

LAPORAN
TUGAS AHIR
APLIKASI MOBILE BANKING
Mata Kuliah Pemrograman Berorientasi Objek



Disusun Oleh:

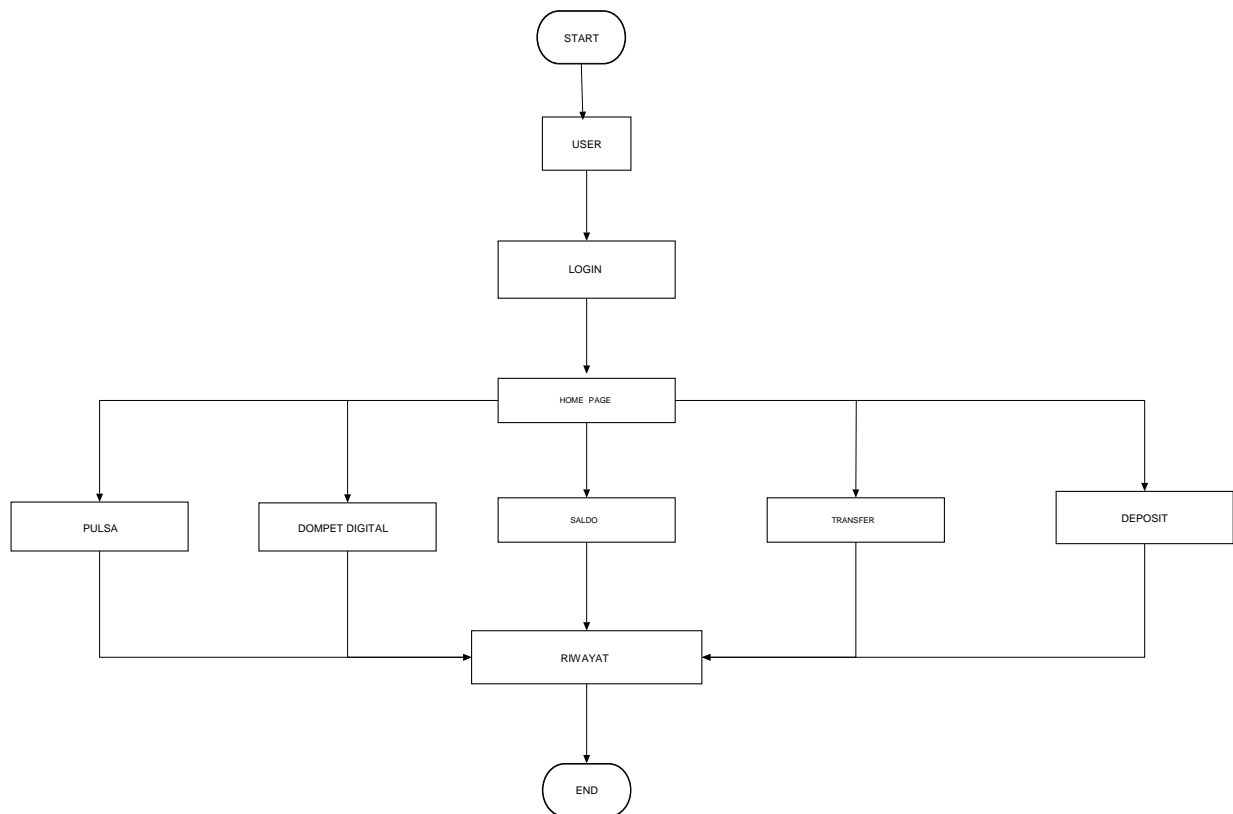
- | | |
|------------------------|--------------|
| 1. Yoga prasetyo | (2213020106) |
| 2. Alvin Arya | (2213020222) |
| 3. Frizky Wahyu Andika | (2213020178) |

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK DAN ILMU KOMPUTER
UNIVERSITAS NUSANTARA PGRI KEDIRI
TAHUN 2023

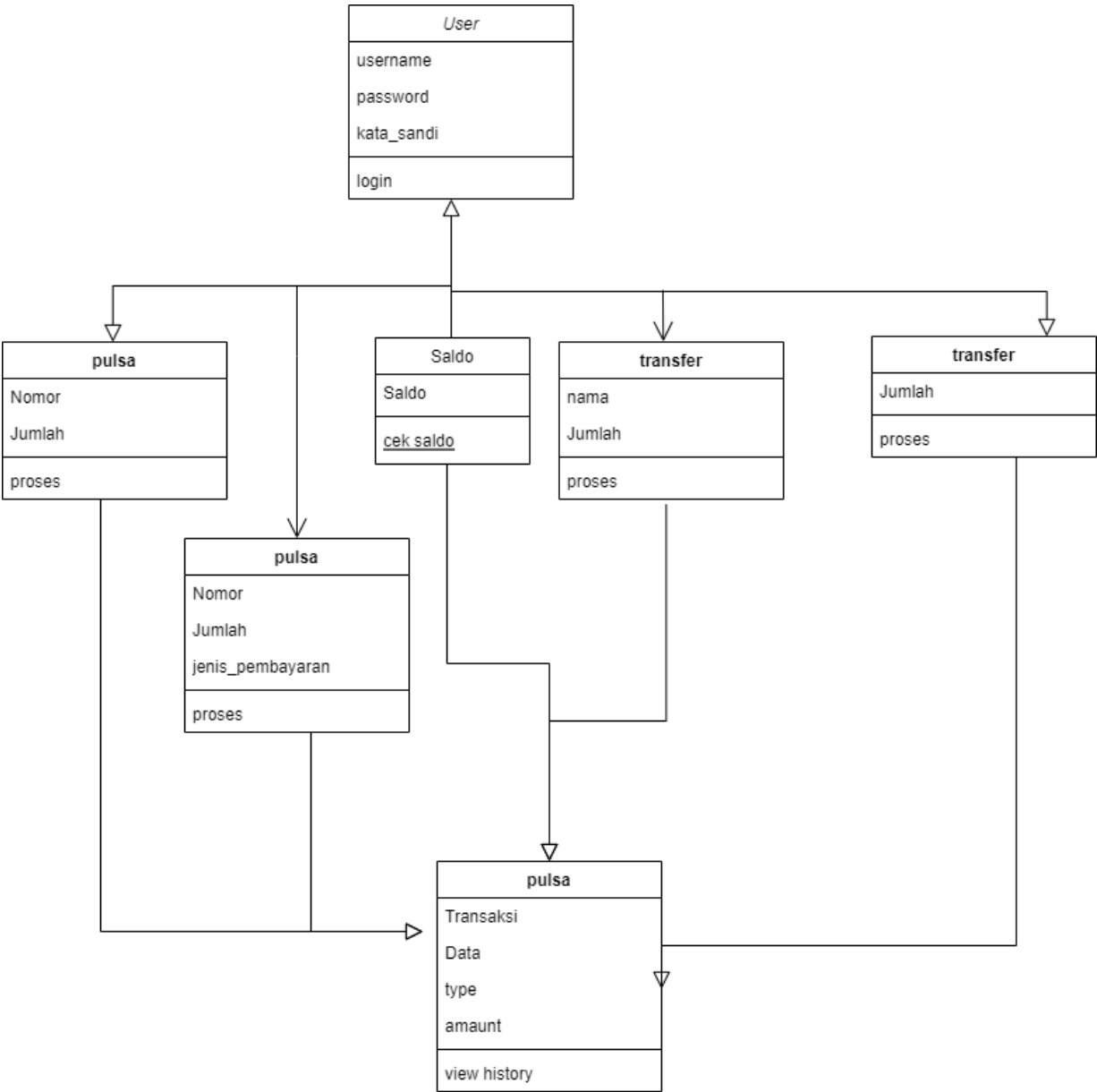
DAFTAR ISI

DAFTAR ISI.....	2
Flowchat sistem Mobile Banking	Error! Bookmark not defined.
Class diagram	Error! Bookmark not defined.
Hasil program dan penjelasan.....	Error! Bookmark not defined.
Daftar pustaka.....	19

Flowchat sistem Mobile Banking



Class diagram



Hasil program dan penjelasan

```
from PyQt5 import QtCore, QtGui, QtWidgets
import mysql.connector
import sys
class HalamanUtama(QtWidgets.QWidget):
    def __init__(self, username, user_id, cursor, connection):
        super().__init__()
        self.setWindowTitle("Mobile Banking")
        self.setGeometry(100, 100, 400, 300)
        self.username = username
        self.user_id = user_id
        self.cursor = cursor
        self.connection = connection
        layout = QtWidgets.QVBoxLayout(self)

        # Menambahkan widget ke dalam layout
        label_gambar = QtWidgets.QLabel(self)
        pixmap = QtGui.QPixmap("bank.png") # Ganti dengan path ke gambar Anda
        pixmap_terukur = pixmap.scaled(200, 200, QtCore.Qt.KeepAspectRatio)
        label_gambar.setPixmap(pixmap_terukur)
        label_gambar.setAlignment(QtCore.Qt.AlignCenter)
        layout.addWidget(label_gambar)

        label_selamat_datang = QtWidgets.QLabel(f"Selamat datang di Mobile
Banking, {username}!", self)
        label_selamat_datang.setAlignment(QtCore.Qt.AlignCenter)
        layout.addWidget(label_selamat_datang)

        tombol_saldo = QtWidgets.QPushButton("Saldo", self)
        tombol_saldo.clicked.connect(self.tampilkan_saldo)
        layout.addWidget(tombol_saldo)

        tombol_transfer = QtWidgets.QPushButton("Transfer", self)
        tombol_transfer.clicked.connect(self.tampilkan_menu_transfer)
        layout.addWidget(tombol_transfer)

        tombol_deposit = QtWidgets.QPushButton("Deposit", self)
        tombol_deposit.clicked.connect(self.tampilkan_menu_deposit)
        layout.addWidget(tombol_deposit)

        tombol_pulsa = QtWidgets.QPushButton("Pulsa", self)
        tombol_pulsa.clicked.connect(self.tampilkan_menu_pulsa)
        layout.addWidget(tombol_pulsa)

        tombol_dompet_digital = QtWidgets.QPushButton("Dompet Digital", self)
        tombol_dompet_digital.clicked.connect(self.tampilkan_menu_dompet_digital)
        layout.addWidget(tombol_dompet_digital)
```

```

        tombol_riwayat = QtWidgets.QPushButton("Riwayat", self)
        tombol_riwayat.clicked.connect(self.tampilkan_menu_riwayat)
        layout.addWidget(tombol_riwayat)
def tampilkan_menu_riwayat(self):
    # Ambil transaksi terbaru setiap kali menu riwayat ditampilkan
    transaksi = self.ambil_transaksi_terbaru()

    menu_riwayat = QtWidgets.QDialog(self)
    menu_riwayat.setWindowTitle("Riwayat Transaksi")
    menu_riwayat.setGeometry(200, 200, 600, 400)

    tabel = QtWidgets.QTableWidget(menu_riwayat)
    tabel.setColumnCount(3)
    tabel.setHorizontalHeaderLabels(["Tanggal", "Jenis", "Jumlah"])

    for baris, transaksi in enumerate(transaksi):
        tabel.insertRow(baris)
        tabel.setItem(baris, 0,
QtWidgets.QTableWidgetItem(transaksi["tanggal"]))
        tabel.setItem(baris, 1, QtWidgets.QTableWidgetItem(transaksi["jenis"]))
        tabel.setItem(baris, 2,
QtWidgets.QTableWidgetItem(str(transaksi["jumlah"])))

    tombol_keluar = QtWidgets.QPushButton("Kembali", menu_riwayat)
    tombol_keluar.clicked.connect(menu_riwayat.reject)

    layout_grid = QtWidgets.QGridLayout(menu_riwayat)
    layout_grid.addWidget(tabel, 0, 0, 1, 2)
    layout_grid.addWidget(tombol_keluar, 1, 0, 1, 2)

    menu_riwayat.exec_()
def fetch_latest_transactions(self):
    try:
        query = "SELECT date, type, amount FROM transactions_history WHERE
user_id = %s ORDER BY date DESC"
        self.cursor.execute(query, (self.user_id,))
        latest_transactions = self.cursor.fetchall()

        # Print for debugging
        print("Latest Transactions:", latest_transactions)

        transactions_list = []
        for transaction in latest_transactions:
            # Convert date to a human-readable format
            formatted_date = datetime.strptime(transaction[0], "%Y-%m-%d")

            transactions_list.append({
                "date": formatted_date,

```

```

        "type": transaction[1],
        "amount": transaction[2]
    })

    return transactions_list
except Exception as e:
    print(f"Error fetching transactions_history: {e}")
    return []

def tampilkan_menu_pulsa(self):
    menu_pulsa = QtWidgets.QDialog(self)
    menu_pulsa.setWindowTitle("Menu Pulsa")
    menu_pulsa.setGeometry(200, 200, 400, 300)

    # Tambahkan QLabel untuk ikon pulsa
    label_icon_pulsa = QtWidgets.QLabel(menu_pulsa)
    pixmap_icon_pulsa = QtGui.QPixmap("pulsa.png") # Ganti dengan path ke ikon
pulsa Anda
    pixmap_icon_pulsa_terukur = pixmap_icon_pulsa.scaled(100, 100,
QtCore.Qt.KeepAspectRatio)
    label_icon_pulsa.setPixmap(pixmap_icon_pulsa_terukur)
    label_icon_pulsa.setAlignment(QtCore.Qt.AlignCenter)

    label_nomor_hp = QtWidgets.QLabel("Nomor HP:", menu_pulsa)
    label_pembelian = QtWidgets.QLabel("Pilihan Pembelian:", menu_pulsa)

    self.entry_nomor_hp = QtWidgets.QLineEdit(menu_pulsa)

    # Buat tombol untuk setiap opsi pembelian dengan label seragam
    tombol_5000 = QtWidgets.QPushButton("Rp. 5.000", menu_pulsa)
    tombol_10000 = QtWidgets.QPushButton("Rp. 10.000", menu_pulsa)
    tombol_15000 = QtWidgets.QPushButton("Rp. 15.000", menu_pulsa)
    tombol_20000 = QtWidgets.QPushButton("Rp. 20.000", menu_pulsa)
    tombol_25000 = QtWidgets.QPushButton("Rp. 25.000", menu_pulsa)
    tombol_30000 = QtWidgets.QPushButton("Rp. 30.000", menu_pulsa)

    # Hubungkan setiap tombol ke fungsi beli_pulsa dengan jumlah yang sesuai
    tombol_5000.clicked.connect(lambda: self.beli_pulsa(5000))
    tombol_10000.clicked.connect(lambda: self.beli_pulsa(10000))
    tombol_15000.clicked.connect(lambda: self.beli_pulsa(15000))
    tombol_20000.clicked.connect(lambda: self.beli_pulsa(20000))
    tombol_25000.clicked.connect(lambda: self.beli_pulsa(25000))
    tombol_30000.clicked.connect(lambda: self.beli_pulsa(30000))

    tombol_keluar = QtWidgets.QPushButton("Kembali", menu_pulsa)
    tombol_keluar.clicked.connect(menu_pulsa.reject)

```

```

layout_grid = QtWidgets.QGridLayout(menu_pulsa)
layout_grid.addWidget(label_nomor_hp, 0, 0)
layout_grid.addWidget(self.entry_nomor_hp, 0, 1)
layout_grid.addWidget(label_pembelian, 1, 0)

# Tambahkan tombol ke dalam layout
layout_grid.addWidget(tombol_5000, 1, 1)
layout_grid.addWidget(tombol_10000, 1, 2)
layout_grid.addWidget(tombol_15000, 2, 1)
layout_grid.addWidget(tombol_20000, 2, 2)
layout_grid.addWidget(tombol_25000, 3, 1)
layout_grid.addWidget(tombol_30000, 3, 2)

# Tambahkan QLabel untuk ikon pulsa
layout_grid.addWidget(label_icon_pulsa, 0, 3, 4, 1)
layout_grid.addWidget(tombol_keluar, 4, 0, 1, 4)
menu_pulsa.exec_()
def beli_pulsa(self, jumlah_beli):
    try:
        # Dapatkan informasi pulsa dari field input
        nomor_hp = self.entry_nomor_hp.text()

        # Lakukan validasi dan operasi pembelian pulsa
        if nomor_hp and jumlah_beli > 0:
            # Panggil fungsi untuk menangani pembelian pulsa
            self.beli_pulsa(nomor_hp, jumlah_beli)

            QtWidgets.QMessageBox.information(
                self,
                "Pembelian Pulsa Berhasil",
                f"Pembelian pulsa sejumlah {jumlah_beli} untuk {nomor_hp}"
                berhasil."
            )
        else:
            QtWidgets.QMessageBox.warning(
                self,
                "Input Tidak Valid",
                "Harap masukkan informasi yang valid untuk pembelian pulsa."
            )
    except ValueError:
        QtWidgets.QMessageBox.warning(
            self,
            "Input Tidak Valid",
            "Harap masukkan jumlah numerik yang valid untuk pembelian pulsa."
        )
def beli_pulsa(self, nomor_hp, jumlah_beli):
    try:
        # Output debugging untuk mencetak Nomor HP yang diberikan
        print(f"Nomor HP yang Diberikan: {nomor_hp}")

```



```

        # Dapatkan user_id berdasarkan nomor_hp yang diberikan
        query_user_id = "SELECT id FROM users WHERE LOWER(TRIM(nomor_hp)) = LOWER(TRIM(%s))"
        self.cursor.execute(query_user_id, (nomor_hp,))
        hasil_user_id = self.cursor.fetchone()

        if hasil_user_id is not None:
            user_id = hasil_user_id[0]

            # Lakukan logika untuk menangani pembelian pulsa, sebagai contoh, mengurangi saldo
            # Untuk kesederhanaan, mari anggap pembelian pulsa hanya mengurangi saldo
            query_saldo = "SELECT saldo FROM users WHERE id = %s"
            self.cursor.execute(query_saldo, (user_id,))
            hasil_saldo = self.cursor.fetchone()

            if hasil_saldo is not None:
                saldo_sekarang = hasil_saldo[0]
                # Kurangkan jumlah dari saldo sekarang
                saldo_baru = saldo_sekarang - jumlah_beli
                # Perbarui saldo di database
                query_perbarui = "UPDATE users SET saldo = %s WHERE id = %s"
                self.cursor.execute(query_perbarui, (saldo_baru, user_id))
                self.connection.commit()

                # Masukkan informasi pembelian pulsa ke dalam tabel pulsa_purchases
                query_masukkan_pembelian_pulsa = "INSERT INTO pulsa_purchases (user_id, nomor_hp, jumlah_beli) VALUES (%s, %s, %s)"
                self.cursor.execute(query_masukkan_pembelian_pulsa, (user_id, nomor_hp, jumlah_beli))
                self.connection.commit()

                print(f"Pembelian pulsa berhasil untuk pengguna {user_id}. Saldo baru: {saldo_baru}")
            else:
                print("Pengguna tidak ditemukan.")
        else:
            print("Nomor HP tidak ditemukan.")
    except Exception as e:
        print(f"Error dalam beli_pulsa: {e}")

    def tampilkan_menu_dompeter_digital(self):
        menu_dompeter_digital = QtWidgets.QDialog(self)
        menu_dompeter_digital.setWindowTitle("Dompeter Digital")
        menu_dompeter_digital.setGeometry(200, 200, 400, 300)

        # Tambahkan QLabel untuk gambar dompeter digital
        label_gambar_dompeter_digital = QtWidgets.QLabel(menu_dompeter_digital)

```

```

        pixmap_gambar_dompét_digital = QtGui.QPixmap("wallet.png")
        pixmap_dompét_digital_terukur =
pixmap_gambar_dompét_digital.scaled(100, 100, QtCore.Qt.KeepAspectRatio)
        label_gambar_dompét_digital.setPixmap(pixmap_dompét_digital_terukur)
        label_gambar_dompét_digital.setAlignment(QtCore.Qt.AlignCenter)

        label_nomor = QtWidgets.QLabel("Nomor:", menu_dompét_digital)
        label_jumlah = QtWidgets.QLabel("Jumlah:", menu_dompét_digital)
        label_jenis_dompét = QtWidgets.QLabel("Jenis Dompét:",
menu_dompét_digital)

    # Gunakan atribut instance di sini
    self.entry_nomor = QtWidgets.QLineEdit(menu_dompét_digital)
    self.entry_jumlah = QtWidgets.QLineEdit(menu_dompét_digital)
    self.combo_jenis_dompét = QtWidgets.QComboBox(menu_dompét_digital)
    self.combo_jenis_dompét.addItem(["OVO", "Shopee", "Dana"])

    tombol_top_up = QtWidgets.QPushButton("Top Up", menu_dompét_digital)
    tombol_top_up.clicked.connect(self.top_up_amount)
    tombol_kembali = QtWidgets.QPushButton("Kembali", menu_dompét_digital)
    tombol_kembali.clicked.connect(menu_dompét_digital.reject)

    layout_grid = QtWidgets.QGridLayout(menu_dompét_digital)
    layout_grid.addWidget(label_gambar_dompét_digital, 0, 0, 1, 2) #
Tambahkan label gambar
    layout_grid.addWidget(label_nomor, 1, 0)
    layout_grid.addWidget(self.entry_nomor, 1, 1)
    layout_grid.addWidget(label_jumlah, 2, 0)
    layout_grid.addWidget(self.entry_jumlah, 2, 1)
    layout_grid.addWidget(label_jenis_dompét, 3, 0)
    layout_grid.addWidget(self.combo_jenis_dompét, 3, 1)
    layout_grid.addWidget(tombol_top_up, 4, 0, 1, 2)
    layout_grid.addWidget(tombol_kembali, 5, 0, 1, 2)
    menu_dompét_digital.exec_()

def top_up_amount(self):
    try:
        # Dapatkan informasi top-up dari field input
        nomor = self.entry_nomor.text()
        jumlah = float(self.entry_jumlah.text())
        jenis_dompét = self.combo_jenis_dompét.currentText()

        # Lakukan validasi dan operasi top-up
        if nomor and jumlah > 0:
            # Top-up saldo (panggil fungsi yang benar)
            self.top_up_balance(jumlah)
            # Hubungkan dompét digital ke akun pengguna
            self.connect_digital_wallet(nomor, jenis_dompét)
            QtWidgets.QMessageBox.information(

```

```

        self,
        "Top Up Berhasil",
        f"Top Up sejumlah {jumlah} ke {jenis_dompot} dengan Nomor
{nomor} berhasil.")
    else:
        QtWidgets.QMessageBox.warning(
            self, "Input Tidak Valid", "Harap masukkan informasi yang valid
untuk top-up.")
    except ValueError:
        QtWidgets.QMessageBox.warning(
            self, "Input Tidak Valid", "Harap masukkan jumlah numerik yang valid
untuk top-up.")

def connect_digital_wallet(self, nomor_dompot, jenis_dompot):
    try:
        # Periksa apakah pengguna sudah memiliki dompet digital
        query_periksa = "SELECT * FROM digital_wallets WHERE user_id = %s"
        self.cursor.execute(query_periksa, (self.user_id,))
        dompet_digital_ada = self.cursor.fetchone()

        if dompet_digital_ada:
            # Perbarui dompet yang sudah ada
            query_perbarui = "UPDATE digital_wallets SET nomor_dompot = %s,
jenis_dompot = %s WHERE user_id = %s"
            self.cursor.execute(query_perbarui, (nomor_dompot, jenis_dompot,
self.user_id))
        else:
            # Masukkan entri dompet baru
            query_masukkan = "INSERT INTO digital_wallets (user_id,
nomor_dompot, jenis_dompot) VALUES (%s, %s, %s)"
            self.cursor.execute(query_masukkan, (self.user_id, nomor_dompot,
jenis_dompot))
            self.connection.commit()

        print(f"Dompot digital terhubung dengan sukses. Nomor Dompot:
{nomor_dompot}, Jenis Dompot: {jenis_dompot}")
    except Exception as e:
        print(f"Error dalam connect_digital_wallet: {e}")

def top_up_balance(self, jumlah):
    try:
        # Query saldo sekarang dari database
        query_saldo = "SELECT saldo FROM users WHERE id = %s"
        self.cursor.execute(query_saldo, (self.user_id,))
        hasil_saldo = self.cursor.fetchone()
        if hasil_saldo is not None:
            saldo_sekarang = hasil_saldo[0]
            # Top-up sejumlah ke saldo sekarang
            saldo_baru = saldo_sekarang + jumlah

```

```

        # Perbarui saldo di database
        query_perbarui = "UPDATE users SET saldo = %s WHERE id = %s"
        self.cursor.execute(query_perbarui, (saldo_baru, self.user_id))
        self.connection.commit()
        print(f"Top-up berhasil. Saldo baru: {saldo_baru}")
    else:
        print("Pengguna tidak ditemukan.")
except Exception as e:
    print(f"Error dalam top_up_balance: {e}")
def tampilkan_saldo(self):
    dialog_saldo = QtWidgets.QDialog(self)
    dialog_saldo.setWindowTitle("Saldo Anda")
    dialog_saldo.setGeometry(200, 200, 400, 250)

    query_saldo = "SELECT saldo FROM users WHERE id = %s"
    self.cursor.execute(query_saldo, (self.user_id,))
    hasil_saldo = self.cursor.fetchone()

    if hasil_saldo is not None:
        jumlah_saldo = hasil_saldo[0]
        saldo_terformat = "{:,}".format(jumlah_saldo)
        label_saldo = QtWidgets.QLabel(f"Saldo Anda: Rp {saldo_terformat}",
dialog_saldo)
        label_saldo.setAlignment(QtCore.Qt.AlignCenter)

        # Menambahkan gambar ATM
        label_gambar = QtWidgets.QLabel(dialog_saldo)
        pixmap = QtGui.QPixmap("kartu.png") # Ganti dengan path ke gambar Anda
        pixmap_terukur = pixmap.scaledToWidth(200)
        label_gambar.setPixmap(pixmap_terukur)
        label_gambar.setAlignment(QtCore.Qt.AlignCenter)

        tombol_ok = QtWidgets.QPushButton("OK", dialog_saldo)
        tombol_ok.clicked.connect(dialog_saldo.accept)

        layout_grid = QtWidgets.QGridLayout(dialog_saldo)
        layout_grid.addWidget(label_gambar, 0, 0, 1, 2) # Tambahkan gambar ATM
        layout_grid.addWidget(label_saldo, 1, 0, 1, 2)
        layout_grid.addWidget(tombol_ok, 2, 0, 1, 2)
        dialog_saldo.exec_()
    else:
        QtWidgets.QMessageBox.warning(
            self, "Error", "Gagal mendapatkan informasi saldo. Silakan coba
lagi.")
def tampilkan_menu_transfer(self):
    menu_transfer = QtWidgets.QDialog(self)
    menu_transfer.setWindowTitle("Menu Transfer")
    menu_transfer.setGeometry(200, 200, 400, 300)

```

```

# Tambahkan widget untuk gambar transfer
label_gambar_transfer = QtWidgets.QLabel(menu_transfer)
pixmap_gambar_transfer = QtGui.QPixmap("transfer.jpg")
pixmap_terukur_transfer = pixmap_gambar_transfer.scaled(100, 100,
QtCore.Qt.KeepAspectRatio)
label_gambar_transfer.setPixmap(pixmap_terukur_transfer)
label_gambar_transfer.setAlignment(QtCore.Qt.AlignCenter)

# Tambahkan widget untuk detail transfer
label_pengirim = QtWidgets.QLabel(f"Pengirim: {self.username}",
menu_transfer)
label_penerima = QtWidgets.QLabel("Penerima:", menu_transfer)
label_jumlah = QtWidgets.QLabel("Jumlah:", menu_transfer)
entry_penerima = QtWidgets.QLineEdit(menu_transfer)
entry_jumlah = QtWidgets.QLineEdit(menu_transfer)

tombol_transfer = QtWidgets.QPushButton("Transfer", menu_transfer)
tombol_transfer.clicked.connect(lambda:
self.transfer_amount(entry_penerima.text(), entry_jumlah.text()))
tombol_kembali = QtWidgets.QPushButton("Kembali", menu_transfer)
tombol_kembali.clicked.connect(menu_transfer.reject)

# Atur tata letak untuk menu transfer
layout_grid = QtWidgets.QGridLayout(menu_transfer)
layout_grid.addWidget(label_gambar_transfer, 0, 0, 1, 2)
layout_grid.addWidget(label_pengirim, 1, 0, 1, 2)
layout_grid.addWidget(label_penerima, 2, 0)
layout_grid.addWidget(entry_penerima, 2, 1)
layout_grid.addWidget(label_jumlah, 3, 0)
layout_grid.addWidget(entry_jumlah, 3, 1)
layout_grid.addWidget(tombol_transfer, 4, 0, 1, 2)
layout_grid.addWidget(tombol_kembali, 5, 0, 1, 2)
menu_transfer.exec_()

def transfer_amount(self, username_penerima, jumlah_str):
    try:
        # Dapatkan ID pengirim
        query_id_pengirim = "SELECT id FROM users WHERE username = %s"
        self.cursor.execute(query_id_pengirim, (self.username,))
        hasil_pengirim = self.cursor.fetchone()

        if hasil_pengirim:
            id_pengirim = hasil_pengirim[0]

            # Dapatkan ID penerima
            query_id_penerima = "SELECT id FROM users WHERE username = %s"
            self.cursor.execute(query_id_penerima, (username_penerima,))
            hasil_penerima = self.cursor.fetchone()

```

```

        if hasil_penerima:
            id_penerima = hasil_penerima[0]
            # Konversi jumlah ke integer
            jumlah = int(jumlah_str)
            # Periksa apakah pengirim memiliki saldo yang cukup
            if jumlah > 0:
                # Perbarui saldo pengirim
                query_perbarui_saldo_pengirim = "UPDATE users SET saldo =
saldo - %s WHERE id = %s"
                self.cursor.execute(query_perbarui_saldo_pengirim, (jumlah,
id_pengirim))
                # Perbarui saldo penerima
                query_perbarui_saldo_penerima = "UPDATE users SET saldo =
saldo + %s WHERE id = %s"
                self.cursor.execute(query_perbarui_saldo_penerima, (jumlah,
id_penerima))
                # Masukkan rekam transfer
                query_masukkan_transfer = "INSERT INTO transfers
(sender_id, receiver_id, amount) VALUES (%s, %s, %s)"
                self.cursor.execute(query_masukkan_transfer, (id_pengirim,
id_penerima, jumlah))
                # Commit perubahan ke database
                self.connection.commit()

                # Tampilkan pesan sukses (sesuaikan jika diperlukan)
                QtWidgets.QMessageBox.information(
                    self, "Transfer Berhasil", f"Berhasil mentransfer
{jumlah} ke {username_penerima}." )
            else:
                # Tampilkan pesan jumlah tidak valid (sesuaikan jika
diperlukan)
                QtWidgets.QMessageBox.warning(
                    self, "Transfer Gagal", "Jumlah tidak valid. Silakan coba
lagi.")
        else:
            # Tampilkan pesan penerima tidak ditemukan (sesuaikan jika
diperlukan)
            QtWidgets.QMessageBox.warning(
                self, "Transfer Gagal", f"Penerima dengan nama pengguna
{username_penerima} tidak ditemukan.")
        else:
            # Tampilkan pesan pengirim tidak ditemukan (sesuaikan jika
diperlukan)
            QtWidgets.QMessageBox.warning(
                self, "Transfer Gagal", f"Pengirim dengan nama pengguna
{self.username} tidak ditemukan.")
    except mysql.connector.Error as err:
        print("Error MySQL: {}".format(err))

```

```

        QtWidgets.QMessageBox.critical(
            self, "Error", f"Terjadi kesalahan saat melakukan transfer: {err}")
    import traceback
    traceback.print_exc()

def tampilkan_menu_deposit(self):
    menu_deposit = QtWidgets.QDialog(self)
    menu_deposit.setWindowTitle("Menu Deposit")
    menu_deposit.setGeometry(200, 200, 400, 300)
    # Tambahkan widget untuk detail deposit
    label_jumlah = QtWidgets.QLabel("Jumlah:", menu_deposit)
    entry_jumlah = QtWidgets.QLineEdit(menu_deposit)

    tombol_deposit = QtWidgets.QPushButton("Deposit", menu_deposit)
    tombol_deposit.clicked.connect(lambda:
self.deposit_amount(entry_jumlah.text()))
    tombol_kembali = QtWidgets.QPushButton("Kembali", menu_deposit)
    tombol_kembali.clicked.connect(menu_deposit.reject)

    # Tambahkan widget untuk gambar deposito
    label_gambar_deposito = QtWidgets.QLabel(menu_deposit)
    pixmap = QtGui.QPixmap("deposito.png")
    pixmap_terukur = pixmap.scaled(100, 100, QtCore.Qt.KeepAspectRatio)
    label_gambar_deposito.setPixmap(pixmap_terukur)
    label_gambar_deposito.setAlignment(QtCore.Qt.AlignCenter)

    # Atur tata letak untuk menu deposit
    layout_grid = QtWidgets.QGridLayout(menu_deposit)
    layout_grid.addWidget(label_gambar_deposito, 0, 0, 1, 2)
    layout_grid.addWidget(label_jumlah, 1, 0)
    layout_grid.addWidget(entry_jumlah, 1, 1)
    layout_grid.addWidget(tombol_deposit, 2, 0, 1, 2)
    layout_grid.addWidget(tombol_kembali, 3, 0, 1, 2)
    menu_deposit.exec_()

def deposito_jumlah(self, jumlah_str):
    try:
        # Konversi jumlah menjadi integer
        jumlah = int(jumlah_str)
        # Periksa apakah jumlah deposito valid
        if jumlah > 0:
            # Perbarui saldo pengguna
            query_perbarui_saldo = "UPDATE users SET saldo = saldo + %s WHERE
id = %s"
            self.cursor.execute(query_perbarui_saldo, (jumlah, self.user_id))
            # Masukkan catatan deposito
            query_masukkan_deposito = "INSERT INTO deposits (user_id, amount)
VALUES (%s, %s)"

```

```

        self.cursor.execute(query_masukkan_deposito, (self.user_id,
jumlah))
        # Komit perubahan ke database
        self.connection.commit()
        # Tampilkan pesan sukses (dapat disesuaikan)
        QtWidgets.QMessageBox.information(
            self, "Deposit Berhasil", f"Berhasil melakukan deposito sejumlah
{jumlah}.")
        else:
            # Tampilkan pesan jumlah deposito tidak valid (dapat disesuaikan)
            QtWidgets.QMessageBox.warning(
                self, "Deposit Gagal", "Jumlah deposito tidak valid. Silakan
coba lagi.")
        except mysql.connector.Error as err:
            print("Error MySQL: {}".format(err))
            QtWidgets.QMessageBox.critical(
                self, "Error", f"Terjadi kesalahan saat melakukan deposito: {err}")
        import traceback
        traceback.print_exc()

class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(330, 442)
        Form.setWindowFlags(QtCore.Qt.FramelessWindowHint)
        Form.setAttribute(QtCore.Qt.WA_TranslucentBackground)
        Form.setStyleSheet("QPushButton#pushButton{\n""background-color:rgba(2,
65, 118, 255);\n""color:rgba(255, 255, 255, 200);\n""border-radius:5px;\n"
            "}\n""QPushButton#pushButton:pressed{\n""padding-
left:5px;\n""padding-top:5px;\n""background-color:rgba(2, 65, 118, 100);\n"
            "background-position:calc(100% -
10px)center;\n""}\n""QPushButton#pushButton:hover{\n""background-color:rgba(2,
65, 118, 200);\n""}")
        self.widget = QtWidgets.QWidget(Form)
        self.widget.setGeometry(QtCore.QRect(10, 10, 290, 410))
        self.widget.setObjectName("widget")
        self.label = QtWidgets.QLabel(self.widget)
        self.label.setGeometry(QtCore.QRect(0, 0, 290, 410))
        self.label.setStyleSheet("background-color:rgba(16, 30, 41,
240);\n""border-radius:10px;")
        self.label.setText("")
        self.label.setObjectName("label")
        self.lineEdit = QtWidgets.QLineEdit(self.widget)
        self.lineEdit.setGeometry(QtCore.QRect(20, 210, 250, 30))
        font = QtGui.QFont()
        font.setPointSize(10)
        self.lineEdit.setFont(font)

```



```

        self.lineEdit.setStyleSheet("background-color:rgba(0, 0, 0,
0);\n""border:1px solid rgba(0, 0, 0, 0);\n""border-bottom-color:rgba(46, 82,
101, 255);\n"
                                   "color:rgb(255, 255, 255);\n""padding-
bottom:7px")
        self.lineEdit.setObjectName("lineEdit")
        self.lineEdit_2 = QtWidgets.QLineEdit(self.widget)
        self.lineEdit_2.setGeometry(QtCore.QRect(20, 260, 250, 30))
        font = QtGui.QFont()
        font.setPointSize(10)
        self.lineEdit_2.setFont(font)
        self.lineEdit_2.setStyleSheet("background-color:rgba(0, 0, 0, 0);\n"
"border:1px solid rgba(0, 0, 0, 0);\n""border-bottom-color:rgba(46, 82, 101,
255);\n"
                                   "color:rgb(255, 255, 255);\n""padding-
bottom:7px")
        self.lineEdit_2.setEchoMode(QtWidgets.QLineEdit.Password)
        self.lineEdit_2.setObjectName("lineEdit_2")
        self.pushButton = QtWidgets.QPushButton(self.widget)
        self.pushButton.setGeometry(QtCore.QRect(20, 320, 250, 40))
        font = QtGui.QFont()
        font.setPointSize(10)
        font.setBold(True)
        font.setWeight(75)
        self.pushButton.setFont(font)
        self.pushButton.setObjectName("pushButton")
        self.label_2 = QtWidgets.QLabel(self.widget)
        self.label_2.setGeometry(QtCore.QRect(60, 30, 180, 150))
        original_pixmap = QtGui.QPixmap("icon orang.png") # Ganti dengan path
logo Anda
        scaled_pixmap = original_pixmap.scaled(180, 150,
QtCore.Qt.KeepAspectRatio)
        self.label_2.setPixmap(scaled_pixmap)
        self.label_2.setAlignment(QtCore.Qt.AlignCenter)
        self.label_2.setObjectName("label_2")
        self.label_3 = QtWidgets.QLabel(self.widget)
        self.label_3.setGeometry(QtCore.QRect(50, 365, 211, 16))
        self.label_3.setStyleSheet("color:rgba(255, 255, 255, 150);")
        self.label_3.setObjectName("label_3")
        # Menu setup
        self.menuBar = QtWidgets.QMenuBar(Form)
        self.menuBar.setGeometry(QtCore.QRect(0, 0, 330, 21))
        self.menuBar.setObjectName("menuBar")
        self.menuBar.setStyleSheet("background-color: rgba(16, 30, 41, 240);
color: white;")
        self.menuBar.setGeometry(QtCore.QRect(0, 0, 330, 25))
        Form.setMenuBar(self.menuBar)
        self.retranslateUi(Form)
        QtCore.QMetaObject.connectSlotsByName(Form)

```

```

def retranslateUi(self, Form):
    _translate = QtCore.QCoreApplication.translate
    Form.setWindowTitle(_translate("Form", "Form"))
    self.lineEdit.setPlaceholderText(_translate("Form", " Nama Pengguna"))
    self.lineEdit_2.setPlaceholderText(_translate("Form", " Kata Sandi"))
    self.pushButton.setText(_translate("Form", "L o g   I n"))
    self.label_3.setText(_translate("Form", "Lupa Nama Pengguna atau Kata
Sandi?"))

def show_home_page(self):
    username = self.lineEdit.text()
    password = self.lineEdit_2.text()
    try:
        connection = mysql.connector.connect(
            host="localhost",
            user="root",
            password="",
            database="bankpbo")

        cursor = connection.cursor()
        query = "SELECT * FROM users WHERE username = %s AND user_password
= %s"

        cursor.execute(query, (username, password))
        result = cursor.fetchone()

        if result:
            user_id = result[0]
            self.home_page = HomePage(username, user_id, cursor,
connection)

            self.home_page.show()
        else:
            QtWidgets.QMessageBox.warning(
                self.widget, "Login Gagal",
                "Nama pengguna atau kata sandi tidak valid. Silakan coba
lagi.")

    except mysql.connector.Error as err:
        print("Error MySQL: {}".format(err))
        import traceback
        traceback.print_exc()

if __name__ == "__main__":
    app = QtWidgets.QApplication(sys.argv)
    Form = QtWidgets.QMainWindow()
    ui = Ui_Form()
    ui.setupUi(Form)
    ui.pushButton.clicked.connect(ui.show_home_page)
    Form.setCentralWidget(ui.widget)

```

```
Form.show()  
sys.exit(app.exec_())
```

Alur program ini adalah sebagai berikut

1. **Login Page (Ui_Form):**

- Aplikasi dimulai dengan menampilkan halaman login (**Ui_Form**), yang mencakup elemen seperti **QLineEdit** untuk memasukkan nama pengguna dan kata sandi, serta tombol **QPushButton** untuk masuk.
- Gambar logo dan latar belakang juga ditampilkan pada halaman login.

2. **Login Process:**

- Ketika pengguna mengklik tombol "Login", fungsi **show_home_page** dijalankan.
- Dalam **show_home_page**, program mencoba melakukan koneksi ke database MySQL menggunakan **mysql.connector**.
- Setelah koneksi berhasil, program mengeksekusi query SQL untuk mencocokkan nama pengguna dan kata sandi yang dimasukkan oleh pengguna dengan entri dalam database.
- Jika kombinasi nama pengguna dan kata sandi ditemukan, halaman utama (**HomePage**) ditampilkan.

3. **Home Page (HalamanUtama):**

- Halaman utama (**HomePage**) menampilkan layar utama aplikasi mobile banking setelah login berhasil.
- Ini mencakup elemen seperti gambar logo bank, ucapan selamat datang, dan beberapa tombol untuk fungsi utama seperti cek saldo, transfer, deposit, pembelian pulsa, dompet digital, dan riwayat transaksi.

4. **Menu Transfer:**

- Saat pengguna mengklik tombol "Transfer" di halaman utama, fungsi **tampilkan_menu_transfer** dijalankan.
- Ini membuka dialog transfer (**menu_transfer**) yang meminta informasi pengirim, penerima, dan jumlah transfer.
- Fungsi **transfer_amount** dipanggil untuk mengeksekusi transfer dana antar pengguna.

5. **Transfer Process:**

- Dalam **transfer_amount**, program mencari ID pengirim dan penerima berdasarkan nama pengguna.
- Jumlah transfer dikonversi menjadi integer, dan saldo pengirim diperbarui di database.
- Saldo penerima juga diperbarui, dan rekam transfer dimasukkan ke dalam tabel transaksi.
- Seluruh proses dijalankan dalam satu transaksi database untuk menjaga konsistensi data.

6. **Menu Deposit:**

- Saat pengguna mengklik tombol "Deposit" di halaman utama, fungsi **tampilkan_menu_deposit** dijalankan.
- Ini membuka dialog deposit (**menu_deposit**) yang meminta informasi jumlah deposit.

7. Deposit Process:

- Dalam **deposito_jumlah**, jumlah deposit dikonversi menjadi integer, dan saldo pengguna diperbarui di database.
- Rekam deposit dimasukkan ke dalam tabel deposit untuk tujuan pelacakan.

8. Menu Pulsa dan Dompot Digital:

- Fungsi **tampilkan_menu_pulsa** dan **tampilkan_menu_dompot_digital** masing-masing menangani pembelian pulsa dan top-up dompet digital.
- Informasi seperti nomor HP, jumlah pembelian, dan jenis dompet diminta dari pengguna, kemudian diproses dan diperbarui di database.

9. Menu Riwayat:

- Fungsi **tampilkan_menu_riwayat** menampilkan riwayat transaksi pengguna dalam bentuk tabel menggunakan **QTableWidget**.

10. Saldo:

- Fungsi **tampilkan_saldo** menampilkan dialog yang menampilkan saldo pengguna.

11. Error Handling:

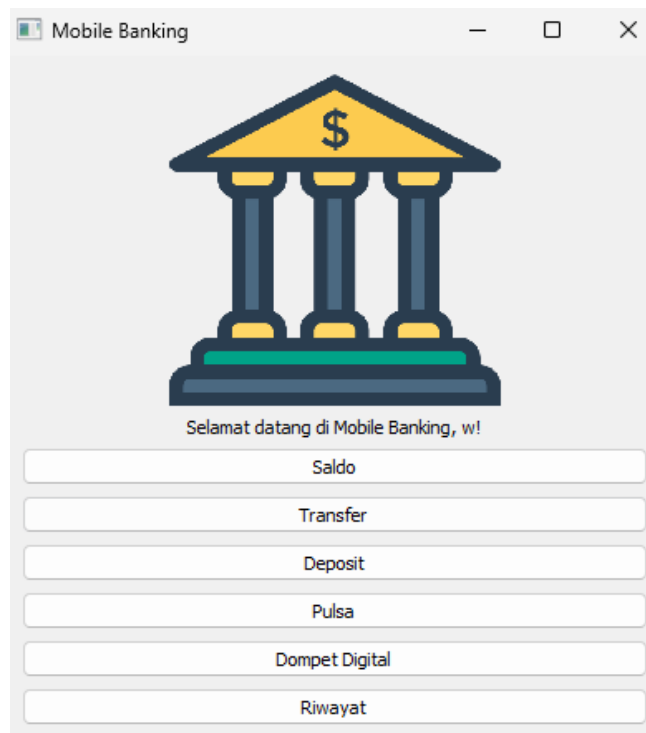
- Terdapat beberapa mekanisme penanganan kesalahan dan pesan peringatan untuk memberi informasi kepada pengguna jika ada masalah selama eksekusi program atau proses transaksi.

Semua fungsi terkait dengan interaksi pengguna dan pemrosesan transaksi telah diimplementasikan di dalam kelas **HalamanUtama**.



1. tampilan pertam adalah menampilkan page log in untuk melakukan transaksi

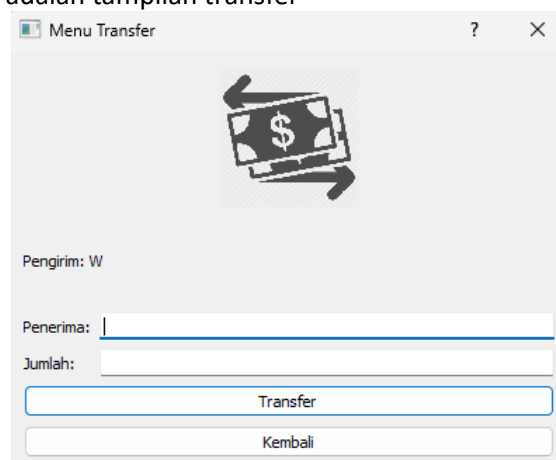
2. tampilan ke dua adalah menu menu yang terdapat pada mobile banking



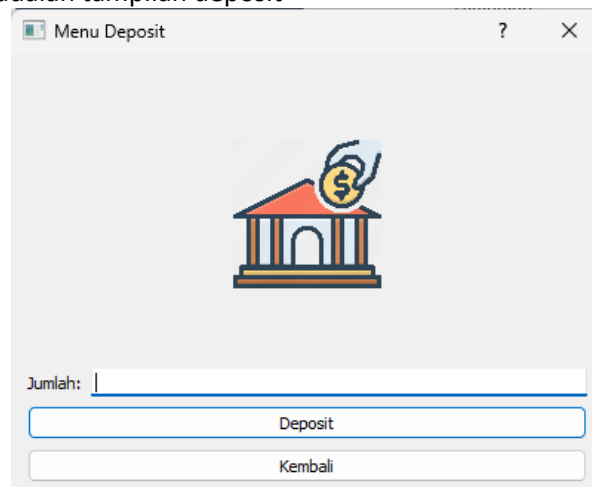
3. tampilan ke tiga menampilkan saldo



4. tampilan ke empat adalah tampilan transfer



5. tampilan ke lima adalah tampilan deposit



Menu Deposit

Jumlah:

Deposit

Kembali

6. tampilan ke enam adalah tampilan pembelian pulsa



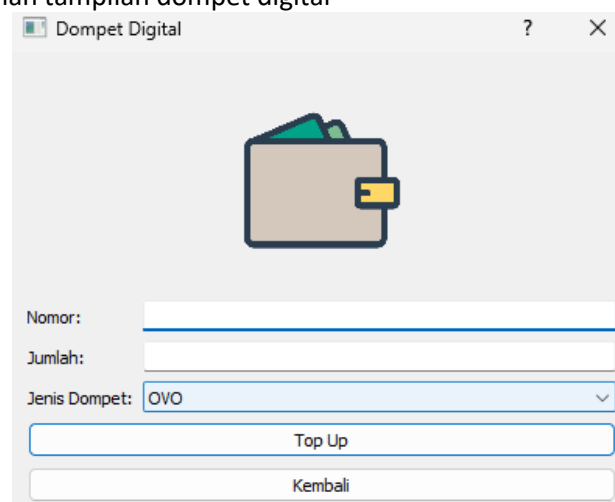
Pulsa Menu

Nomor HP:

Pilihan Pembelian:

Kembali

7. tampilan ke tuju adalah tampilan dompet digital



Dompet Digital

Nomor:

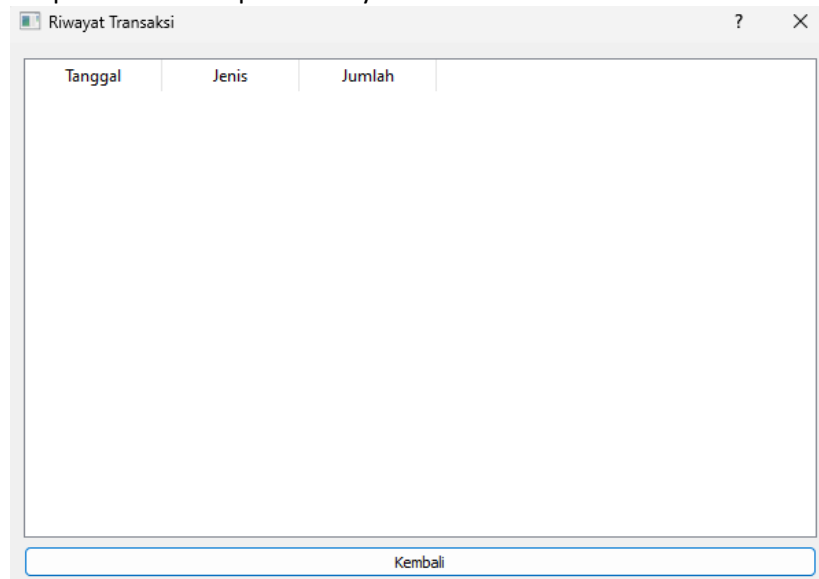
Jumlah:

Jenis Dompet:

Top Up

Kembali

8. tampilan ke delapan adalah tampilan Riwayat



Tanggal	Jenis	Jumlah
---------	-------	--------

Kembali

Daftar Pustaka

1. Sathye, M. (1999). Adoption of Internet banking by Australian consumers: an empirical investigation. *International Journal of Bank Marketing*, 17(7), 324-334.
2. Lee, Y., Lee, J., & Lee, T. (2003). An empirical study on the wireless Internet acceptance model: a focus on motivation and habit. *International Journal of Information Management*, 23(6), 347-356.
3. Liao, Z., & Cheung, M. T. (2002). Internet-based e-banking and consumer attitudes: an empirical study. *Information & Management*, 39(4), 283-295.
4. Agarwal, R., & Prasad, J. (1997). The role of innovation characteristics and perceived voluntariness in the acceptance of information technologies. *Decision Sciences*, 28(3), 557-582.
5. Laukkanen, T. (2007). Internet vs mobile banking: comparing customer value perceptions. *Business Process Management Journal*, 13(6), 788-797.