Tomasz Królikowski

Nr albumu: 153790

Zadanie znajduję się w repozytorium GIT pod adresem:
https://github.com/krolikowski80/studia_WSB/tree/main/Python/intro/zad_2

Lab 2 – Testowanie aplikacji: Test-Driven Development (TDD)

1. Cel zadania

- Zapoznanie się z podstawami testowania aplikacji w Pythonie.

- Praktyka podejścia Test-Driven Development (TDD).

- Implementacja testów jednostkowych przy użyciu `unittest`.

- Poprawa jakości kodu dzięki testom.

- Sprawdzenie pokrycia kodu testami.

2. Struktura projektu

```
Lab2/
├── app.py                  # Zawiera 5 funkcji do przetestowania
├── test_app.py             # Zawiera testy jednostkowe
├── README_TDD_Lab2.md      # Dokumentacja zadania
└── htmlcov/                # wygenerowano raport HTML z coverage - wysyłanie tego do repo jest antywzorcem, ale dla celów zadania zostało wysłane. ;)
```

3. Funkcje zaimplementowane w app.py

1. is_valid_email(email) – sprawdza poprawność adresu e-mail (regex).

2. is_palindrome(text) – sprawdza, czy dany tekst to palindrom (ignorując wielkość liter i znaki).

3. rectangle_area(width, height) – zwraca pole prostokąta lub błąd przy ujemnych wymiarach.

4. filter_even_numbers(numbers) – zwraca listę liczb parzystych.

5. convert_date_format(date_str) – konwertuje datę z "YYYY-MM-DD" na "DD.MM.YYYY".

4. Testy jednostkowe (test_app.py)

- Każda funkcja ma minimum 3 testy:

  • Przypadki typowe

  • Przypadki brzegowe

  • Przypadki błędne (jeśli mają sens)

Testy są zorganizowane w klasie dziedziczącej po unittest.TestCase.

Aby uruchomić testy: python test_app.py

5. Pokrycie kodu testami

Instalacja:

pip install coverage

Uruchomienie testów z pokryciem:

coverage run -m unittest test_app.py

Wyświetlenie raportu:

coverage report

Raport HTML (opcjonalnie):

coverage html

6. Wymagania

Projekt działa w standardowym Pythonie (>= 3.7) u mnie 3.13.2

- unittest (wbudowany)

- re, datetime (wbudowane)

- coverage (do analizy pokrycia)