Politechnika Świętokrzyska	
Nazwa Przedmiotu: Języki Skryptowe	Prowadzący: dr inż. Dariusz Michalski
Grupa: Piotr Strzępek 3ID15B	Data oddania: 25.06.2025
System do monitorowania poziomu nawodnienia	

2. Opis projektu

Cel projektu

Celem projektu jest analiza dziennego spożycia wody przez dwóch użytkowników na podstawie danych z pliku CSV, obliczenie średniego zużycia i zapisanie wyników do osobnych plików tekstowych. Dzięki temu można w prosty sposób monitorować i porównywać zwyczaje nawodnienia.

Funkcje aplikacji

Aplikacja umożliwia:

1. Dodawanie użytkowników
2. Aktualizację nazw użytkowników
3. Dodawanie dziennych rekordów spożycia wody
4. Obliczanie i wyświetlanie średniego spożycia
5. Usuwanie użytkowników
6. Wczytywanie/zapisywanie danych do pliku CSV
7. Wyświetlanie danych wszystkich użytkowników

Zakres funkcjonalny (co zostało zaimplementowane)

- Interfejs tekstowy (menu w konsoli)
- Obsługa pliku CSV (wczytywanie i zapisywanie danych)
- Walidacja danych wejściowych (np. brak wartości ujemnych)
- Obsługa wyjątków (np. błędy konwersji, brak pliku)
- Obsługa wielu użytkowników i ich rekordów

3. Struktura projektu

- **User** – klasa reprezentująca pojedynczego użytkownika i jego dane
- **ConsumptionAnalyzer** – klasa zarządzająca wieloma użytkownikami i danymi
- **main()** – główna pętla programu z interfejsem tekstowym
- **Plik CSV** – struktura: user_id,name,amount

4. Technologie i biblioteki

- Python 3.9

- Wbudowane moduły:

    csv – czytanie plików CSV

    os – sprawdzanie istnienia katalogów i ścieżek

    sys – odczyt argumentów z linii poleceń

5. Sposób działania programu

1. Przy uruchomieniu program próbuje wczytać dane z pliku CSV.
2. Użytkownik porusza się po menu, wybierając numer opcji.
3. Zmiany mogą być zapisane do pliku (lub porzucone).
4. Program umożliwia zarządzanie danymi użytkowników lokalnie w sesji.

6. Przykłady kodu (z wyjaśnieniem)

Przykład 1: Dodanie rekordu i walidacja

```
def add_record(self, amount):
    value = float(amount)
    if value < 0:
        raise ValueError("Wartość spożycia nie może być ujemna")
    self.records.append(value)
```

Funkcja waliduje dane – nie pozwala na wartości ujemne.

Przykład 2: Wczytywanie danych z CSV

```
with open(self.filepath, newline='', encoding='utf-8') as csvfile:
    reader = csv.DictReader(csvfile)
    for row in reader:
        ...
```

Dzięki DictReader można łatwo odczytywać dane na podstawie nazw kolumn.

7. Testowanie

- **Testowanie manualne** poprzez wybór opcji z menu
- Testowane przypadki:

    - Błędny format liczby (np. litery zamiast cyfr)
    - Nieistniejący plik CSV
    - Puste wartości
    - Wprowadzanie zduplikowanego ID
- Aplikacja poprawnie komunikuje błędy i informuje użytkownika

8. Wnioski

Co się udało?

- Stabilna obsługa danych użytkowników
- Wczytywanie i zapisywanie danych z pliku
- Wyraźne komunikaty i struktura menu
- Prosta, ale solidna architektura obiektowa

Co można było zrobić lepiej?

- Walidacja unikalności nazw (obecnie tylko ID)
- Obsługa dat i dokładnych pomiarów (np. z podziałem na dni)

Jakie kompetencje zostały rozwinięte?

- Praca z plikami CSV
- Obsługa wyjątków
- Programowanie obiektowe w Pythonie
- Projektowanie konsolowego interfejsu użytkownika
- Refaktoryzacja i czytelność kodu