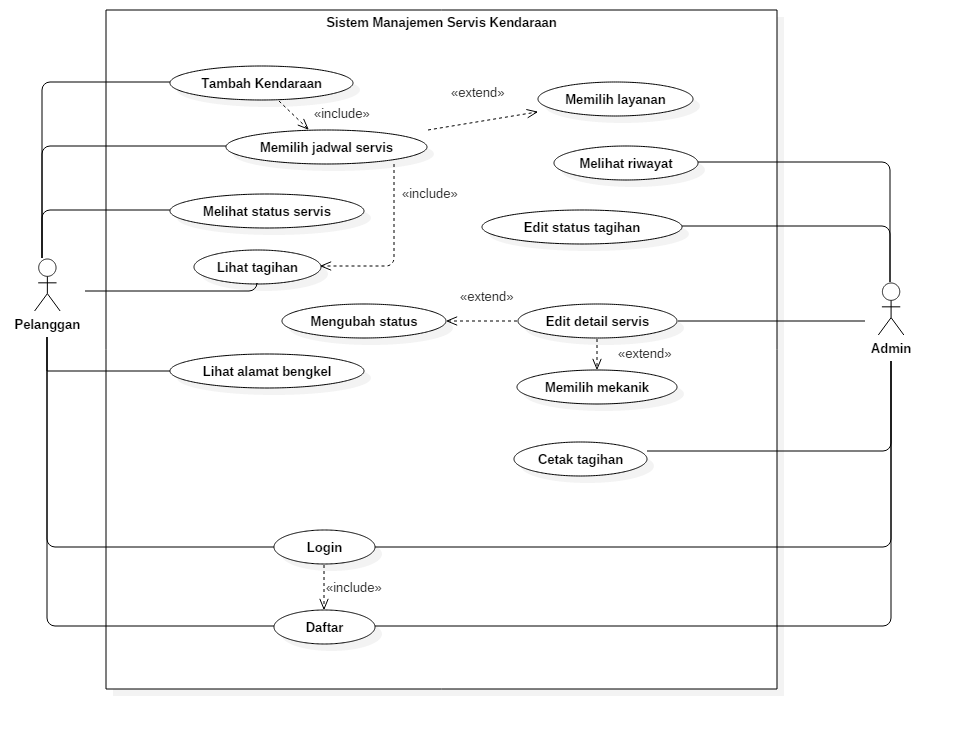
32

1. Latar Belakang
2. Detail Sistem
3. Library yang Digunakan
4. Database
5. Struktur File
6. Teknis Perancangan Aplikasi
7. Source Code
8. Output/Keluaran
9. **Latar Belakang**

Saya memilih tema ini karena menurut saya aplikasi ini akan sering digunakan terutama untuk bisnis bengkel baik resmi atau mandiri. Aplikasi ini dapat membantu sistem transaksi konsumen dan juga admin bengkel. Saya terinspirasi untuk memakai brand mobil Honda karena saya penggemar dari brand ini.

1. **Detail Sistem**

Saya telah membuat rancangan sistem ini menggunakan use case diagram, yang dimana diagram ini adalah standar dari UML (Unified Modelling Language) diagram.



Aplikasi ini terdiri dari 2 aktor, yaitu admin dan pelanggan. Untuk fitur pelanggan terdapat tambah data kendaraan, memilih jadwal servis, melihat status servis, melihat tagihan, dan melihat alamat bengkel. Dan fitur admin terdiri dari, melihat semua riwayat, mengubah status tagihan pelanggan, mengubah status servis pelanggan dan memilih mekanik, serta dapat mencetak tagihan dalam format pdf. Kedua aktor ini melewati tahap autentikasi (login) terlebih dahulu sebelum memasuki tampilan utama aplikasi. Jika pelanggan belum memiliki akun, pelanggan dapat registrasi, dan mengakses sebagai akses pelanggan.

1. **Library yang digunakan**

* Os (built-in)

Library ini digunakan untuk merapihkan screen.

Dokumentasi library ini dapat dilihat [disini](https://docs.python.org/3/library/os.html).

* Time (built-in)

Library ini digunakan untuk memberikan delay pada screen.

Dokumentasi library ini dapat dilihat [disini](https://docs.python.org/3/library/time.html).

* Datetime (built-in)

Library ini digunakan untuk mencetak waktu saat melakukan aktifitas pada riwayat transaksi dan membuat data tanggal untuk membuat jadwal servis.

Dokumentasi library ini dapat dilihat [disini](https://docs.python.org/3/library/datetime.html).

* Json (built-in)

Library ini digunakan untuk membaca file json pada file configuration aplikasi, dengan tujuan memudahkan pengembang untuk memodifikasi aplikasi.

Dokumentasi library ini dapat dilihat [disini](https://docs.python.org/3/library/json.html).

* Mysql.connector

Library ini digunakan untuk menghubungkan layanan mysql pada localhost dengan pemrograman python.

Dokumentasi library ini dapat dilihat [disini](https://dev.mysql.com/doc/connector-python/en/).

* Rich

Library ini digunakan untuk mengatur tampilan CLI (Command Line Interface) terlihat berwarna dan tidak membosankan.

Dokumentasi library ini dapat dilihat [disini](https://rich.readthedocs.io/en/stable/introduction.html).

* Questionary

Library ini digunakan untuk membuat menu interaktif, serta dapat membuat input jawaban user lebih mudah dipahami layaknya GUI (Graphical User Interface).

Dokumentasi library ini dapat dilihat [disini](https://questionary.readthedocs.io/en/stable/pages/installation.html).

* Bcrypt

Library ini digunakan untuk mengenkripsi password dengan metode hashing. Sehingga data sensitif pengguna lebih aman disimpan di database.

Dokumentasi library ini dapat dilihat [disini](https://pypi.org/project/bcrypt/).

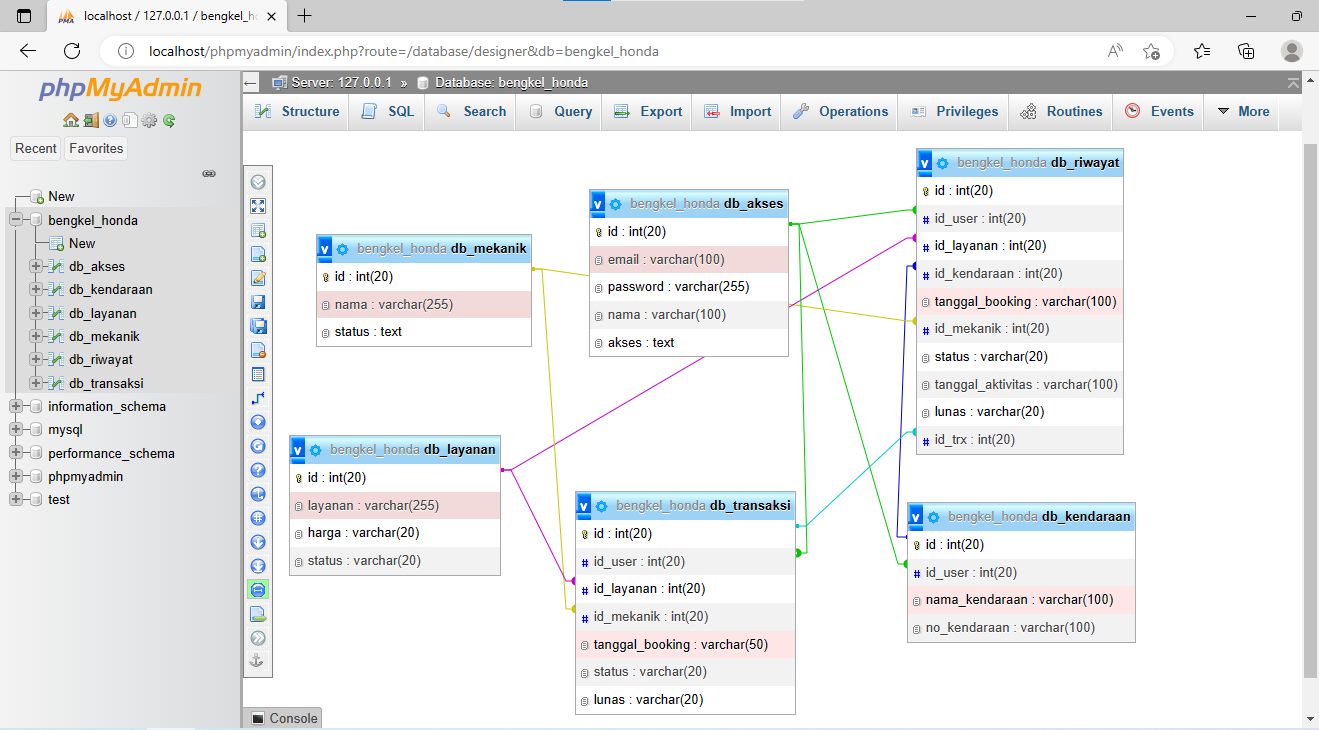
* FPDF

Library ini digunakan untuk membuat struk tagihan dalam format pdf yang bisa dicetak oleh printer.

Dokumentasi library ini dapat dilihat [disini](https://pyfpdf.readthedocs.io/en/latest/).

1. **Database**

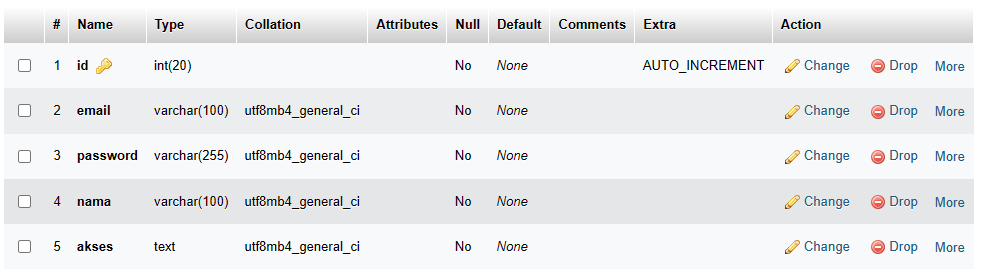
Setelah saya mengamati sistem dan beberapa referensi, saya membuat beberpa tabel yang berhubungan dengan aplikasi ini. Saya juga sudah membuat relasi tabel data yang merupakan hal penting untuk membuat rancangan sistem.



Berikut detail penjelasan mengenai tabel-tabel:

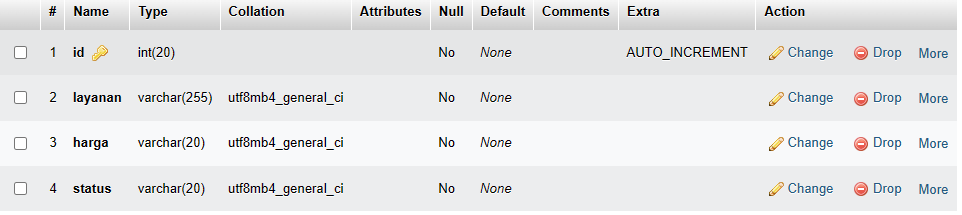
* db\_akses

Tabel ini bertujuan untuk menyimpan data akun pengguna, baik pelanggan maupun admin.



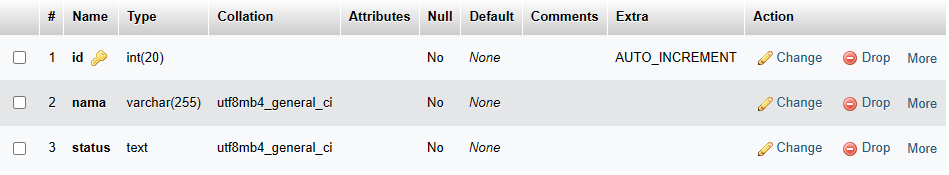
* db\_layanan

Tabel ini bertujuan untuk menyimpan data layanan, harga, serta status yang disediakan oleh bengkel.



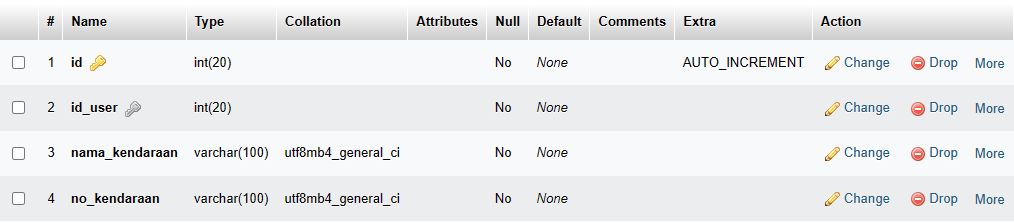
* db\_mekanik

Tabel ini bertujuan untuk menyimpan nama mekanik dan status yang bekerja di bengkel.



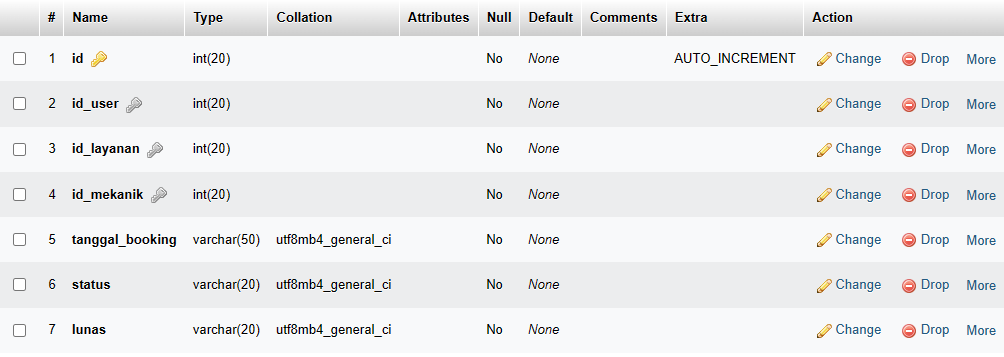
* db\_kendaraan

Tabel ini bertujuan untuk menyimpan data model mobil, dan nomor kendaraan dari pelanggan. Dengan syarat, 1 pelanggan hanya dapat menyimpan 1 data mobil saja.



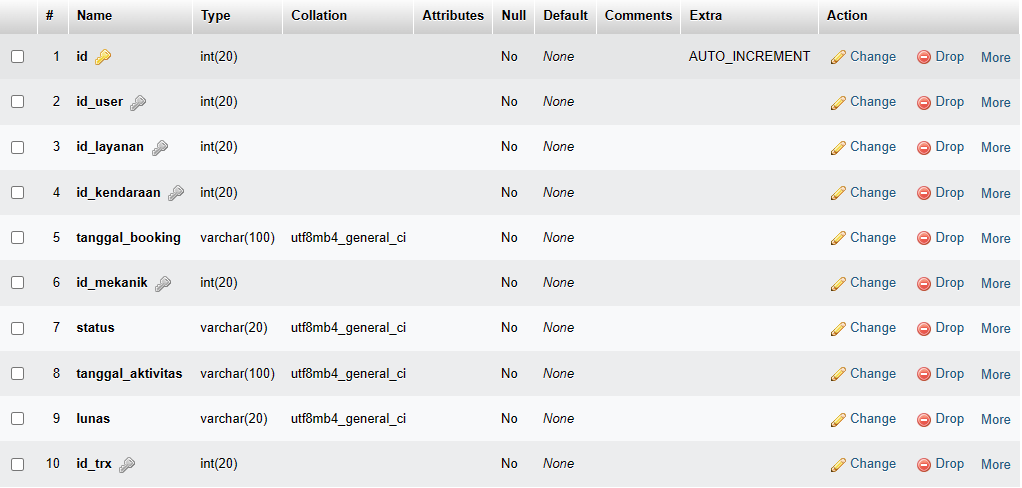
* db\_transaksi

Tabel ini bertujuan untuk menyimpan data transaksi pengguna.



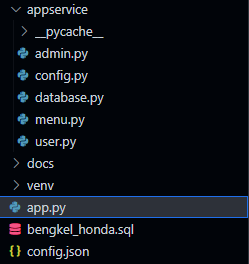
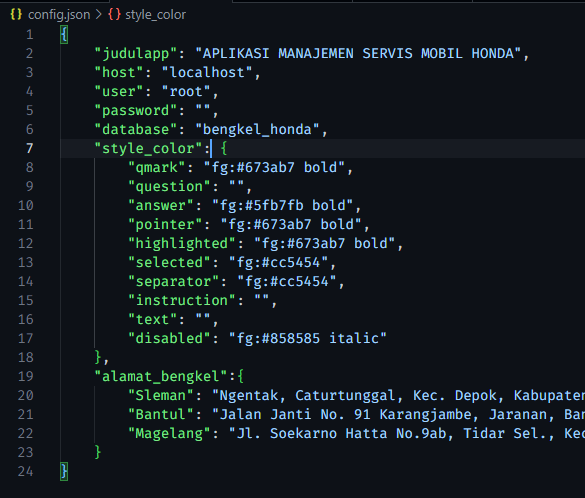
* db\_riwayat

Tabel ini bertujuan untuk menyimpan pertinggal dari data transaksi dan disediakan kolom yang lebih lengkap dibanding tabel db\_transaksi.



1. **Struktur File**

Struktur file projek yang rapih dapat membuat proses pengembangan menjadi lebih mudah, berikut hasil struktur yang saya buat.

*File config.json*

*Struktur file*

* config.json

File ini berfungsi sebagai file konfigurasi aplikasi ini, jadi pemilik komputer bisa mengatur judul aplikasi, konfigurasi localhost, dan tema warna tampilan.

* app.py

File ini merupakan file utama projek.

* bengkel\_honda.sql

File ini berfungsi sebagai database cadangan, ini dibutuhkan ketika menjalankan aplikasi pada komputer yang berbeda.

* appservice\config.py

File ini mengatur jalannya config.json ke fungsi-fungsi python dan menghubungkan ke sub program lain.

* appservice\database.py

File ini berisikan class session yang memiliki beberapa atribut mysql dan method yang menyambungkan ke layanan mysql database menggunakan paradigma OOP (Object Oriented Programming).

* appservice\menu.py

File ini berisikan fungsi-fungsi modular tampilan/front-end yang akan berhubungan dengan jalannya proses sistem/back-end.

* appservice\admin.py

File ini berisikan class dari actor admin. Memiliki atribut id, memiliki method yang berbeda dari pelanggan, dan berkitan dengan class session.

* appservice\user.py

File ini berisikan class dari actor user/pelanggan. Memiliki atribut id, memiliki method yang berbeda dari admin, dan berkitan dengan class session.

1. **Teknis Perancangan Aplikasi**

* Sebelum daya membuat aplikasi ini, saya membuat konsep dahulu dengan use case diagram, lalu saya membuat struktur database dan membuat relasi antar tabel data.
* Saya mencari beberapa referensi mengenai library built-in maupun library eksternal.
* Di projek ini saya menggunakan VE (Virtual Environment) yang mana dapat membantu kelancaran aplikasi ini dan bisa digunakan di PC manapun.
* Saya merancang aplikasi ini dengan python versi 3.11.1 dan software pengolah teks Visual Studio Code.
* File source projek dapat diunduh [disini](https://drive.google.com/drive/folders/1jIElbN1GVSEm-3W7vAL0j13BOJwoQVKI?usp=sharing).

1. **Source Code**

* **App.py (File Utama) [63 Baris]**

"""

+ ------------------------------------------------ +

|  Nama Lengkap    : Muhammad Ali Pratama Putra    |

|  NIM             : 5220411416                    |

|  Prodi           : Informatika                   |

+ ------------------------------------------------ +

"""

import time

from appservice import menu

from appservice import database

from appservice import user

from appservice import admin

session = database.Session()

session.connect()

def main\_menu():

    if session.jenis\_akses == 'admin':

        users = admin.Admin(session.nama\_akun, session.id\_akun)

    else:

        users = user.User(session.nama\_akun, session.id\_akun)

    menu.main\_menu\_akses(users.nama, session.jenis\_akses)

    users.menu(session)

    del users

while True:

    menu.menu\_01()

    pilihan\_01 = menu.pilihan\_01\_01()

    if pilihan\_01 == 0:

        menu.menu\_01()

        login = menu.menu\_02()

        data\_akun = session.get\_data\_where(str(login[0]).lower(),'db\_akses', 1)

        akses = session.auth(login[1], data\_akun, 2)

        menu.login\_load()

        menu.login\_alert(akses)

        time.sleep(2)

        if akses:

            session.jenis\_akses = data\_akun[-1]

            session.nama\_akun = data\_akun[-2]

            session.id\_akun = data\_akun[0]

            menu.menu\_akses(data\_akun[-1])

            main\_menu()

    elif pilihan\_01 == 1:

        menu.menu\_01()

        data = menu.menu\_03()

        if bool(str(data[1]).find('@') == -1 and str(data[1]).find('.') == -1) or data[2] != data[3]:

            menu.register\_alert(False)

        else:

            menu.register\_load()

            menu.register\_alert(True) if session.create\_data('db\_akses', [i for i in range(1, 5)], [str(data[1]).lower(), data[2], data[0], 'user'], True, [1]) == 1 else menu.register\_alert(False)

        time.sleep(2)

        session.jenis\_akses = 'user'

        session.nama\_akun = data[0]

        data\_akun = session.get\_data\_where(data[1],'db\_akses', 1)

        session.id\_akun = data\_akun[0]

        menu.menu\_akses('user')

        main\_menu()

    elif pilihan\_01 == 2:

        menu.menu\_quit()

        menu.clear\_with\_delay(1)

        session.quit()

        quit()

* **Menu.py [271 Baris]**

"""

+ ------------------------------------------------ +

|  Nama Lengkap    : Muhammad Ali Pratama Putra    |

|  NIM             : 5220411416                    |

|  Prodi           : Informatika                   |

+ ------------------------------------------------ +

"""

import os

import datetime

import time

import questionary

from questionary import Style

from rich.console import Console

from rich.progress import track

from rich.panel import Panel

from rich.table import Table

try:

    import config

except:

    from appservice import config

console = Console(width = 90)

tema = Style(config.getStyleColor())

def clear\_with\_delay(seconds):

    seconds = int(seconds)

    time.sleep(seconds)

    os.system('cls')

def menu\_01():

    os.system('cls')

    console.print('Selamat Datang', style='#af1d1e blink bold', justify='center')

    console.print(config.get\_string('judulapp'), justify='center')

    print('')

def menu\_02():

    email = questionary.text("Masukkan Email Anda", style=tema).ask()

    passwd = questionary.password("Masukkan Password Anda", style=tema).ask()

    print('')

    return [email, passwd]

def menu\_03():

    nama = questionary.text("Masukkan Nama Lengkap Anda", style=tema).ask()

    email = questionary.text("Masukkan Email", style=tema).ask()

    pass1 = questionary.password("Masukkan Password", style=tema).ask()

    pass2 = questionary.password("Masukkan Konfirmasi Password", style=tema).ask()

    print('')

    return [nama, email, pass1, pass2]

def main\_menu\_akses(nama: str, akses: str):

    os.system('cls')

    console.print(f'Selamat Datang {nama}', style='#af1d1e blink bold', justify='center')

    console.print(f'Anda masuk sebagai {akses}', justify='center')

    print('')

def main\_menu\_user():

    pilihan = ('🚗 Cek Kendaraan','📅 Buat Jadwal Servis', '🔍 Lihat Status Servis','🧾 Lihat Tagihan','📍 Lihat Alamat Bengkel Kami','❌ Keluar')

    choise = pilihan.index(questionary.select(

        'Ada yang bisa kami bantu? 👋',

        choices=pilihan,

        style=tema

    ).ask())

    return choise

def menu\_user\_buat\_kendaraan():

    os.system('cls')

    console.print('Sepertinya anda belum menambahkan data kendaraan', justify='center')

    print('')

    tipe = questionary.text("Masukkan Model Kendaraan Anda", style=tema).ask()

    plat = questionary.text("Masukkan Nomor Kendaraan Anda", style=tema).ask()

    return [tipe, plat]

def menu\_user\_00\_create():

    os.system('cls')

    tipe = questionary.text("Masukkan Model Kendaraan Anda", style=tema).ask()

    plat = questionary.text("Masukkan Nomor Kendaraan Anda", style=tema).ask()

    return [tipe, plat]

def menu\_user\_00\_check(model: str, plat: str):

    os.system('cls')

    pilihan = ('📝 Ubah', '⬅ Kembali')

    console.print(f'🚗 Model Kendaraan\t: {model}', style='white bold')

    console.print(f'📟 Nomor Kendaraan\t: {plat}', style='white bold')

    print('')

    return pilihan.index(questionary.select(

        'Apakah anda ingin mengubah data?',

        choices=pilihan,

        style=tema

    ).ask())

def menu\_user\_01\_sudah():

    os.system('cls')

    console.print('Anda sudah booking', style='#673ab7 blink bold')

    print('')

    questionary.select('',choices=['⬅ Kembali'],style=tema).ask()

def menu\_user\_01\_pilih\_layanan(layanan: list):

    os.system('cls')

    return [layanan.index(i) for i in questionary.checkbox(

        '🔧 Silahkan pilih layanan servis',

        choices=layanan,

        style=tema

    ).ask()]

def menu\_user\_01\_pilih\_hari():

    os.system('cls')

    pilih\_hari = ('Hari Ini', 'Besok', 'Kustom')

    pilihan = pilih\_hari.index(questionary.select(

        '📆 Silahkan pilih mode tanggal',

        choices=pilih\_hari,

        style=tema

    ).ask())

    hari\_ini = datetime.datetime.now()

    besok = hari\_ini + datetime.timedelta(1)

    if pilihan == 0:

        return [hari\_ini.day, hari\_ini.month, hari\_ini.year]

    elif pilihan == 1:

        return [besok.day, besok.month, besok.year]

    elif pilihan == 2:

        return str(questionary.text("Masukkan Tanggal dengan format HH-BB-TTTT", style=tema).ask()).split('-')

def menu\_user\_02(layanan: list, tanggal\_booking: list, status: list):

    os.system('cls')

    [console.print(f"[white]{layanan[i]}{'      ' if len(layanan[i]) < 15 else ''}\t[#673ab7]{tanggal\_booking[i]}\t[#af1d1e][b]{status[i]}[/b]") for i in range(len(layanan))]

    print('')

    questionary.select('',choices=['⬅ Kembali'],style=tema).ask()

def menu\_user\_03(layanan: list, harga: list):

    os.system('cls')

    console.print('Rincian Tagihan Anda', style='#af1d1e blink bold')

    print('')

    [console.print(f"[white]{layanan[i]}{'      ' if len(layanan[i]) < 15 else ''}\t[#673ab7]Rp{harga[i]}") for i in range(len(layanan))]

    print('')

    console.print(f'Total Pembayaran: Rp{sum([int(i) for i in harga])}', style='#af1d1e blink bold')

    print('')

    questionary.select('',choices=['⬅ Kembali'],style=tema).ask()

def menu\_user\_04():

    os.system('cls')

    console.print('Alamat Bengkel Resmi Honda', style='#673ab7 blink bold')

    data = dict(config.get\_string('alamat\_bengkel')).items()

    [console.print(f"[#af1d1e][b]{i[0]}[/b]\n[white]{i[1]}") for i in data]

    print('')

    questionary.select('',choices=['⬅ Kembali'],style=tema).ask()

def main\_menu\_admin():

    pilihan = ('🖍 Edit Detail Servis','📅 Edit Status Tagihan', '🖨 Cetak Tagihan','🧾 Lihat Riwayat Transaksi','❌ Keluar')

    choise = pilihan.index(questionary.select(

        'Silahkan pilih fitur admin 💻',

        choices=pilihan,

        style=tema

    ).ask())

    return choise

def menu\_admin\_00\_pilih\_trx(layanan: list, status: list, id\_transaksi: list):

    pilihan = ['⬅ Kembali']

    for i in range(len(layanan)):

        pilihan.append(f'{layanan[i]} | {status[i]}')

    choise = pilihan.index(questionary.select(

        'Silahkan pilih daftar booking 🔧',

        choices=pilihan,

        style=tema

    ).ask())

    if choise > 0:

        return [id\_transaksi[choise - 1], status[choise - 1]]

def menu\_admin\_00\_pilih\_mekanik(mekanik: list, id\_mekanik: list):

    pilihan = ['⬅ Kembali']

    for i in range(len(mekanik)):

        pilihan.append(mekanik[i])

    choise = pilihan.index(questionary.select(

        'Silahkan pilih mekanik 👷‍♂️',

        choices=pilihan,

        style=tema

    ).ask())

    if choise > 0:

        return id\_mekanik[choise - 1]

def menu\_admin\_00\_pilih\_status():

    pilihan = ('⬅ Kembali', '⏱ Proses', '✔ Selesai')

    choise = questionary.select(

        'Silahkan pilih status 💻',

        choices=pilihan,

        style=tema

    ).ask()

    if choise == pilihan[1]:

        return 'proses'

    elif choise == pilihan[2]:

        return 'selesai'

def menu\_admin\_01(user: list):

    os.system('cls')

    return user.index(questionary.select(

        'Silahkan pilih user 🧍‍♂️',

        choices=user,

        style=tema

    ).ask())

def menu\_admin\_01\_ubah(layanan: list, harga: list, lunas: list, id\_transaksi: list):

    pilihan = ['⬅ Kembali']

    for i in range(len(layanan)):

        pilihan.append(f'{layanan[i]} | {harga[i]} | {lunas[i]}')

    choise = pilihan.index(questionary.select(

        'Silahkan ubah status pelunasan 💵',

        choices=pilihan,

        style=tema

    ).ask())

    if choise > 0:

        return [id\_transaksi[choise - 1], lunas[choise - 1]]

def menu\_admin\_02(user: list):

    os.system('cls')

    return user.index(questionary.select(

        'Silahkan pilih user 🧍‍♂️',

        choices=user,

        style=tema

    ).ask())

def menu\_admin\_04(data: list):

    os.system('cls')

    table = Table(title="Riwayat Seluruh Transaksi")

    table.add\_column("Email", style="cyan", no\_wrap=True)

    table.add\_column("Nama", style="magenta", no\_wrap=True)

    table.add\_column("Layanan", style="green", no\_wrap=True)

    table.add\_column("Mobil", style="cyan", no\_wrap=True)

    table.add\_column("Tanggal Booking", style="magenta", no\_wrap=True)

    table.add\_column("Mekanik", style="green", no\_wrap=True)

    table.add\_column("Status", style="cyan", no\_wrap=True)

    table.add\_column("Lunas", style="magenta", no\_wrap=True)

    for i in data:

        i = str(i).split(' | ')

        table.add\_row(i[0], i[1], i[2], i[3], i[4], i[5], i[6], i[7])

    console.print(table, justify="center")

    print('')

    questionary.select('',choices=['⬅ Kembali'],style=tema).ask()

def pilihan\_01\_01():

    pilihan = ('🚗 Masuk','📖 Registrasi','❌ Keluar')

    choise = pilihan.index(questionary.select(

        'Ada yang bisa kami bantu? 👋',

        choices=pilihan,

        style=tema

    ).ask())

    return choise

def error\_alert(error\_msg):

    os.system('cls')

    error = Table.grid()

    error.add\_row(

    Panel.fit(

        f"[white]{error\_msg}",

        title="Error!",

        border\_style="#af1d1e",

        padding=(2, 2)

    ))

    console.print(error)

def bool\_alert(bool: bool):

    if bool:

        console.print('Berhasil', style='#673ab7 blink bold')

    else:

        console.print('Gagal', style='#af1d1e blink bold')

def login\_alert(bool: bool):

    if bool:

        console.print('Berhasil', style='#673ab7 blink bold')

    else:

        console.print('Gagal', style='#af1d1e blink bold')

def register\_alert(bool: bool):

    if bool:

        console.print('Berhasil Membuat Akun', style='#673ab7 blink bold')

    else:

        console.print('Gagal Membuat Akun', style='#af1d1e blink bold')

def login\_load():

    for i in track(range(20), description="Autentikasi Akun..."):

        time.sleep(0.075)

def register\_load():

    for i in track(range(20), description="Membuat akun anda..."):

        time.sleep(0.1)

def savedata\_load():

    for i in track(range(20), description="Menyimpan data..."):

        time.sleep(0.05)

def pdf\_alert():

    console.print('PDF sudah disimpan, silahkan print...', style='#673ab7 blink bold')

    print('')

    questionary.select('',choices=['⬅ Kembali'],style=tema).ask()

def menu\_quit():

    os.system('cls')

    console.print('Terima Kasih', style='#af1d1e blink bold', justify='center')

def menu\_akses(akses):

    os.system('cls')

    console.print(f'Mengarahkan anda ke menu {akses}', style='white blink bold', justify='center')

    time.sleep(2)

* **Config.py [19 Baris]**

"""

+ ------------------------------------------------ +

|  Nama Lengkap    : Muhammad Ali Pratama Putra    |

|  NIM             : 5220411416                    |

|  Prodi           : Informatika                   |

+ ------------------------------------------------ +

"""

import json

import os

dir = os.getcwd().replace('\\', '/')

with open(f'{dir}/config.json') as config\_file:

    config = json.load(config\_file)

def get\_string(key):

    return config.get(key)

def getStyleColor():

    return list(config['style\_color'].items())

* **Database.py [210 Baris]**

"""

+ ------------------------------------------------ +

|  Nama Lengkap    : Muhammad Ali Pratama Putra    |

|  NIM             : 5220411416                    |

|  Prodi           : Informatika                   |

+ ------------------------------------------------ +

"""

import mysql.connector as mysql

import bcrypt

try:

import config

except:

from appservice import config

try:

import menu

except:

from appservice import menu

class Session():

global mysql

def \_\_init\_\_(self):

self.host = config.config['host']

self.user = config.config['user']

self.passwd = config.config['password']

self.db = config.config['database']

self.jenis\_akses = 'user'

self.nama\_akun = ''

self.id\_akun = 0

def connect(self):

try:

self.conn = mysql.connect(

host=self.host,

user=self.user,

passwd=self.passwd,

database=self.db

)

except:

menu.error\_alert('Tidak Terkoneksi Database')

quit()

else:

self.cursor = self.conn.cursor(buffered=True)

def quit(self):

self.conn.disconnect()

def get\_column(self, tabel: str):

try:

self.cursor.execute(f"""

SHOW COLUMNS FROM {tabel}

""")

except:

menu.error\_alert('Ada kesalahan read database')

return ''

else:

columns = self.cursor.fetchall()

return tuple([column[0] for column in columns])

def get\_data(self, tabel: str, index\_kolom: list = [], isSingle: bool = False):

tabel, isSingle, index\_kolom = str(tabel), bool(isSingle), tuple(index\_kolom)

kolom\_data = [self.get\_column(tabel)[i] for i in index\_kolom]

if len(index\_kolom) < 1:

sql = f"""

SELECT \* FROM {tabel}

"""

else:

sql = f"""

SELECT {str(kolom\_data)[1:-1].replace("'", '')} FROM {tabel}

"""

try:

self.cursor.execute(sql)

except:

menu.error\_alert('Ada kesalahan read database')

return ''

else:

if isSingle == True:

return self.cursor.fetchone()

else:

return self.cursor.fetchall()

def create\_data(self, tabel: str, index\_kolom: list = [], values: list = [], isEncrypt: bool = False, index\_enkripsi: list = []):

kolom = [self.get\_column(tabel)[i] for i in index\_kolom]

values = [str(i) for i in values]

if isEncrypt == True:

for i in index\_enkripsi:

values[i] = str(bcrypt.hashpw(values[i].encode('utf-8'), bcrypt.gensalt()))[2:-1]

sql = f"""INSERT INTO {tabel} ({str(kolom)[1:-1].replace("'", '')}) VALUES ({str(values)[1:-1]})"""

try:

self.cursor.execute(sql)

except:

menu.error\_alert('Ada kesalahan write database')

return 0

else:

self.conn.commit()

return 1

def delete\_data(self, tabel: str, index\_kolom: list = [], values: list = [], logical: str = 'AND'):

kolom = [self.get\_column(tabel)[i] for i in index\_kolom]

values = [str(i) for i in values]

if len(index\_kolom) > 1:

sql = f"""DELETE FROM {tabel} WHERE {str([str(f'{kolom[i]}="{values[i]}"') for i, v in enumerate(index\_kolom)])[1:-1].replace(',', f' {logical}').replace("'", '')}"""

else:

sql = f"""DELETE FROM {tabel} WHERE {kolom[index\_kolom[0]]}='{values[0]}'"""

try:

self.cursor.execute(sql)

except:

menu.error\_alert('Ada kesalahan delete database')

return 0

else:

self.conn.commit()

return 1

def edit\_data(self, tabel: str, index\_kolom: list = [], values: list = [], index\_anchor: int = 0, value\_anchor: str = '', isEncrypt: bool = False, index\_enkripsi: list = []):

kolom = [self.get\_column(tabel)[i] for i in index\_kolom]

values = [str(i) for i in values]

if isEncrypt == True:

for i in index\_enkripsi:

values[i] = str(bcrypt.hashpw(values[i].encode('utf-8'), bcrypt.gensalt()))[2:-1]

set\_statement = [f'{kolom[i]}="{values[i]}"' for i in range(len(values))]

sql = f"""UPDATE {tabel} SET {str(set\_statement)[1:-1].replace("'", '')} WHERE {self.get\_column(tabel)[index\_anchor]}=\"{value\_anchor}\""""

try:

self.cursor.execute(sql)

except:

menu.error\_alert('Ada kesalahan write database')

return 0

else:

self.conn.commit()

return 1

def get\_data\_where(self, input\_string: str, tabel: str, index\_kolom: int):

kolom = self.get\_column(tabel)[index\_kolom]

sql = f"""SELECT \* FROM {tabel} WHERE {kolom}=\"{input\_string}\""""

try:

self.cursor.execute(sql)

except:

menu.error\_alert('Ada kesalahan read database')

return ''

else:

return self.cursor.fetchone()

def get\_data\_wheres(self, tabel: str, index\_anchor: list, anchor\_string: list, index\_target: list, isSingle: bool = False):

kolom = [self.get\_column(tabel)[i] for i in index\_anchor]

where\_statement = [f'{kolom[i]}="{anchor\_string[i]}"' for i in range(len(index\_anchor))]

if len(index\_target) > 0:

select\_statement = [self.get\_column(tabel)[i] for i in index\_target]

sql = f"""SELECT {str(select\_statement)[1:-1].replace("'", '')} FROM {tabel} WHERE {str(where\_statement)[1:-1].replace(',', 'AND ').replace("'", '')}"""

else:

sql = f"""SELECT \* FROM {tabel.replace("'", '')} WHERE {str(where\_statement)[1:-1].replace(',', 'AND ').replace("'", '')}"""

try:

self.cursor.execute(sql)

except:

menu.error\_alert('Ada kesalahan read database')

return ''

else:

if isSingle:

return self.cursor.fetchone()

else:

return self.cursor.fetchall()

def auth(self, input\_string: str, data: str, index\_target: int):

try:

input\_string = bytes(input\_string, 'utf-8')

hash\_value = bytes(data[index\_target], 'utf-8')

except:

return False

else:

return bcrypt.checkpw(input\_string, hash\_value)

def get\_data\_sort(self, tabel: str, index\_kolom: list = [], index\_anchor: int = 0, asc: bool = True, isSingle: bool = False):

tabel, isSingle, index\_kolom = str(tabel), bool(isSingle), tuple(index\_kolom)

kolom\_data = self.get\_column(tabel)

if len(index\_kolom) < 1:

sql = f"""

SELECT \* FROM {tabel} ORDER BY {kolom\_data[index\_anchor]} {'DESC' if asc == False else 'ASC'}

"""

else:

sql = f"""

SELECT {str(kolom\_data)[1:-1].replace("'", '')} FROM {tabel} ORDER BY {kolom\_data[index\_anchor]} {'DESC' if asc == False else ''}

"""

try:

self.cursor.execute(sql)

except:

menu.error\_alert('Ada kesalahan read database')

return ''

else:

if isSingle == True:

return self.cursor.fetchone()

else:

return self.cursor.fetchall()

def create\_dataa(self, tabel: str, list\_kolom: list = [], values: list = [], isEncrypt: bool = False, index\_enkripsi: list = []):

kolom = list\_kolom

values = [str(i) for i in values]

if isEncrypt == True:

for i in index\_enkripsi:

values[i] = str(bcrypt.hashpw(values[i].encode('utf-8'), bcrypt.gensalt()))[2:-1]

sql = f"""INSERT INTO {tabel} ({str(kolom)[1:-1].replace("'", '')}) VALUES ({str(values)[1:-1]})"""

try:

self.cursor.execute(sql)

except:

menu.error\_alert('Ada kesalahan write database')

return 0

else:

self.conn.commit()

return 1

def get\_data\_sortt(self, tabel: str, list\_kolom: list = [], index\_anchor: int = 0, asc: bool = True, isSingle: bool = False):

tabel, isSingle = str(tabel), bool(isSingle)

kolom\_data = list\_kolom

sql = f"""

SELECT \* FROM {tabel} ORDER BY {kolom\_data[index\_anchor]} {'DESC' if asc == False else 'ASC'}

"""

try:

self.cursor.execute(sql)

except:

menu.error\_alert('Ada kesalahan read database')

return ''

else:

if isSingle == True:

return self.cursor.fetchone()

else:

return self.cursor.fetchall()

* **Admin.py [156 Baris]**
* """

+ ------------------------------------------------ +

|  Nama Lengkap    : Muhammad Ali Pratama Putra    |

|  NIM             : 5220411416                    |

|  Prodi           : Informatika                   |

+ ------------------------------------------------ +

"""

import time

import datetime

from fpdf import FPDF

try:

    import menu

except:

    from appservice import menu

class Admin():

    def \_\_init\_\_(self, nama, id\_akun):

        self.nama = nama

        self.id\_akun = id\_akun

    def menu(self, session):

        while True:

            menu.main\_menu\_akses(self.nama, 'admin')

            pilihan = menu.main\_menu\_admin()

            if pilihan == 4:

                session.nama\_akun = ''

                session.id\_akun = 0

                break

            elif pilihan == 3:

                akses = session.get\_data('db\_akses', [0,1,3])

                d\_akses = [i[0] for i in akses]

                layanan = session.get\_data('db\_layanan', [0,1])

                d\_layanan = [i[0] for i in layanan]

                kendaraan = session.get\_data('db\_kendaraan', [0,2])

                d\_kendaraan = [i[0] for i in kendaraan]

                mekanik = session.get\_data('db\_mekanik', [0,1])

                d\_mekanik = [i[0] for i in mekanik]

                tabel = []

                for i in session.get\_data('db\_riwayat', []):

                    tabel.append(f'{akses[d\_akses.index(i[1])][1]} | {akses[d\_akses.index(i[1])][2]} | {layanan[d\_layanan.index(i[2])][1]} | {kendaraan[d\_kendaraan.index(i[3])][1]} | {i[4]} | {mekanik[d\_mekanik.index(i[5])][1]} | {i[6]} | {i[8]}')

                menu.menu\_admin\_04(tabel)

            elif pilihan == 2:

                m\_data = session.get\_data\_wheres('db\_transaksi', [6], ['belum'], [1])

                m\_id\_akun = list(set([i[0] for i in m\_data]))

                m\_data\_akun = session.get\_data('db\_akses', [])

                m\_d\_akun = [i[0] for i in m\_data\_akun]

                m\_n\_akun = [i[1] for i in m\_data\_akun]

                m\_nn\_akun = [i[3] for i in m\_data\_akun]

                m\_nama\_akun = [f'{m\_n\_akun[m\_d\_akun.index(m\_id\_akun[i])]} | {m\_nn\_akun[m\_d\_akun.index(m\_id\_akun[i])]}' for i in range(len(m\_id\_akun))]

                id\_akun\_pilih = m\_d\_akun[menu.menu\_admin\_02(m\_nama\_akun)]

                # print(f'{id\_akun\_pilih}')

                # import os

                # os.system('pause')

                data\_h = session.get\_data('db\_layanan', [])

                id\_layanan = [i[0] for i in data\_h]

                d\_layanan = [i[1] for i in data\_h]

                d\_harga = [i[2] for i in data\_h]

                layanan = []

                harga = []

                data = session.get\_data\_wheres('db\_transaksi', [1, 6], [id\_akun\_pilih, 'belum'], [])

                for i in data:

                    layanan.append(d\_layanan[id\_layanan.index(i[2])])

                    harga.append(d\_harga[id\_layanan.index(i[2])])

                tghn = FPDF()

                tghn.add\_page()

                tghn.set\_font('Arial', 'B', 16)

                tghn.ln(20)

                tghn.cell(40, 10, 'Tagihan Servis Mobil')

                tghn.ln(7)

                tghn.set\_font('Arial', 'I', 13)

                tghn.cell(40, 10, 'Servis & Dealer Resmi Honda Yogyakarta')

                tghn.ln(20)

                for i in range(len(layanan)):

                    tghn.set\_font('Arial', '', 11)

                    tghn.cell(40, 10, f"{layanan[i]}{'      ' if len(layanan[i]) < 15 else ''}\tRp{harga[i]}")

                    tghn.ln(7)

                tghn.ln(15)

                tghn.set\_font('Arial', 'B', 11)

                tghn.cell(40, 10, f"Total Pembayaran: Rp{sum([int(i) for i in harga])}")

                tghn.output(f'tagihan\_{id\_akun\_pilih}.pdf', 'F')

                menu.savedata\_load()

                menu.bool\_alert(True)

                menu.pdf\_alert()

            elif pilihan == 1:

                m\_data = session.get\_data('db\_transaksi', [1])

                m\_id\_akun = list(set([i[0] for i in m\_data]))

                m\_data\_akun = session.get\_data('db\_akses', [])

                m\_d\_akun = [i[0] for i in m\_data\_akun]

                m\_n\_akun = [i[1] for i in m\_data\_akun]

                m\_nn\_akun = [i[3] for i in m\_data\_akun]

                m\_nama\_akun = [f'{m\_n\_akun[m\_d\_akun.index(m\_id\_akun[i])]} | {m\_nn\_akun[m\_d\_akun.index(m\_id\_akun[i])]}' for i in range(len(m\_id\_akun))]

                id\_akun\_pilih = m\_d\_akun[menu.menu\_admin\_01(m\_nama\_akun)]

                data\_h = session.get\_data('db\_layanan', [])

                id\_layanan = [i[0] for i in data\_h]

                d\_layanan = [i[1] for i in data\_h]

                d\_harga = [i[2] for i in data\_h]

                layanan = []

                harga = []

                lunas = []

                id\_transaksi = []

                data = session.get\_data\_wheres('db\_transaksi', [1], [id\_akun\_pilih], [])

                for i in data:

                    layanan.append(d\_layanan[id\_layanan.index(i[2])])

                    harga.append(d\_harga[id\_layanan.index(i[2])])

                    lunas.append(i[6])

                    id\_transaksi.append(i[0])

                pilih\_trx = menu.menu\_admin\_01\_ubah(layanan, harga, lunas, id\_transaksi)

                if pilih\_trx != None:

                    data\_riwayat = session.get\_data\_wheres('db\_riwayat', [9], [pilih\_trx[0]], [0])[0][0]

                    if pilih\_trx[1] == 'belum':

                        session.edit\_data('db\_transaksi', [6], ['sudah'], 0, pilih\_trx[0])

                        session.edit\_data('db\_riwayat', [7, 8], [datetime.datetime.now(), 'sudah'], 0, data\_riwayat)

                    else:

                        session.edit\_data('db\_transaksi', [6], ['belum'], 0, pilih\_trx[0])

                        session.edit\_data('db\_riwayat', [7, 8], [datetime.datetime.now(), 'belum'], 0, data\_riwayat)

                    menu.savedata\_load()

                    menu.bool\_alert(True)

                    time.sleep(2)

            elif pilihan == 0:

                m\_data = session.get\_data('db\_transaksi', [1])

                m\_id\_akun = list(set([i[0] for i in m\_data]))

                m\_data\_akun = session.get\_data('db\_akses', [])

                m\_d\_akun = [i[0] for i in m\_data\_akun]

                m\_n\_akun = [i[1] for i in m\_data\_akun]

                m\_nn\_akun = [i[3] for i in m\_data\_akun]

                m\_nama\_akun = [f'{m\_n\_akun[m\_d\_akun.index(m\_id\_akun[i])]} | {m\_nn\_akun[m\_d\_akun.index(m\_id\_akun[i])]}' for i in range(len(m\_id\_akun))]

                id\_akun\_pilih = m\_d\_akun[menu.menu\_admin\_01(m\_nama\_akun)]

                data\_h = session.get\_data('db\_layanan', [])

                id\_layanan = [i[0] for i in data\_h]

                d\_layanan = [i[1] for i in data\_h]

                layanan = []

                status = []

                id\_transaksi = []

                data = session.get\_data\_wheres('db\_transaksi', [1], [id\_akun\_pilih], [])

                for i in data:

                    layanan.append(d\_layanan[id\_layanan.index(i[2])])

                    status.append(i[5])

                    id\_transaksi.append(i[0])

                pilih\_trx = menu.menu\_admin\_00\_pilih\_trx(layanan, status, id\_transaksi)

                if pilih\_trx != None:

                    data\_riwayat = session.get\_data\_wheres('db\_riwayat', [9], [pilih\_trx[0]], [0])[0][0]

                    if pilih\_trx[1] == 'baru':

                        data\_mekanik = session.get\_data\_wheres('db\_mekanik', [2], ['ready'], [])

                        id\_mekanik = [i[0] for i in data\_mekanik]

                        nama\_mekanik = [i[1] for i in data\_mekanik]

                        id\_mekanik.remove(1)

                        nama\_mekanik.remove('null')

                        pilih\_mekanik = menu.menu\_admin\_00\_pilih\_mekanik(nama\_mekanik, id\_mekanik)

                        session.edit\_data('db\_transaksi', [3, 5], [pilih\_mekanik, 'proses'], 0, pilih\_trx[0])

                        session.edit\_data('db\_riwayat', [5, 6, 7], [pilih\_mekanik, 'proses', datetime.datetime.now()], 0, data\_riwayat)

                    else:

                        pilih\_status = menu.menu\_admin\_00\_pilih\_status()

                        session.edit\_data('db\_transaksi', [5], [pilih\_status], 0, pilih\_trx[0])

                        session.edit\_data('db\_riwayat', [6, 7], [pilih\_status, datetime.datetime.now()], 0, data\_riwayat)

                    menu.savedata\_load()

                    menu.bool\_alert(True)

                    time.sleep(2)

* **User.py [120 Baris]**

"""

+ ------------------------------------------------ +

|  Nama Lengkap    : Muhammad Ali Pratama Putra    |

|  NIM             : 5220411416                    |

|  Prodi           : Informatika                   |

+ ------------------------------------------------ +

"""

import time

import datetime

import mysql.connector as mysql

try:

    import menu

except:

    from appservice import menu

class User():

    def \_\_init\_\_(self, nama, id\_akun):

        self.nama = nama

        self.id\_akun = id\_akun

        self.status\_cek\_kendaraan = False

        self.id\_kendaraan = 0

        self.model\_kendaraan = ''

        self.nomor\_kendaraan = ''

        self.status\_booking = False

        self.id\_booking = []

    def cek\_kendaraan(self, session):

        data = session.get\_data\_where(self.id\_akun, 'db\_kendaraan', 1)

        if data == None:

            pengisian = menu.menu\_user\_buat\_kendaraan()

            menu.savedata\_load()

            session.create\_data('db\_kendaraan', [i for i in range(1, 4)], [self.id\_akun, pengisian[0], pengisian[1]])

            menu.bool\_alert(True)

            time.sleep(2)

        else:

            self.status\_cek\_kendaraan = True

            self.id\_kendaraan = int(data[0])

            self.model\_kendaraan = data[2]

            self.nomor\_kendaraan = data[3]

    def cek\_booking(self, session):

        if self.status\_booking == False:

            data\_sudah\_booking = session.get\_data\_wheres('db\_transaksi', [1], [self.id\_akun], [0])

            if len(data\_sudah\_booking) > 0:

                self.status\_booking = True

                for i in data\_sudah\_booking:

                    self.id\_booking.append(i[0])

    def tambah\_data\_riwayat(self, session, id\_layanan, tanggal, id\_trx):

        # session.create\_data('db\_riwayat', [i for i in range(1, 10)], [self.id\_akun, id\_layanan, self.id\_kendaraan, tanggal, '1', 'baru', datetime.datetime.now(), 'belum', id\_trx])

        session.cursor.execute("""

                               INSERT INTO db\_riwayat (id\_user, id\_layanan, id\_kendaraan, tanggal\_booking, id\_mekanik, status, tanggal\_aktivitas, lunas, id\_trx) VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s)

                               """, (self.id\_akun, id\_layanan, self.id\_kendaraan, tanggal, '1', 'baru', datetime.datetime.now(), 'belum', id\_trx))

        session.conn.commit()

    def menu(self, session):

        while self.status\_cek\_kendaraan == False:

            self.cek\_kendaraan(session)

        while True:

            menu.main\_menu\_akses(self.nama, 'user')

            pilihan = menu.main\_menu\_user()

            if pilihan == 5:

                session.nama\_akun = ''

                session.id\_akun = 0

                break

            elif pilihan == 4:

                menu.menu\_user\_04()

            elif pilihan == 0:

                if menu.menu\_user\_00\_check(self.model\_kendaraan, self.nomor\_kendaraan) == 0:

                    data = menu.menu\_user\_00\_create()

                    session.edit\_data('db\_kendaraan', [2, 3], [data[0], data[1]], 0, self.id\_kendaraan)

                    self.model\_kendaraan = data[0]

                    self.nomor\_kendaraan = data[1]

                    menu.savedata\_load()

                    menu.bool\_alert(True)

                    time.sleep(2)

            elif pilihan == 1:

                self.cek\_booking(session)

                if self.status\_booking == True:

                    menu.menu\_user\_01\_sudah()

                else:

                    data = session.get\_data('db\_layanan', [])

                    layanan = [i[1] for i in data]

                    layanan.insert(0,'Kembali')

                    index\_layanan = menu.menu\_user\_01\_pilih\_layanan(layanan)

                    if len(index\_layanan) > 0:

                        tanggal = str(menu.menu\_user\_01\_pilih\_hari())[1:-1].replace(', ', '-')

                        kolom\_trx = session.get\_column('db\_transaksi')

                        kolom\_riw = session.get\_column('db\_riwayat')

                        for il in index\_layanan:

                            il = int(il)

                            session.create\_dataa('db\_transaksi', kolom\_trx[1:], [self.id\_akun, data[il][0], '1', tanggal, 'baru', 'belum'])

                            id\_trx = session.get\_data\_sortt('db\_transaksi', kolom\_trx, 0, False, True)

                            session.create\_dataa('db\_riwayat', kolom\_riw[1:], [self.id\_akun, data[il][0], self.id\_kendaraan, tanggal, '1', 'baru', datetime.datetime.now(), 'belum', id\_trx[0]])

                        menu.savedata\_load()

                        menu.bool\_alert(True)

                        time.sleep(2)

            elif pilihan == 2:

                self.cek\_booking(session)

                data\_l = session.get\_data('db\_layanan', [])

                id\_layanan = [i[0] for i in data\_l]

                d\_layanan = [i[1] for i in data\_l]

                layanan = []

                tanggal\_booking = []

                status = []

                for i in self.id\_booking:

                    data = session.get\_data\_where(i, 'db\_transaksi', 0)

                    layanan.append(d\_layanan[id\_layanan.index(data[2])])

                    tanggal\_booking.append(data[4])

                    status.append(data[5])

                menu.menu\_user\_02(layanan, tanggal\_booking, status)

            elif pilihan == 3:

                self.cek\_booking(session)

                data\_h = session.get\_data('db\_layanan', [])

                id\_layanan = [i[0] for i in data\_h]

                d\_layanan = [i[1] for i in data\_h]

                d\_harga = [i[2] for i in data\_h]

                layanan = []

                harga = []

                data = session.get\_data\_wheres('db\_transaksi', [1, 6], [self.id\_akun, 'belum'], [])

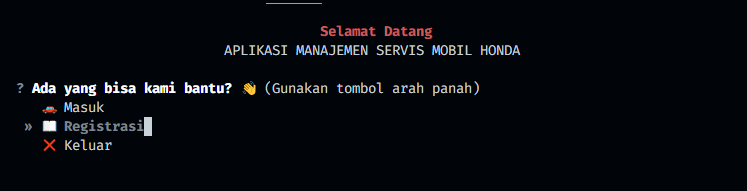
                for i in data:

                    layanan.append(d\_layanan[id\_layanan.index(i[2])])

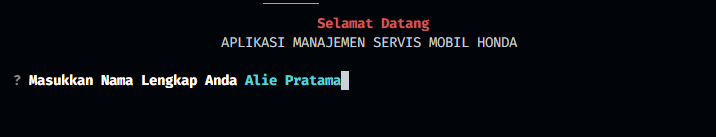
                    harga.append(d\_harga[id\_layanan.index(i[2])])

                menu.menu\_user\_03(layanan, harga)

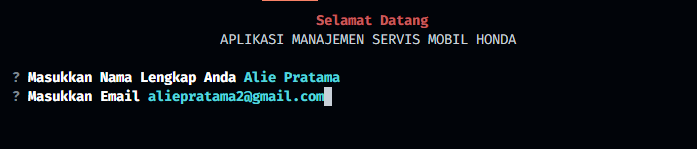
1. **Output/Keluaran**
2. **Login & Registrasi Pelanggan**
3. **Registrasi**
4. Menu utama

****

1. Mengisi nama lengkap

****

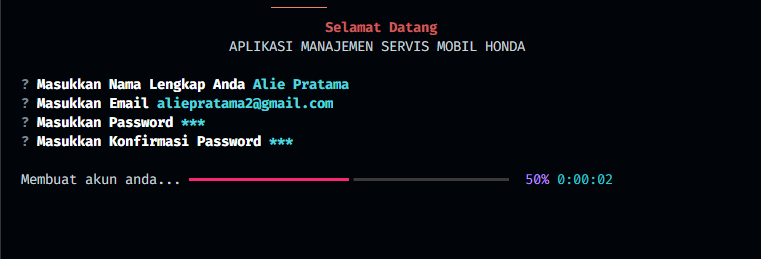
1. Mengisi data email

****

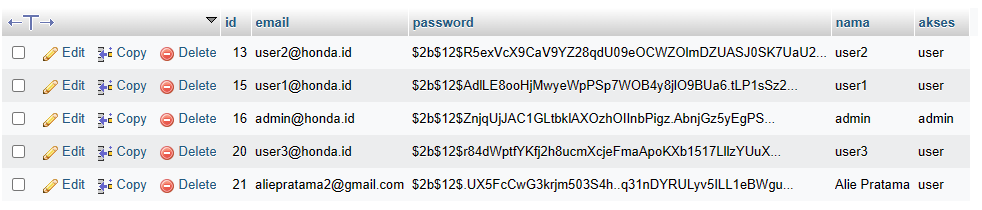
1. Mengisi data password

****

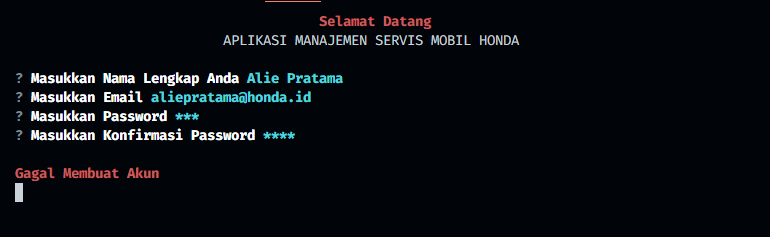
1. Proses membuat akun

****

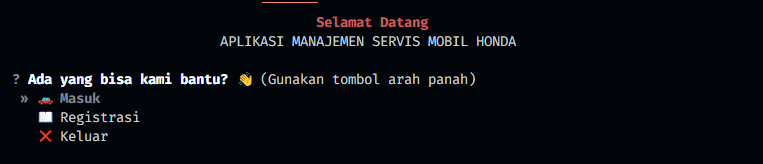
1. Data berhasil masuk pada database

****

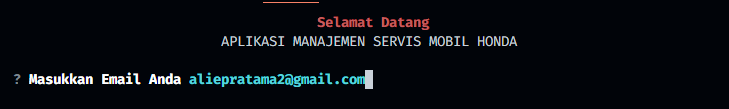
1. Tampilan jika gagal membuat akun

****

1. **Login**
2. Menu utama

****

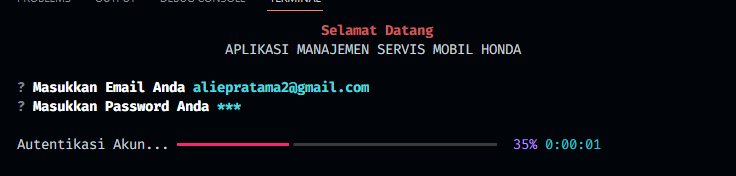
1. Masukkan email yang sudah terdaftar

****

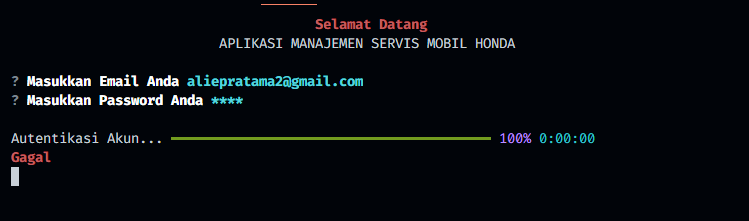
1. Masukkan password

****

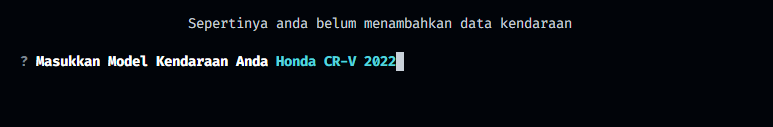
1. Proses autentikasi

****

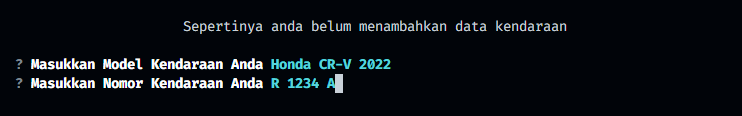
1. Tampilan jika pengguna gagal autentikasi

****

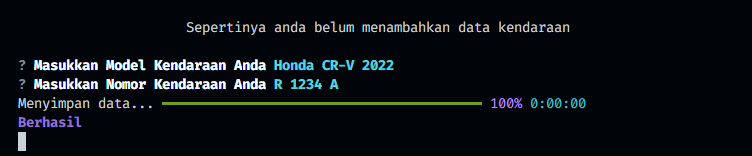
1. **Akses Pelanggan**
2. **Tampilan menu akun baru**
3. Meminta informasi model mobil

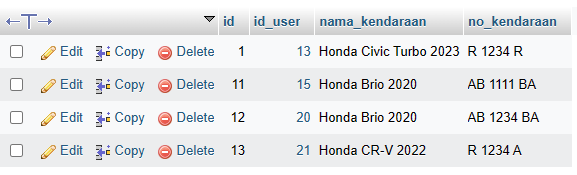
****

1. Meminta informasi nomor kendaraan

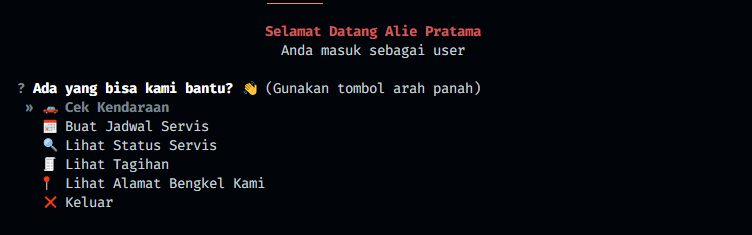
****

1. Data berhasil disimpan pada database

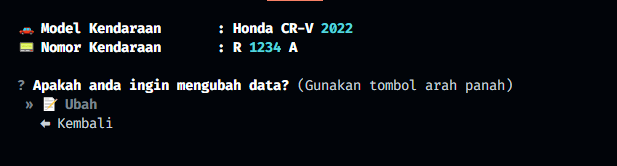
****

****

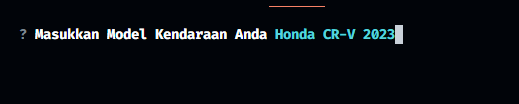
1. **Tampilan menu utama**

****

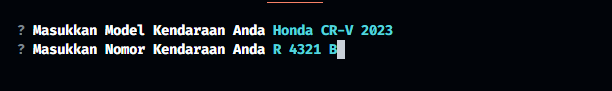
1. **Fitur cek kendaraan**
2. Menampilkan data kendaraan

****

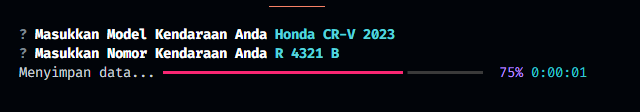
1. Mengubah informasi model mobil

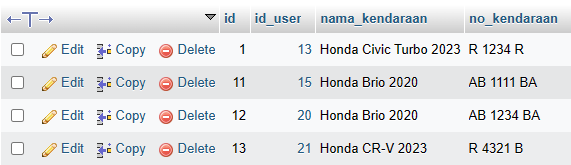
****

1. Mengubah informasi nomor kendaraan

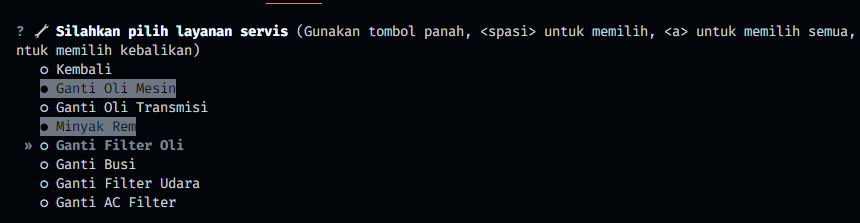
****

1. Data berhasil diubah pada database

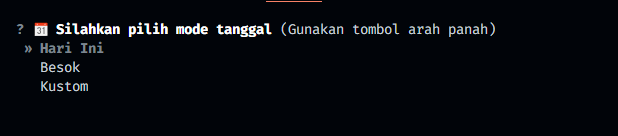
****

****

1. **Fitur buat jadwal servis**
2. Memilih layanan servis (bisa memilih dari 1 opsi)

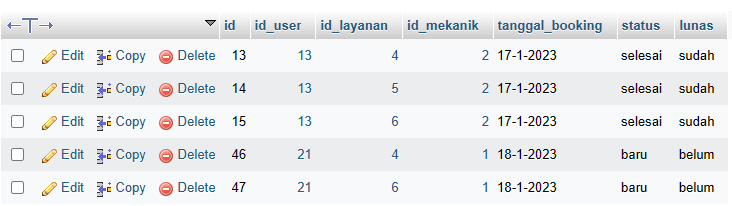
****

1. Memilih waktu servis

****

1. Data berhasil disimpan pada database

****

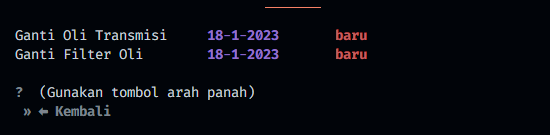
****

*db\_transaksi*

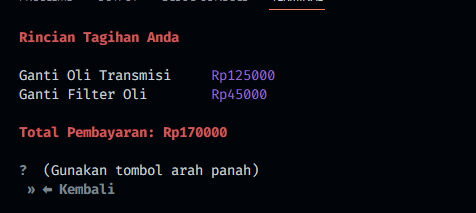
****

*db\_riwayat*

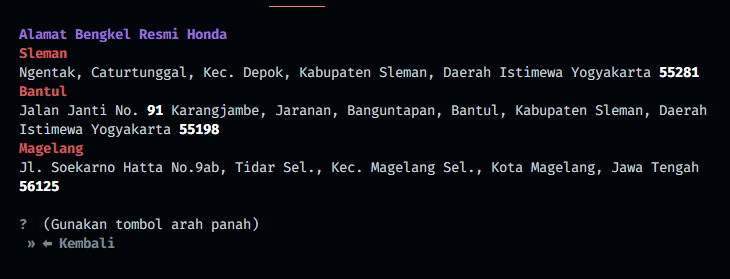
1. **Fitur lihat status servis**

****

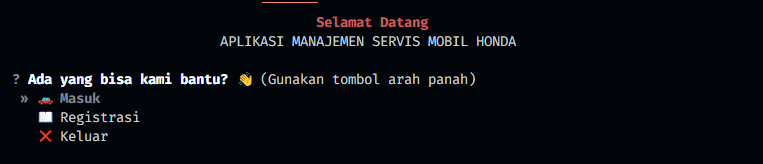
1. **Fitur lihat tagihan**

****

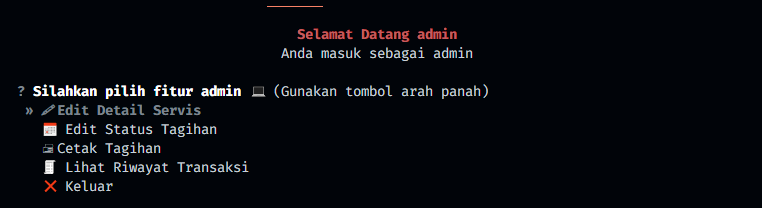
1. **Fitur lihat alamat bengkel**

****

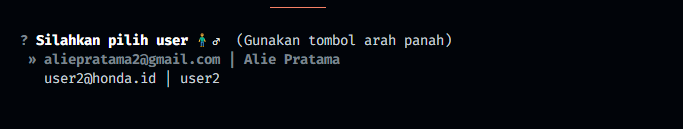
1. **Keluar/Logout**

****

1. **Akses Admin**
2. **Tampilan menu utama**

****

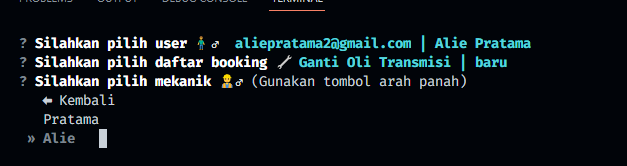
1. **Fitur edit detail servis**
2. Memilih data user/pelanggan

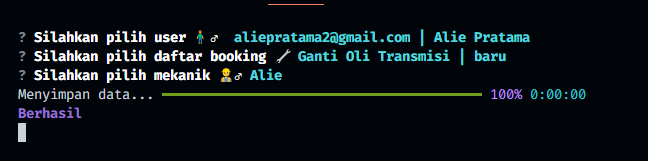
****

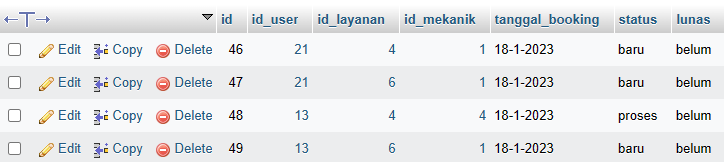
1. Memilih data booking

****

1. Jika data booking masih status ‘baru’, maka admin dapat memilih mekaniknya terlebih dahulu. Status akan otomatis menjadi ‘proses’

****

****

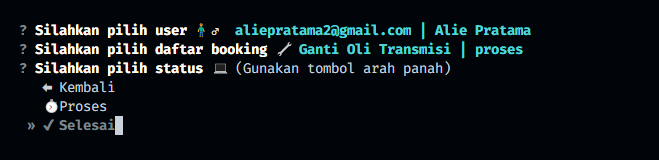
****

*db\_transaksi*

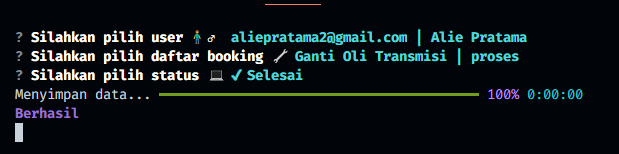
****

*db\_riwayat*

1. Jika data booking status selain ‘baru’, maka admin dapat memilih status menjadi ‘proses’ atau ‘selesai’

****

1. Data berhasil diubah pada database

****

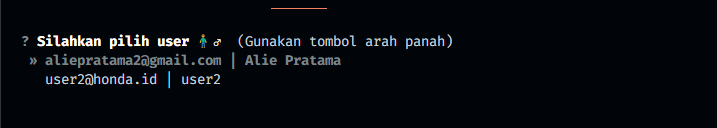
****

*db\_transaksi*

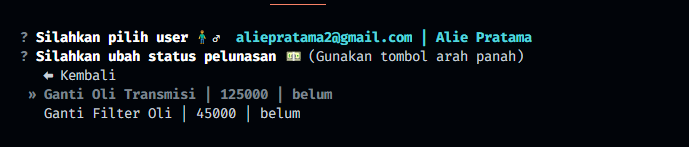
****

*db\_riwayat*

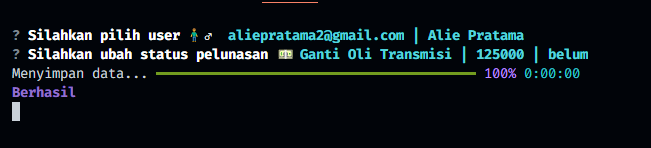
1. **Fitur edit status tagihan**
2. Memilih data user/pelanggan

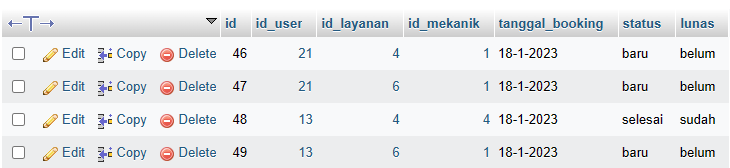
****

1. Memilih data booking (Status lunas akan berubah menjadi ‘sudah’, jika status sebelumnya ‘belum’, berlaku sebaliknya)

****

1. Data berhasil diubah pada database

****

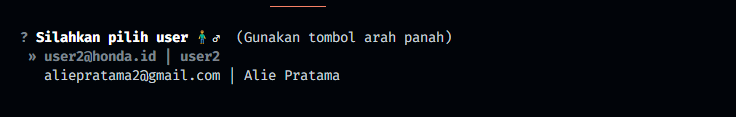
****

*db\_transaksi*

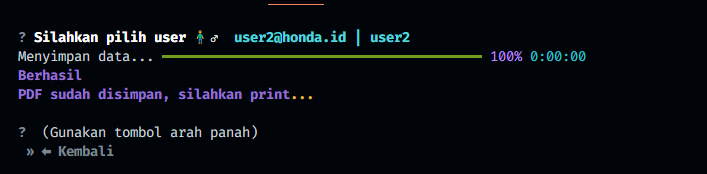
****

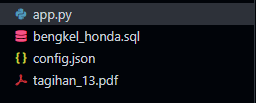
*db\_riwayat*

1. **Fitur cetak tagihan**
2. Memilih data user/pelanggan

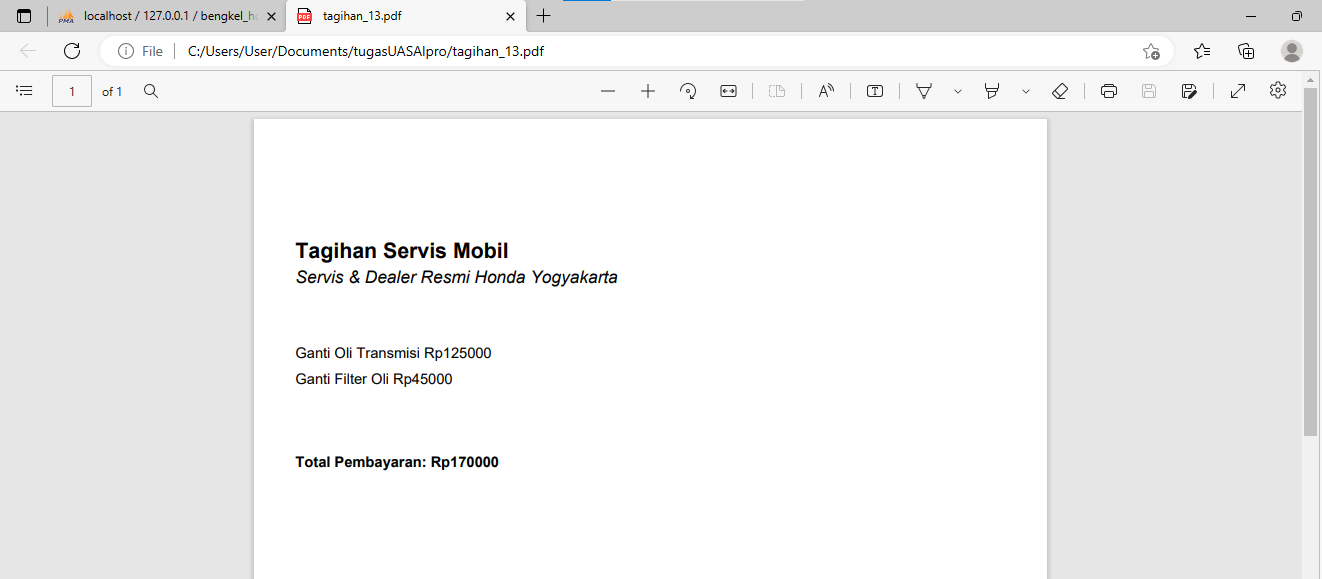
****

1. Hasil tagihan akan muncul di folder dengan format pdf

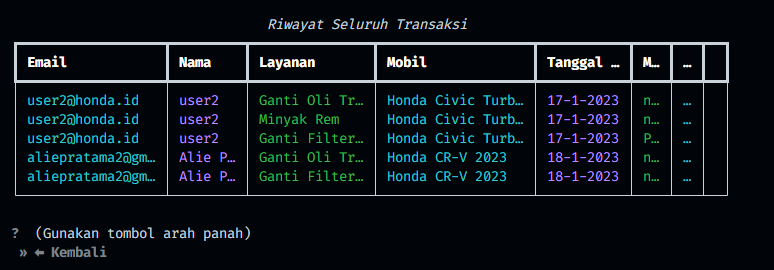
****

****

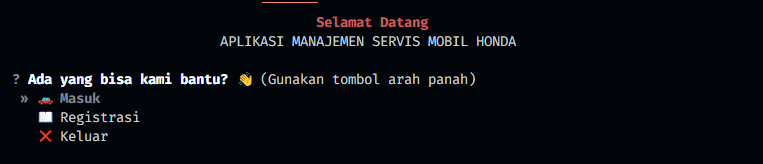
1. Berikut tampilan pdf dari tagihan pengguna

****

1. **Fitur lihat riwayat transaksi**

****

1. **Keluar/Logout**

****