**LAPORAN PROJECT PEMROGRAMAN VISUAL**

**MONEY TRACKER**

****

**Oleh:**

**I Gede Manik Ariyasa**

**F1D022046**

**PROGRAM STUDI TEKNIK INFORMATIKA FAKULTAS TEKNIK**

**UNIVERSITAS MATARAM**

**Juni 2025**

## Deskripsi Aplikasi

Aplikasi ***Money Tracker*** adalah sebuah alat sederhana yang dirancang untuk membantu pengguna mencatat dan mengelola pemasukan serta pengeluaran harian mereka. Aplikasi ini memungkinkan pengguna untuk memasukkan detail transaksi seperti tanggal, kategori, jumlah, jenis (pemasukan atau pengeluaran), dan catatan. Semua data transaksi disimpan secara lokal menggunakan *database* SQLite dan ditampilkan dalam tabel yang mudah dibaca. Aplikasi ini juga dilengkapi dengan fitur ekspor data ke format CSV, memudahkan pengguna untuk menganalisis data mereka lebih lanjut, serta memberikan antarmuka pengguna yang intuitif dengan sentuhan desain modern.

## Pembuatan Proyek

Proyek Money Tracker ini dibangun menggunakan Python dengan *framework* PyQt5 untuk antarmuka pengguna grafis (GUI) dan SQLite sebagai sistem manajemen database. Berikut adalah langkah-langkah pembuatannya:

1. Desain Antarmuka Pengguna (.ui)

Antarmuka pengguna (UI) aplikasi dirancang menggunakan QtDesigner, sebuah alat grafis untuk membuat *file* .ui. *File* design.ui mendefinisikan elemen-elemen GUI seperti:

1. QMainWindow: Jendela utama aplikasi.
2. QGroupBox: Mengelompokkan *widget* *input* data.
3. QLabel: Label untuk setiap kolom *input* (Date, *Category*, Amount, *Type*, Notes).
4. QDateEdit: Untuk memilih tanggal transaksi.
5. QComboBox: Untuk memilih kategori (*Food*, *Transport*, *Salary*, *Others*) dan jenis transaksi (*Income*, *Expense*).
6. QLineEdit: Untuk memasukkan jumlah transaksi.
7. QTextEdit: Untuk memasukkan catatan tambahan.
8. QPushButton: Tombol "Add" untuk menambahkan transaksi.
9. QTableWidget: Menampilkan daftar transaksi yang telah dicatat.
10. QMenuBar: Berisi menu "File" (dengan "Export CSV") dan "Help" (dengan "About").
11. QStatusBar: Menampilkan pesan status di bagian bawah jendela.

## Implementasi Logika Aplikasi (Python)

Setelah desain UI selesai, file design.ui dimuat ke dalam kode Python. Kelas MoneyTracker (turunan dari QtWidgets.QMainWindow) mengelola semua fungsionalitas aplikasi:

### Inisialisasi:

1. Memuat file design.ui.
2. Membuat koneksi ke database SQLite (money.db) dan memanggil create\_table() untuk memastikan tabel money ada.
3. Menampilkan pesan di status bar.
4. Mengatur tanggal default di dateEdit ke tanggal saat ini.
5. Menghubungkan *event* klik tombol dan aksi menu ke fungsi-fungsi terkait.
6. Menerapkan *styling* menggunakan *stylesheet* Qt untuk memberikan tampilan yang menarik.
7. Mengatur properti QTableWidget seperti warna baris bergantian dan penyesuaian ukuran kolom.
8. Memuat data yang sudah ada dari database ke dalam QTableWidget saat aplikasi dimulai.

### Manajemen Database:

## Fungsi create\_table() membuat tabel money jika belum ada, dengan kolom id, *date*, *category*, *amount*, *type*, dan *notes*.

### Penambahan Entri:

## Fungsi add\_entry() mengambil data dari *widget* *input*, melakukan validasi (memastikan jumlah tidak kosong), dan menyisipkan data baru ke dalam tabel money di database.

## Setelah penambahan, load\_data() dipanggil untuk memperbarui tampilan tabel, dan clear\_form() membersihkan *widget* *input*.

### Pemuatan Data:

## Fungsi load\_data() mengambil semua data dari tabel money, mengurutkannya berdasarkan tanggal, dan menampilkannya di QTableWidget. Data jumlah diformat ke format Rupiah menggunakan format\_rupiah().

### Ekspor CSV:

## Fungsi export\_csv() memungkinkan pengguna memilih lokasi penyimpanan file CSV. Data diambil dari database dan ditulis ke file CSV dengan header kolom yang sesuai.

### Fungsi Pembantu:

## clear\_form(): Mengatur ulang semua widget input ke nilai default.

## show\_about(): Menampilkan kotak pesan dengan informasi tentang aplikasi.

## format\_rupiah(): Mengubah angka menjadi format mata uang Rupiah (misalnya, "Rp. 1.000.000").

### Penanganan Penutupan Aplikasi:

## Fungsi closeEvent() memastikan koneksi database ditutup dengan benar saat aplikasi ditutup.

## Fungsi

## Berikut adalah penjelasan mengenai fungsi-fungsi utama dalam aplikasi Money Tracker:

## 1. \_\_init\_\_(self)

## Fungsi: Metode konstruktor kelas MoneyTracker. Ini adalah titik awal inisialisasi aplikasi.

## Detail: Memuat UI dari file design.ui, menginisialisasi koneksi database SQLite, memanggil create\_table(), mengatur tanggal default, menghubungkan sinyal dari tombol dan menu ke *slot* (fungsi) yang sesuai, menerapkan gaya visual aplikasi, dan memuat data awal ke dalam tabel.

## 2. create\_table(self)

## Fungsi: Membuat tabel database money jika belum ada.

## Detail: Mengeksekusi perintah SQL CREATE TABLE IF NOT EXISTS untuk membuat tabel dengan kolom id, date, category, amount, type, dan notes. id disetel sebagai primary key yang otomatis bertambah (AUTOINCREMENT).

## 3. add\_entry(self)

## Fungsi: Menambahkan entri transaksi baru ke database.

## Detail: Mengambil nilai dari QDateEdit, QComboBox (Category dan Type), QLineEdit (Amount), dan QTextEdit (Notes). Melakukan validasi sederhana untuk memastikan amount tidak kosong. Jika valid, data disisipkan ke tabel money. Setelah penambahan, data di tabel diperbarui (load\_data()) dan formulir input dibersihkan (clear\_form()).

## 4. load\_data(self)

## Fungsi: Memuat semua data transaksi dari database dan menampilkannya di QTableWidget.

## Detail: Mengatur ulang jumlah baris di QTableWidget menjadi nol. Mengeksekusi query SQL SELECT untuk mengambil semua entri dari tabel money, diurutkan berdasarkan tanggal secara menurun. Setiap baris data kemudian disisipkan ke dalam QTableWidget, dengan kolom 'Amount' diformat menjadi Rupiah menggunakan format\_rupiah().

## 5. clear\_form(self)

## Fungsi: Membersihkan semua widget input pada formulir.

## Detail: Mengatur QDateEdit ke tanggal saat ini, mengatur QComboBox ke indeks pertama (default), mengosongkan QLineEdit, dan mengosongkan QTextEdit.

## 6. export\_csv(self)

## Fungsi: Mengekspor semua data transaksi ke file CSV.

## Detail: Membuka dialog QFileDialog untuk memungkinkan pengguna memilih lokasi dan nama file untuk ekspor CSV. Jika pengguna memilih file, data transaksi diambil dari database dan ditulis ke file CSV yang ditentukan, dengan header kolom yang jelas.

## 7. show\_about(self)

## Fungsi: Menampilkan kotak pesan "About" tentang aplikasi.

## Detail: Menggunakan QMessageBox.information untuk menampilkan informasi singkat tentang aplikasi, termasuk nama, fitur, dan pengembang.

## 8. format\_rupiah(self, number)

## Fungsi: Mengubah nilai numerik menjadi format mata uang Rupiah.

## Detail: Menerima sebuah angka (dalam bentuk string atau integer), mencoba mengonversinya menjadi integer, lalu mengembalikan string yang diformat sebagai "Rp. X.XXX.XXX". Menangani ValueError jika input bukan angka.

## 9. closeEvent(self, event)

## Fungsi: Menangani event saat jendela aplikasi ditutup.

## Detail: Memastikan koneksi database SQLite ditutup dengan aman sebelum aplikasi benar-benar keluar, mencegah potensi kerusakan database.

## Dokumentasi

|  |
| --- |
|  |