JURNAL

KONSTRUKSI PERANGKAT LUNAK

MODUL 14

Oleh :

Muhammad Edgar Nadhif

2211104019

SE0601

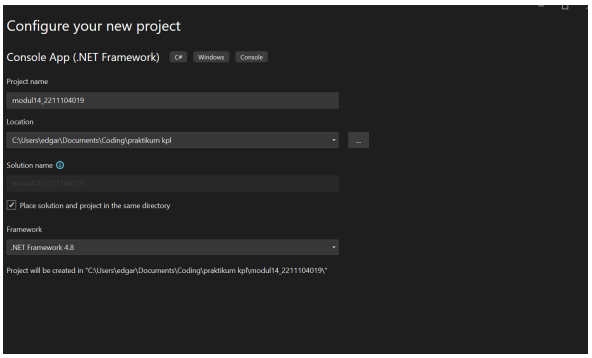
PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

1. Membuat Project Baru



disini saya menggunakan modul 8

2. Refactoring yang Dilakukan
   - Mengikuti standar penamaan PascalCase untuk kelas, metode, dan properti.
   - Menambahkan komentar di bagian penting kode.
   - Menambahkan validasi input menggunakan int.TryParse() agar program tidak crash jika input salah.
   - Menjaga konsistensi indentasi dan whitespace.

3. Kode sebelum dan sesudah

   BankTransferConfig.cs
   Before :

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Text.Json;

6 references
public class BankTransferConfig
{
    2 references
    public string Lang { get; set; }
    4 references
    public Transfer Transfer { get; set; }
    3 references
    public List<string> Methods { get; set; }
    3 references
    public Confirmation Confirmation { get; set; }

    private const string ConfigPath = "bank_transfer_config.json";

    1 reference
    public static BankTransferConfig LoadConfig()
    {
        if (File.Exists(ConfigPath))
        {
            try
            {
                string json = File.ReadAllText(ConfigPath);
                return JsonSerializer.Deserialize<BankTransferConfig>(json);
            }
            catch
            {
                Console.WriteLine("Failed to read config, using default.");
                return DefaultConfig();
            }
        }
        else
        {
```

```
            var defaultConfig = DefaultConfig();
            string json = JsonSerializer.Serialize(defaultConfig, new JsonSerializerOptions { WriteIndented = true });
            File.WriteAllText(ConfigPath, json);
            return defaultConfig;
        }
    }

    2 references
    private static BankTransferConfig DefaultConfig()
    {
        return new BankTransferConfig
        {
            Lang = "en",
            Transfer = new Transfer
            {
                Threshold = 25000000,
                LowFee = 6500,
                HighFee = 15000
            },
            Methods = new List<string> { "RTO (real-time)", "SKN", "RTGS", "BI FAST" },
            Confirmation = new Confirmation
            {
                En = "yes",
                Id = "ya"
            }
        };
    }
}
```

```
2 references
public class Transfer
{
    2 references
    public int Threshold { get; set; }
    2 references
    public int LowFee { get; set; }
    2 references
    public int HighFee { get; set; }
}

2 references
public class Confirmation
{
    2 references
    public string En { get; set; }
    2 references
    public string Id { get; set; }
}
```

after

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Text.Json;

6 references
public class BankTransferConfig
{
    2 references
    public string Lang { get; set; }
    4 references
    public Transfer Transfer { get; set; }
    4 references
    public List<string> Methods { get; set; }
    3 references
    public Confirmation Confirmation { get; set; }

    private const string ConfigPath = "bank_transfer_config.json";

    // Load configuration from file, or create default if not found
    1 reference
    public static BankTransferConfig LoadConfig()
    {
        if (File.Exists(ConfigPath))
        {
            try
            {
                string json = File.ReadAllText(ConfigPath);
                return JsonSerializer.Deserialize<BankTransferConfig>(json);
            }
            catch
            {
                Console.WriteLine("Failed to read config, using default.");
                return DefaultConfig();
            }
        }
```

```
    }

    // Default configuration if file is not found
    2 references
    private static BankTransferConfig DefaultConfig()
    {
        return new BankTransferConfig
        {
            Lang = "en",
            Transfer = new Transfer
            {
                Threshold = 25000000,
                LowFee = 6500,
                HighFee = 15000
            },
            Methods = new List<string> { "RTO (real-time)", "SKN", "RTGS", "BI FAST" },
            Confirmation = new Confirmation
            {
                En = "yes",
                Id = "ya"
            }
        };
    }
}

2 references
public class Transfer
{
    2 references
    public int Threshold { get; set; }
    2 references
    public int LowFee { get; set; }
    2 references
    public int HighFee { get; set; }
}

2 references
public class Confirmation
{
    2 references
    public string En { get; set; }
    2 references
    public string Id { get; set; }
}
```

[Program.cs]

before

```
using System;

0 references
class Program
{
    0 references
    static void Main()
    {
        BankTransferConfig config = BankTransferConfig.LoadConfig();
        string lang = config.Lang;

        // Step 1: Prompt nominal transfer
        Console.WriteLine(lang == "en"
            ? "Please insert the amount of money to transfer:"
            : "Masukkan jumlah uang yang akan di-transfer:");

        int amount = int.Parse(Console.ReadLine());

        // Step 2: Calculate transfer fee
        int fee = amount <= config.Transfer.Threshold ? config.Transfer.LowFee : config.Transfer.HighFee;
        int total = amount + fee;

        // Step 3: Display fee and total
        if (lang == "en")
        {
            Console.WriteLine($"Transfer fee = {fee}");
            Console.WriteLine($"Total amount = {total}");
        }
        else
        {
            Console.WriteLine($"Biaya transfer = {fee}");
            Console.WriteLine($"Total biaya = {total}");
        }
```

```
        // Step 4: Transfer method
        Console.WriteLine(lang == "en" ? "Select transfer method:" : "Pilih metode transfer:");
        for (int i = 0; i < config.Methods.Count; i++)
        {
            Console.WriteLine($"{i + 1}. {config.Methods[i]}");
        }

        int methodChoice = int.Parse(Console.ReadLine());

        // Step 5: Confirmation
        string confirmText = lang == "en" ? config.Confirmation.En : config.Confirmation.Id;
        Console.WriteLine(lang == "en"
            ? $"Please type \"{confirmText}\" to confirm the transaction:"
            : $"Ketik \"{confirmText}\" untuk mengkonfirmasi transaksi:");

        string userInput = Console.ReadLine();

        // Step 6: Final confirmation
        if (userInput == confirmText)
        {
            Console.WriteLine(lang == "en"
                ? "The transfer is completed"
                : "Proses transfer berhasil");
        }
        else
        {
            Console.WriteLine(lang == "en"
                ? "Transfer is cancelled"
                : "Transfer dibatalkan");
        }
    }
}
```

After


```
static void Main()
{
    BankTransferConfig config = BankTransferConfig.LoadConfig();
    string lang = config.Lang;

    // Step 1: Prompt nominal transfer
    Console.WriteLine(lang == "en"
        ? "Please insert the amount of money to transfer:"
        : "Masukkan jumlah uang yang akan di-transfer:");

    // Input validation
    if (!int.TryParse(Console.ReadLine(), out int amount))
    {
        Console.WriteLine(lang == "en" ? "Invalid input!" : "Input tidak valid!");
        return;
    }

    // Step 2: Calculate transfer fee
    int fee = amount <= config.Transfer.Threshold
        ? config.Transfer.LowFee
        : config.Transfer.HighFee;
    int total = amount + fee;

    // Step 3: Display fee and total
    if (lang == "en")
    {
        Console.WriteLine($"Transfer fee = {fee}");
        Console.WriteLine($"Total amount = {total}");
    }
    else
    {
        Console.WriteLine($"Biaya transfer = {fee}");
        Console.WriteLine($"Total biaya = {total}");
    }

    // Step 4: Transfer method
    Console.WriteLine(lang == "en" ? "Select transfer method:" : "Pilih metode transfer:");
    for (int i = 0; i < config.Methods.Count; i++)
    {
        Console.WriteLine($"{i + 1}. {config.Methods[i]}");
    }

    // Step 5: Input transfer method
    if (!int.TryParse(Console.ReadLine(), out int methodChoice) || methodChoice < 1 || methodChoice > config.Methods.Count)
    {
        Console.WriteLine(lang == "en" ? "Invalid choice!" : "Pilihan tidak valid!");
        return;
    }

    // Step 6: Confirmation
    string confirmText = lang == "en" ? config.Confirmation.En : config.Confirmation.Id;
    Console.WriteLine(lang == "en"
        ? $"Please type \"{confirmText}\" to confirm the transaction:"
        : $"Ketik \"{confirmText}\" untuk mengkonfirmasi transaksi:");

    string userInput = Console.ReadLine();

    if (userInput == confirmText)
    {
        Console.WriteLine(lang == "en"
            ? "The transfer is completed"
            : "Proses transfer berhasil");
    }
    else
    {
        Console.WriteLine(lang == "en"
            ? "Transfer is cancelled"
            : "Transfer dibatalkan");
    }
}
}
```

(diperbaiki dan ditambahkan validasi + komentar)
Perbaikan kode ini digunakan untuk memastikan bahwa input yang dimasukkan pengguna berupa angka. Jika tidak valid, program akan menampilkan pesan kesalahan dan menghentikan eksekusi.