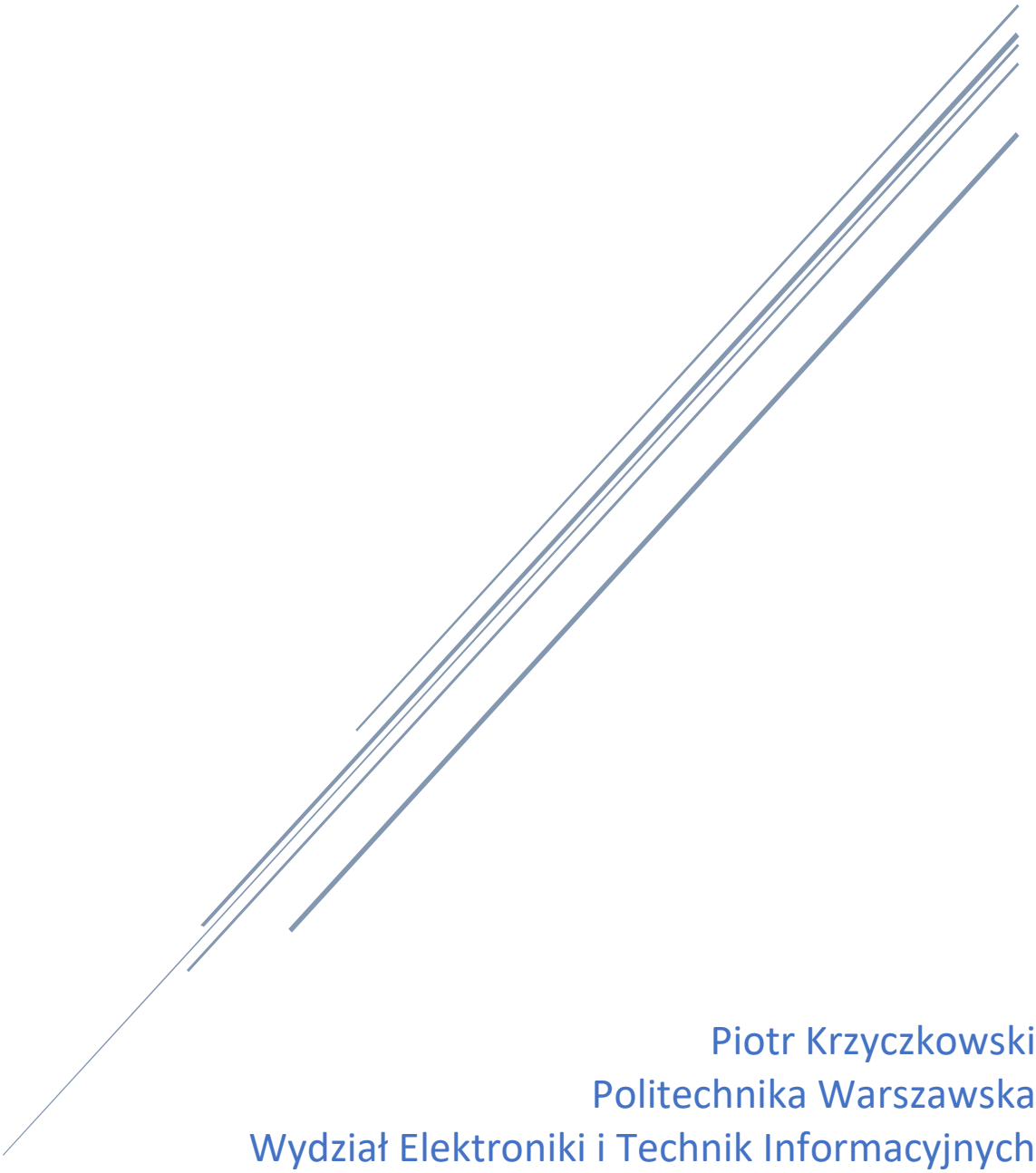
PROGRAM DO OBSŁUGI PŁYWALNI



Piotr Krzyczkowski
Politechnika Warszawska
Wydział Elektroniki i Technik Informacyjnych
2022

1. Opis projektu

Program służy do obsługi pływalni. Pomaga on w rezerwacji miejsc, zakupie biletów oraz generowaniu raportów finansowych. Rezerwacja biletów może odbywać się maksymalnie z sześciodniowym wyprzedzeniem. Zapewnia dodawanie klientów indywidualnych oraz szkółek pływackich. Dodatkowo jest dostosowany pod szeroką gamę obiektów. Umożliwia zmianę nazwy, liczby torów, wgranie planu pracy oraz zmianę cennika.

2. Struktura programu

a) attributes

Folder zawiera pliki odpowiedzialne za pracę z atrybutami pływalni. W jego skład wchodzi:

- *swimmingpool_attributes.txt* Przechowuję nazwę oraz liczbę torów pływalni.

- *name_tracks_data.py* Posiada klasę Attributes, która jest odpowiedzialna za zmianę nazwy oraz liczby torów oraz przywoływanie istniejących danych z pliku txt.

b) prices

Folder zawiera pliki związane z generowaniem, przechowywaniem oraz odwoływaniem się do cennika pływalni. W jego skład wchodzi:

- *new_prices.json* Pokazuje strukturę pliku, za pośrednictwem którego jest generowany cennik.

- *prices_data.py* Posiada klasę DataPrices odpowiedzialną za tworzenie cennika oraz umożliwiąca odwoływanie się do poszczególnych jego elementów.

- *prices.json* Przechowuje obecny cennik. Składający się z 7 rodzajów klientów oraz godziny przed którą obowiązuje promocja.

c) reports

Folder zawiera pliki związane z generowaniem codziennego raportu. Generują się w nim codzienne raporty. W jego skład wchodzi:

- *daily_report.py* Posiada klasę Report, która jest odpowiedzialna za tworzenie raportów.

- *report.txt* Przechowuje dzienne dane związane z klientami oraz finansami.

d) tests

Folder zawiera testy jednostkowe na poszczególnych plikach.

e) work_schedule

Folder zawiera pliki odpowiedzialne za tworzenie oraz przechowywanie danych związanych z planem pracy obiektu. W jego skład wchodzi:

- *work_schedule_data.py* Posiada klasę DataWorkSchedule odpowiedzialną za wgrywanie nowego planu pracy, odwoływanie się do pliku z obecny planem pracy oraz jego modyfikację.

- *work_schedule.json* Przechowuje obecny plan pracy obiektu.

- *work_schedule.txt* Plik wzorcowy. Na jego podstawie tworzony jest plan pracy obiektu. Pokazuje jak powinny być zapisane dane, żeby wygenerować poprawy plik.

f) report_generator.py

Odpowiada za proces generacji dziennego raportu.

g) swimmingpool.py

Posiada klasę Swimmingpool odpowiedzialną za dodawanie klientów indywidualnych oraz szkółek pływackich do pływalni, sprawdzanie miejsca oraz zwracanie wolnych terminów.

h) time_functions.py

Zawiera funkcję operujące na czasie oraz wejściach z nim związanych.

i) text_user_interface.py

Zawiera interfejs tekstowy umożliwiający komunikowanie się użytkownika końcowego z programem.

j) config.json

Plik konfiguracyjny

3. Instrukcja użytkowania

W celu uruchomienia programu należy włączyć plik text_user_interface.py

Po włączeniu programu widzimy ekran startowy. U góry ekranu znajduję się nazwa basenu oraz liczba torów. W prawym górnym rogu widzimy obecny czas. Z prawej strony umieszczony jest ekran pomocniczy. W celu wykonania operacji trzeba postępować zgodnie z komunikatami wyświetlanymi na niebieskim ekranie (ekran funkcyjny). W każdej chwili możemy wprowadzić q, którego wciśnięcie spowoduje powrót do ekranu głównego lub wyłączenie programu w przypadku kiedy się już w nim znajdujemy.

Podczas korzystania z programu natrafimy na dwa rodzaje wprowadzania danych:

- Wciśnięcia

  Umożliwiają nam wybranie danej opcji poprzez wciśnięcie danego przycisku na klawiaturze. Automatycznie po wciśnięciu następuje przekierowanie do następnej funkcji.

- Wiersze

  Wprowadzamy w nich ciągi znaków, które następnie musimy zaakceptować wciskając przycisk „ENTER”

W przypadku pojawienia się jakiegoś komunikatu należy wcisnąć dowolny przycisk aby przejść dalej.

Po rozpoczęciu działania ekran funkcyjny daje nam 4 możliwości:

- Menu
- Add People
- Add Swimming School
- Space and tracks preview

a) Menu

Menu pozwala nam na:

- Zmianę nazwy pływalni. Po jej wybraniu należy wprowadzić nową nazwę.
- Zmianę liczby torów. Należ podać nową liczbę torów.
- Zmianę planu pracy pływalni. Należy podać ścieżkę nowego pliku. Plik powinien być sformatowany zgodnie z plikiem wzorowym work_schedule.txt
- Zmianę cennika. Należy podać ścieżkę nowego pliku. Plik powinien być sformatowany zgodnie z plikiem wzorowym new_prices.json

b) Add People

Pozwala nam dodawanie klientów indywidualnych.

Po wybraniu należy postępować zgodnie z poleceniami: wybrać dzień, wybrać kto jest klientem, wprowadzić liczbę osób, podać godzinę wejścia, wprowadzić liczbę godzin na którą klient chce zakupić bilet, zaakceptować lub odrzucić koszt biletu.

W przypadku braku miejsca wyświetli się nam najbliższy wolny termin.

Po wybraniu dnia tygodnia na ekranie pomocniczym wyświetlają się godziny otwarcia obiektu wybranego dnia.

c) Add Swimming School

Pozwala nam dodawanie szkółek pływackich.

Działa podobnie jak „Add People" z tym, że klientem automatycznie jest szkółka pływacka i zamiast liczby osób podajemy liczbę torów. Maksymalna liczba torów stanowi 35% ich całkowitej liczby.

W celu generowania dziennych raportów należy zainstalować crontaba na swoim urządzeniu. Pozwoli nam to na cykliczne uruchamianie pliku. Pod linkiem znajduję się instrukcja jak go używać:

https://pypi.org/project/python-crontab/

Należy ustawić godzinę wykonywania zadania na późniejszą niż godzina zamknięcia basenu.

d) Space and tracks preview

Pozwala zobaczyć nam podgląd miejsca na basenie w określonym dniu.

4. Część refleksyjna

Po otrzymaniu projektu nie wiedziałem do końca jak on ma wyglądać i jak powinien działać. Pierwsze konsultacje rozjaśniły mi lekko ten obraz. Wiedziałem co powinienem zrobić i czego użyć. Pojawiało się w mojej głowie tylko jedno pytanie „jak?"

Wraz z upływem czasu coraz bardziej rozjaśniał mi się wygląd programu. Niestety zacząłem jego tworzenie od złej strony. Zrobiłem złe założenia, których trzymałem się prawie do samego końca przez co sam utrudniłem sobie wykonanie zadania.

Udało mi się zrobić większość rzeczy, które zaplanowałem. Zaimplementowałem funkcję odpowiedzialne za ceny, atrybuty, plan pracy, raporty oraz funkcję pływalni które drastycznie zmieniłem. Ciężko wybrać mi tylko jedną rzecz, którą mógłbym poprawić. Na pewno mógłbym zmienić koncepcję planu pracy, przez co funkcjonowanie programu mogło by być bardziej wszechstronne. Z pewnością mógłbym wykonać też GUI zamiast TUI, dzięki czemu mój program nie wyglądał by na przestarzały. Nie mniej jednak sądzę, że mój projekt jest dobrze wykonany i spełnia swoje założenia. Udało mi się wprowadzić do niego sporo możliwości. Najbardziej jestem zadowolony z faktu, że wszystko działa jak baza danych. Czyli po wprowadzeniu jakichś zmian jak np. dodanie klienta pomimo wyłączenia programu dane zostają zapisane.

Podsumowując jestem bardzo zadowolony z tego co udało mi się dokonać w tym programie. Kilka miesięcy temu zaczynając z zerową wiedzą, nie uwierzył bym w to, że uda mi się wykonać taki projekt. Dzięki wykonaniu tego zadania zdobyłem sporo cennej wiedzy z zakresu programowania, nauczyłem się pracować samemu i organizować swój czas. Najważniejszą lekcją jest dla mnie, że przy kolejnym projekcie przede wszystkim należy postawić sobie dobre założenia.