Raport z projektu "Program do Obliczania Różnic Kursowych"

Wstęp

Projekt "Program do Obliczania Różnic Kursowych" miał na celu stworzenie prostego programu do dodawania, spłacania i wyświetlania faktur, wykorzystującego kursy walut z API Narodowego Banku Polskiego (NBP) do przeliczania kwot między różnymi walutami.

Proces tworzenia

1. **Analiza wymagań**: Na początku projektu przeprowadziłem analizę wymagań, aby zrozumieć, co jest potrzebne do stworzenia funkcjonalności programu. Zidentyfikowałem trzy główne funkcje: dodawanie faktury, spłacanie faktury i wyświetlanie faktur.
2. **Projektowanie struktury**: Następnie zaprojektowałem strukturę programu, decydując się na wykorzystanie modułu `requests` do komunikacji z API NBP oraz modułu `datetime` do obsługi dat.
3. **Implementacja**: Przystąpiłem do implementacji poszczególnych funkcji, zaczynając od dodawania faktury, a następnie przechodząc do spłacania faktury i wyświetlania faktur.
4. **Testowanie**: Po zaimplementowaniu każdej funkcji przeprowadziłem testy, aby upewnić się, że działa ona poprawnie i zgodnie z moimi oczekiwaniami.

Napotkane problemy i rozwiązania

1. **Komunikacja z API NBP**: Pierwszym wyzwaniem było skuteczne pobieranie kursów walut z API NBP. Rozwiązałem ten problem poprzez wykorzystanie biblioteki `requests` do wysyłania zapytań HTTP i pobierania danych w formacie JSON.
2. **Obsługa błędów**: Kolejnym wyzwaniem było zapewnienie odpowiedniej obsługi błędów, takich jak błędy związane z komunikacją z API lub błędy użytkownika podczas wprowadzania danych. Zaimplementowałem obsługę wyjątków przy użyciu konstrukcji `try...except`, aby program mógł reagować na błędy i odpowiednio informować użytkownika.
3. **Aktualizacja kwoty faktury**: Podczas spłacania faktury, kwota faktury w pliku nie była aktualizowana, co powodowało nieprawidłowe wyświetlanie pozostałej kwoty do spłaty. Rozwiązałem ten problem poprzez dodanie funkcji aktualizującej kwotę faktury w pliku po spłaceniu częściowej lub całkowitej kwoty.

Wnioski

Projekt pokazał mi, jak istotne jest zapewnienie odpowiedniej obsługi błędów w aplikacji. Poprawne reagowanie na błędy użytkownika oraz błędy związane z komunikacją z zewnętrznymi serwisami może znacząco poprawić doświadczenie użytkownika i stabilność programu.