**Raport z przeprowadzonych prac nad projektem   
z zajęć Programowania obiektowego**

**Tytuł projektu:** Budget Buddy

**Skład projektu:** Filip Marczyk, Patryk Kotulski

**Grupa:** 4

**Opis aplikacji:**

Projekt Budget Buddy jest aplikacją do zarządzania budżetem osobistym, napisaną   
w języku C#, przy użyciu technologii Windows Forms. Aplikacja umożliwia użytkownikowi śledzenie wydatków i przychodów, ustawianie limitów budżetowych na różne okresy (dzień, tydzień, miesiąc, rok) oraz generowanie raportów z transakcjami w formie pliku tekstowego.

**Implementacja:**

Projekt składa się z kilku głównych formularzy:

**Login**: Formularz logowania, gdzie użytkownik wprowadza swoje imię i budżet początkowy.

**Main**: Główny interfejs użytkownika po zalogowaniu, wyświetlający bieżący stan budżetu oraz opcje nawigacji do innych funkcji.

**Wydatki**: Formularz do dodawania wydatków, z funkcjonalnością sprawdzania przekroczenia limitów.

**Przychody**: Formularz do dodawania przychodów.

**Limity**: Formularz do ustawiania limitów na różne okresy czasu.

Każdy z formularzy jest odpowiedzialny za zarządzanie swoim stanem oraz komunikację z głównym interfejsem (Main), który koordynuje całość aplikacji.

**Paradygmaty programowania obiektowego:**

W projekcie Budget Buddy wykorzystano kilka kluczowych paradygmatów programowania obiektowego, które pozwoliły na uzyskanie modularnej, skalowalnej i łatwej w utrzymaniu aplikacji. Oto najważniejsze z nich:

1. **Dziedziczenie**

Klasa BaseForm jest klasą bazową dla innych formularzy takich jak Login, Main, Wydatki, Przychody, i Limity. Dzięki temu możliwe było dziedziczenie wspólnych funkcji i zmiennych, co zredukowało powtarzalność kodu i uprościło zarządzanie stanem aplikacji. Na przykład, wszystkie formularze mają dostęp do metod zarządzających danymi użytkownika, takich jak SaveUserData czy LoadUserData zdefiniowanych w klasie BaseForm.

Przykład:

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie

1. **Hermetyzacja**

Hermetyzacja jest realizowana poprzez prywatne zmienne i metody dostępne jedynie wewnątrz klasy oraz publiczne metody dostępne z zewnątrz do interakcji z danymi. Przykładowo, dane użytkownika (UserData) są dostępne poprzez właściwości (get i set), co umożliwia kontrolowanie dostępu do tych danych.

Przykład:

Obraz zawierający tekst, zrzut ekranu, Czcionka, oprogramowanie

Opis wygenerowany automatycznie

1. **Polimorfizm**

Polimorfizm pozwala na definiowanie metod, które mogą działać w różny sposób w zależności od kontekstu. W projekcie Budget Buddy formularze dziedziczące po BaseForm mogą implementować swoje specyficzne wersje metod takich jak SaveUserData czy NavigateToMain, dostosowując ich działanie do specyficznych potrzeb.

Przykład:

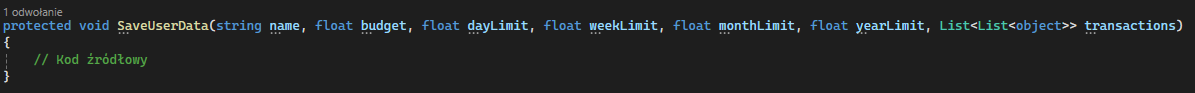
Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

1. **Abstrakcja**

Abstrakcja jest realizowana poprzez tworzenie klas i metod, które ukrywają złożoność i szczegóły implementacyjne, ujawniając jedynie niezbędne funkcje. Klasy takie jak UserData czy BaseForm zapewniają interfejsy do interakcji z danymi użytkownika i operacjami budżetowymi, ukrywając szczegóły implementacji tych operacji.

Przykład:



**Podsumowanie i wnioski:**

Projekt udostępnia użytkownikowi prosty, intuicyjny interfejs do zarządzania budżetem osobistym. Wykorzystanie Windows Forms pozwala na szybkie tworzenie aplikacji desktopowych w C#, co jest idealne do małych projektów.

Podczas implementacji napotkano pewne wyzwania, takie jak zarządzanie danymi transakcyjnymi i ich walidacja, jednak zastosowanie interfejsów i dziedziczenia pozwoliło na efektywne zarządzanie różnymi typami transakcji (wydatki, przychody).

**Załącznik:** Link do repozytorium na GitHub: <https://github.com/kotulski/budget-buddy>