**Penerapan Event-Driven Programming pada Aplikasi Manajemen Keuangan**

**Pengantar**

Aplikasi manajemen keuangan berbasis C# WinForms yang telah dibuat merupakan contoh nyata penerapan event-driven programming. Paradigma ini menekankan bahwa alur program ditentukan oleh *peristiwa* atau *event* yang terjadi selama aplikasi dijalankan, seperti klik tombol, input data, atau pembukaan form. Dalam aplikasi ini, hampir seluruh interaksi pengguna memicu event-event yang akan menjalankan logika tertentu, seperti perhitungan, navigasi form, atau visualisasi grafik keuangan.

**Konsep Event-Driven Programming**

Event-driven programming adalah suatu pendekatan pemrograman di mana alur eksekusi ditentukan oleh kejadian-kejadian (event) yang dihasilkan dari interaksi pengguna (user interface), sistem, atau input eksternal lainnya. Dalam konteks WinForms, komponen seperti Button, Label, dan Form memiliki event handler yang akan dijalankan ketika event tertentu terjadi—misalnya, event Click, Load, atau TextChanged.

**Penerapan Event-Driven dalam Aplikasi**

Berikut beberapa penerapan event-driven programming dalam aplikasi ini:

1. **Navigasi antar Form dengan Event Click pada Button**

Misalnya:

private void button1\_Click(object sender, EventArgs e)

{

Dashboard dashboard = new Dashboard();

dashboard.Show();

this.Hide();

}

Saat pengguna menekan tombol, event Click dipicu dan akan membuka form Dashboard. Event ini sangat penting untuk mendukung alur kerja aplikasi yang multi-form.

1. **Menampilkan Data Total Keuangan Saat Form Dimuat (Form\_Load)**

private void Dashboard\_Load(object sender, EventArgs e)

{

TampilkanTotalPemasukan();

TampilkanTotalPengeluaran();

TampilkanSisaSaldo();

}

Ketiga metode tersebut memuat data dari database dan langsung ditampilkan pada label tanpa harus menunggu pengguna mengklik elemen UI lain. Ini merupakan contoh event Load, di mana event terjadi ketika form pertama kali dimuat.

1. **Perhitungan dan Penampilan Grafik Otomatis (Live Chart)**

private void LoadChart()

{

// Memuat data pemasukan dan pengeluaran

// Ditampilkan dalam bentuk grafik saat form dibuka

}

Method ini dipanggil pada event Form\_Load, sehingga grafik langsung muncul secara otomatis saat aplikasi berjalan, tanpa intervensi pengguna secara manual.

1. **Menutup Aplikasi dengan Event Click pada Button**

private void button3\_Click(object sender, EventArgs e)

{

Application.Exit();

}

Ketika tombol keluar ditekan, event Click akan men-trigger fungsi untuk mengakhiri proses aplikasi. Ini adalah bentuk implementasi sederhana namun vital dari event-driven control flow.

1. **Update Label Total Saat Diklik (sebelum dioptimasi)**

private void Ltotalpemasukan\_Click(object sender, EventArgs e)

{

// Menampilkan total pemasukan

}

Awalnya, label harus diklik untuk memicu event Click dan menghitung total pemasukan. Namun, setelah dioptimasi, proses ini dipindahkan ke event Form\_Load agar lebih efisien.

**Kesimpulan**

Aplikasi manajemen keuangan ini secara menyeluruh mengimplementasikan prinsip event-driven programming. Penggunaan berbagai event seperti Click, Load, dan Close memungkinkan aplikasi menjadi responsif terhadap aksi pengguna. Setiap komponen UI bertindak sebagai pemicu aksi, memastikan bahwa aplikasi berjalan secara interaktif dan dinamis.

Dengan paradigma ini, logika program tidak hanya mengikuti urutan instruksi, tetapi juga merespons peristiwa yang muncul selama waktu eksekusi, menjadikannya ideal untuk aplikasi berbasis GUI (Graphical User Interface) seperti WinForms.