**Dziennik Wydatków**

Projekt na zaliczenie przedmiotu Programowanie 3 – zaawansowane

Dokumentacja projektu (wersja 3.0)

**Autor: Maja Trębacz**

**Spis treści**

[1. Informacje ogólne 2](#_Toc471576100)

[2. Specyfikacja wymagań funkcjonalnych 2](#_Toc471576101)

[3. Komunikacja z użytkownikiem 3](#_Toc471576102)

[4. Przechowywanie danych 4](#_Toc471576103)

[5. Struktura programu - klasy 5](#_Toc471576104)

[1. Expense 5](#_Toc471576105)

[2. ExpenseTracker 5](#_Toc471576106)

[3. ExpenseTrackerReports 6](#_Toc471576107)

[4. User 6](#_Toc471576108)

[5. Users 6](#_Toc471576109)

[6. Program 6](#_Toc471576110)

[6. Diagram klas 7](#_Toc471576111)

# Informacje ogólne

Celem niniejszego projektu jest zrealizowanie programu służącego jako rejestr personalnych wydatków oraz przychodów.

Program ma za zadanie umożliwić dodawanie wpisów o dokonanych zakupach (tytuł, data, kwota, kategoria) oraz dochodach, ich późniejszej edycji, a także generowanie zestawień dotyczących wydatków. Przychód jest technicznie traktowany jako specjalna kategoria wydatku.

Będzie to aplikacja konsolowa, a komunikacja z użytkownikiem będzie się odbywała za pomocą wyznaczonych komend.

# Specyfikacja wymagań funkcjonalnych

Operacje możliwe do wykonania przez użytkownika programu można podzielić na 2 typy: edycji i raportowania.

**Edycja**:

* Dodawanie wpisu
* Wybór wpisu do edycji oraz zmiana jednego z jego atrybutów
* Usunięcie wpisu

**Raportowanie:**

* Raport ogólny: podanie sumy wydatków, sumy przychodów oraz bilansu i wylistowanie wpisów na podstawie daty początkowej i końcowej (możliwość zmiany sortowania: po dacie, po kwocie, po tytule wydatku)
* Raport kategorii: wylistowanie wpisów z podanej kategorii i zakresu dat, podanie ich sumy (możliwość zmiany sortowania: po dacie, po kwocie, po tytule wydatku)
* Raport największych wydatków: lista największych n wydatków w podanym zakresie czasowym (n podawane przez użytkownika)
* Raport struktury wydatków: prezentuje sumę wydatków, ich ilość oraz procentowy udział każdej kategorii w budżecie w podanym zakresie czasowym

# Komunikacja z użytkownikiem

Na poszczególnych etapach dostępne będzie menu w których zostanie podana lista dostępnych akcji oraz odpowiadające im komendy. Użytkownik będzie proszony o podawanie odpowiednich danych. Aplikacja ma za zadanie sprawdzanie ich poprawności oraz wykonywanie odpowiednich akcji.

W celu poprawnego wyświetlania raportów okno powinno mieć szerokość mieszczącą co najmniej 120 znaków.

**Ekran początkowy:**

Możliwe wybory:

1 - Zaloguj

2 - Nowe konto

0 – Zakończ

**Ekran logowania:**

Użytkownik jest proszony o podanie loginu i hasła. W przypadku poprawnego hasła zostanie zalogowany. Po 3 nieudanych próbach podania hasła nastąpi powrót do menu.

Wielkość znaków w nazwie użytkownika i haśle ma znaczenie.

**Ekran dodawania konta:**

Użytkownik jest proszony o podanie loginu i hasła. Jeśli użytkownik o podanej nazwie jeszcze nie istnieje to na podstawie otrzymanych danych zostanie utworzone nowe konto oraz nowy dziennik i nastąpi przejście do ekranu po zalogowaniu.

**Ekran po zalogowaniu:**

Możliwe wybory:

1 - Dodaj wpis

2 - Edytuj / usuń wpis

3 - Generuj raport

4 - Zmiana hasła

0 – Wyloguj

**Po wyborze opcji dodaj wpis:**

Program poprosi o podanie daty, kwoty, kategorii, tytułu i ewentualnego komentarza. W przypadku potwierdzenia poprawności danych doda wpis i powróci do ekranu początkowego. W przeciwnym wypadku poprosi o ponowne wprowadzenie danych.

**Po wyborze opcji edycji:**

Program poprosi o podanie zakresu dat w którym jest wpis pożądany do edycji. W przypadku podania poprawnego zakresu wyświetli listę wydatków wraz z numerami i poprosi o podanie numeru wpisu do edycji. Numery będą indeksami pod którymi zapisane są wpisy, zatem ich kolejność niekoniecznie będzie rosnąca.

Użytkownik następnie powinien wybrać między usunięciem, a edycją. W przypadku wyboru usunięcia podany wpis zostanie skasowany. Jeśli zostanie wybrana edycja program poprosi o nr kolumny do edycji oraz nową wartość. W przypadku potwierdzenia poprawności danych wykona akcję i powróci do ekranu głównego menu. W przeciwnym wypadku poprosi o ponowne wprowadzenie danych.

**Po wyborze opcji raportu:**

Program wylistuje typy raportów i poprosi o wybór jednego.

W przypadku wyboru:

* Raport ogólny: oczekiwane będzie podanie zakresu oraz ewentualnie typu sortowania (domyślnie po dacie)
* Raport kategorii: oczekiwane będzie podanie numeru odpowiadającego kategorii, zakresu oraz ewentualnie typu sortowania (domyślnie po dacie)
* Raport największych wydatków: oczekiwane będzie podanie zakresu dat oraz ilości wpisów do wylistowania
* Raport struktury wydatków: oczekiwane będzie podanie zakresu dat

Po otrzymaniu poprawnych danych program wyświetli raport.

**Po wyborze opcji zmiany hasła:**

Użytkownik zostanie poproszony o podanie starego oraz nowego hasła. Jeśli stare hasło jest poprawne nastąpi jego zmiana. Po 3 nieudanych próbach podania hasła nastąpi powrót do menu.

# Przechowywanie danych

Pamiętanie dane to lista obiektów klasy Expense t.j. informacje o wydatku takie jak: Data transakcji, Tytuł, Kwota, Kategoria, Komentarz.

Zapis oraz odczyt danych odbywać się będzie za pomocą mechanizmu serializacji. Przechowywane będą w pliku typu XML, przy użyciu DataContractSerializer, o nazwie odpowiadającej nazwie użytkownika. Na początku działania programu dane będą pobierane i deserializowane z pliku. Po każdej edycji odbywać się będzie serializacja i ponowny zapis danych.

Dodatkowo w oddzielnym pliku typu pamiętana będzie lista użytkowników wraz z ich hasłami.

Lokalizacją plików jest folder bieżący działania programu.

# Struktura programu - klasy

## Expense

Struktura, która zawiera informacje o wydatku.

Zawiera pola:

* Data transakcji – TransactionDate – typ DateTime
* Kwota transakcji – Amount – typ decimal
* Kategoria - Category – typ wyliczeniowy Categories (zdefiniowany w tym samym pliku)
* Tytuł wpisu - Title – string
* Komentarz do wpisu – Note - string

Typ wyliczeniowy Categories definiuje następujące typy wpisów:

* Jedzenie (=1)
* Dom (=2)
* Rozrywka (=3)
* Okazjonalne (=4)
* Transport (=5)
* Inne (=6)
* Przychód (=-1)

Wpisy z kategorii przychód powinny mieć kwoty o wartościach dodatnich, pozostałe (wydatki) ujemnych. Zatem kwota wpisu powinna być tego samego znaku co reprezentacja liczbowa jego kategorii. W przypadku próby wprowadzenia danych niespełniających tej właściwości wyrzucany będzie wyjątek *AmountCategoryIncosistencyException.*

## ExpenseTracker

Zawiera listę wydatków (obiektów typu Expense) oraz obsługuje dodawanie nowych wpisów, usuwanie oraz modyfikację istniejących na podstawie podanego indeksu wpisu. Obsługuje serializację i deserializację danych.

## ExpenseTrackerReports

Rozszerza klasę ExpenseTracker o metody generujące raporty przy użyciu technologii LINQ. Wprowadzony został typ wyliczeniowy Sorting używany do ustalenia kolejności wyświetlanych w raportach wpisów.

## Typ wyliczeniowy Sorting zawiera:

* Data\_malejaco (=-1),
* Data\_rosnaco (=1)
* Kwota\_malejaco (=-2)
* Kwota\_rosnaco (= 2)
* Tytul\_malejaco (= -3)
* Tytul\_rosnaco (= 3)

Na jego podstawie poprzez metodę ReportOrdering generowany jest komparator typów Expense używany do sortowania wpisów w raportach.

Metody generujące raporty:

* generateGeneralReport – raport ogólny
* generateCategoryReport – raport kategorii
* generateMaxExpensesReport – raport największych wydatków
* generateStructuralReport – raport struktury wydatków

## User

Przechowuje nazwę użytkownika (będącą unikalną wartością) oraz hasło. Zawiera także referencję do ExpenseTracker odpowiadającemu danemu użytkownikowi (to pole nie jest serializowane lecz ładowane na podstawie nazwy użytkownika).

## Users

Przechowuje listę użytkowników. Umożliwia sprawdzenie czy użytkownik istnieje, czy jego hasło jest poprawne oraz dodanie użytkownika. Obsługuje serializację i deserializację danych.

## Program

Odpowiada za interakcję z użytkownikiem, generowanie menu wyboru, nasłuchiwanie jego komend, kontrole wprowadzanych danych oraz uruchamianie odpowiednich metod.

# Diagram klas

Diagram prezentuje klasy, ich pola oraz metody.

