

Politechnika Śląska  
Wydział Matematyki Stosowanej  
Kierunek Informatyka

Gliwice, 31.01.2022

Programowanie I  
**projekt zaliczeniowy**

***"TrainTicketer - aplikacja symulująca automat biletów kolejowych"***

**Tomasz Bury gr. lab. 1/1**

## 1. Opis projektu.

Projekt TrainTicketer to aplikacja w języku C++, służąca do symulacji obsługi automatu biletowego wydającego bilety kolejowe, korzystającego z operacji na plikach, w których będzie znajdować się baza połączeń kolejowych, pociągów oraz cen. Aplikacja posiada interfejs graficzny przygotowany z użyciem frameworku QT6, aby urealnić wykorzystanie automatu w prawdziwym życiu, poprzez symulację ekranu dotykowego w postaci myszki.

## 2. Wymagania

- najpierw, użytkownik jest poproszony o wybranie odpowiedniego miasta początkowego i docelowego z listy wczytanych z pliku miasta.txt lub innego, poprawnego miasta
- wybór użytkownika zostaje zapisany do pliku query.dat, uruchamiany jest scraper pobierający połączenia według realnych danych pobieranych ze strony koleo.pl z użyciem aplikacji dołączonej do projektu napisanej w języku Python i „skompilowanej” do pliku .exe, wynik zapisany zostaje do pliku connections.dat
- aplikacja wczytuje połączenia i pokazuje na ekranie dostępne opcje
- na podstawie parametrów podanych przez użytkownika (np. ulgi, typ pociągu) zostanie wygenerowany bilet, za który użytkownik będzie uiszczał opłatę
- opłata będzie realizowana za pomocą odpowiednich przycisków na klawiaturze, symulujących zapłatę bilonem
- po zakończeniu wyświetla się ekran końcowy

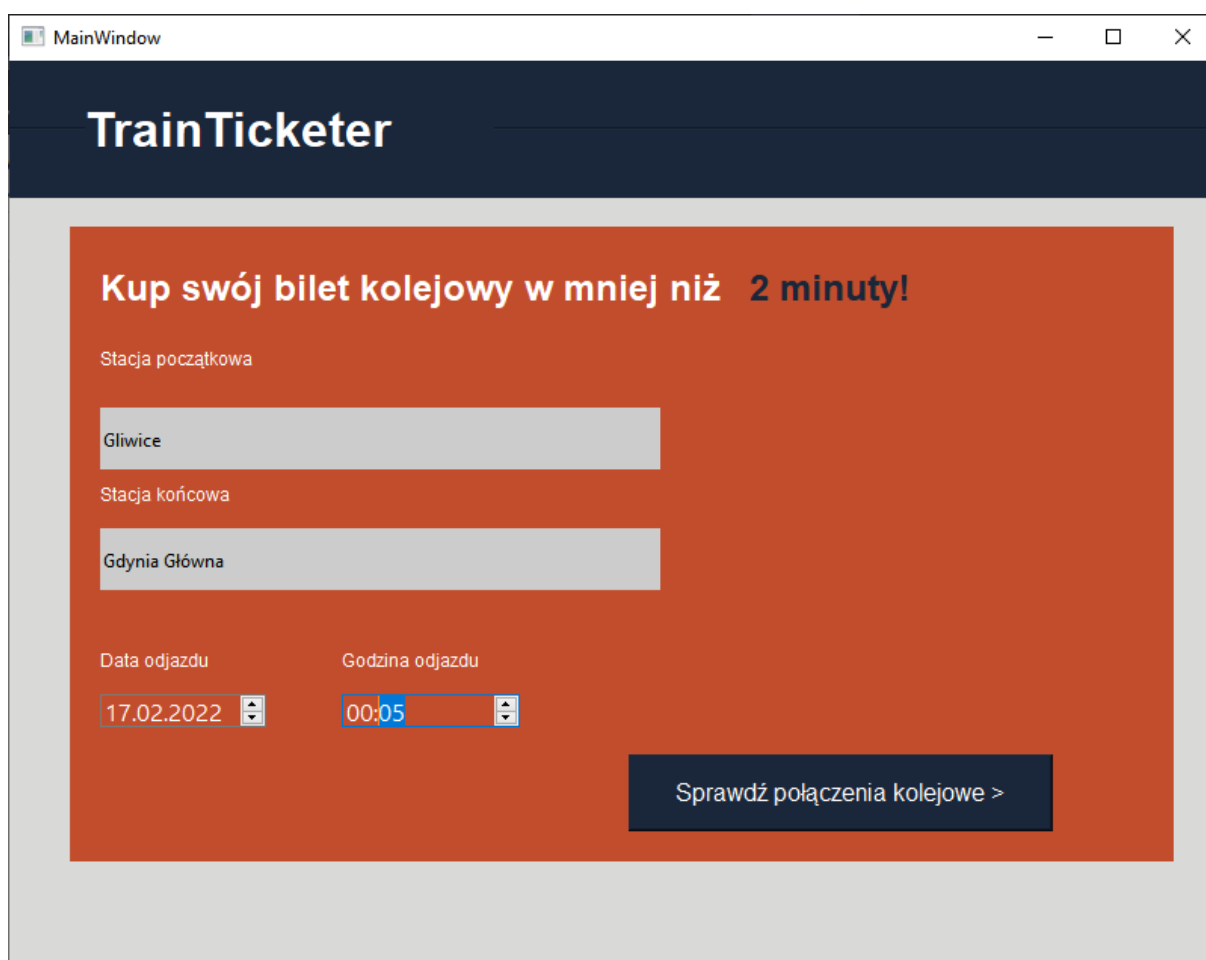
## 3. Przebieg realizacji

Interfejs graficzny projektu został przygotowany w aplikacji Figma, służącej do tworzenia UI aplikacji mobilnych, internetowych i okienkowych. Projekt zbudowany jest na frameworku QT. Jest to zestaw bibliotek służących do tworzenia aplikacji posiadających interfejs graficzny w językach C++, Python oraz Java. Kod aplikacji składa się z plików nagłówkowych **ui\_nazwa\_pliku**, które są definicjami widoków, plików Makefile, będących efektami kompilacji oraz foldery Debug i Release, w których znajdują się pliki .cpp, gdzie znajduje się cała logika aplikacji oraz plik wynikowy .exe. W projekcie znajduje się również plik

**scraper.exe**, który jest “skompilowanym” skryptem Python, który pobiera ze strony koleo.pl realne połączenia kolejowe dla parametrów podanych przez użytkownika.

#### 4. Instrukcja użytkownika

Pierwszym krokiem po otwarciu aplikacji, jest wypełnienie kryteriów wyszukiwania. Aplikacja podpowiada na podstawie wpisywanych liter, jakie miasto należy wyszukać. Nawet, jeżeli miasto nie znajduje się w pliku, zostanie wyszukane, jeżeli jego nazwa jest poprawna. Po wybraniu informacji, klikamy Sprawdź połączenia kolejowe.



The screenshot shows a window titled "MainWindow" with a dark blue header containing the text "TrainTICKETER". The main content area has an orange background and contains the following elements:

- A heading: "Kup swój bilet kolejowy w mniej niż 2 minuty!"
- A label "Stacja początkowa" above a text input field containing "Gliwice".
- A label "Stacja końcowa" above a text input field containing "Gdynia Główna".
- Two date/time pickers: "Data odjazdu" with the value "17.02.2022" and "Godzina odjazdu" with the value "00:05".
- A dark blue button with the text "Sprawdź połączenia kolejowe >" located at the bottom right of the form.

**Uwaga.** Program ze względu na scraping może chwilowo wyglądać na “zawieszony”. Pobranie danych, w zależności od skryptu, może trwać około 5-7 sekund.

Został utworzony interfejs do wyboru połączeń. Klikamy Kup bilet.

Wybrane połączenia kolejowe | TrainTravel

TrainTICKETER

Lista połączeń

Stacja początkowa

Gliwice

Odjazd

01:45

Cena

62.40

Kup bilet

Stacja końcowa

Gdynia Główna

Przyjazd

08:55

Stacja początkowa

Gliwice

Odjazd

03:01

Cena

99.40

Kup bilet

Stacja końcowa

Gdynia Główna

Przyjazd

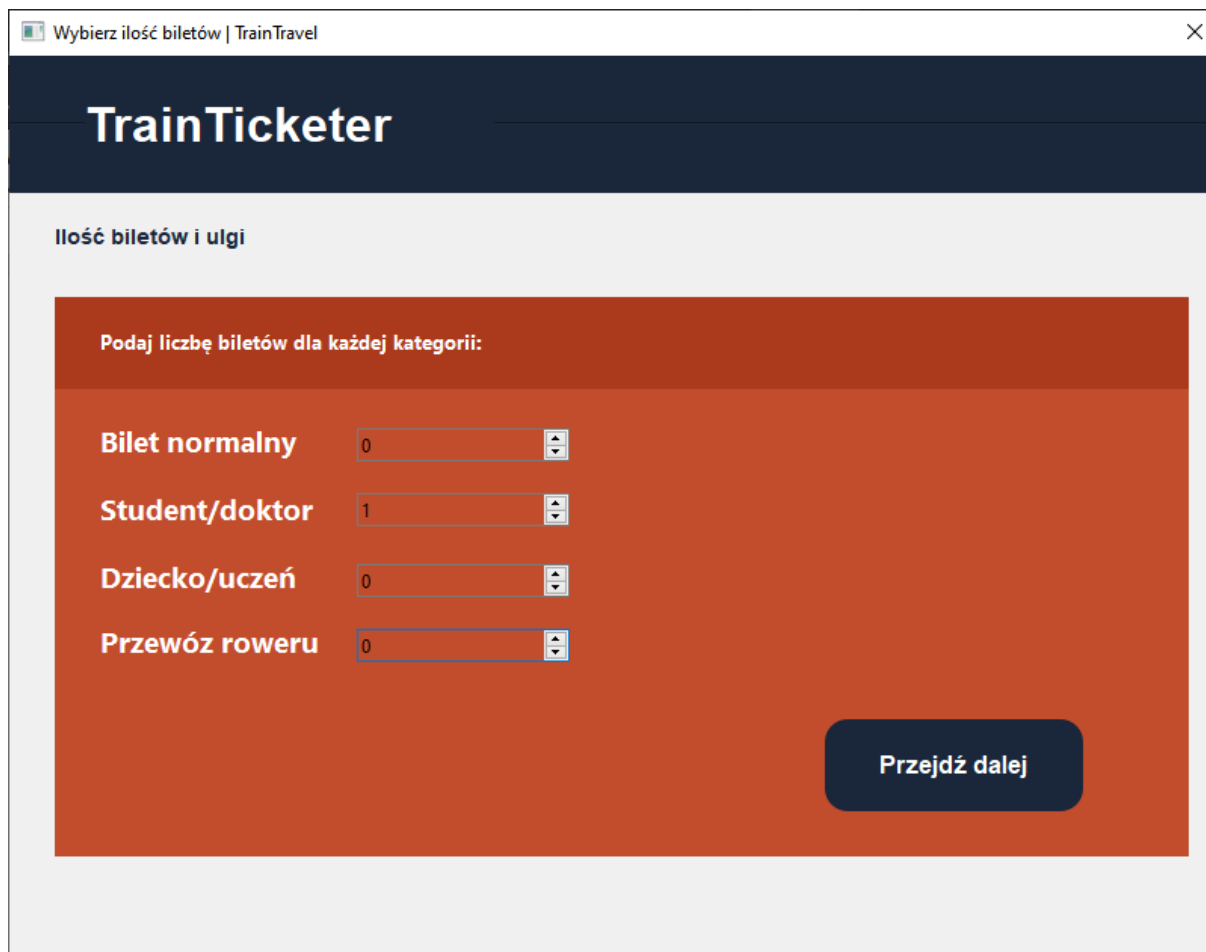
10:24

Stacja początkowa

Gliwice

Odjazd

Wybieramy, jakie ulgi chcemy na bilecie. Klikamy Przejdź dalej.



The screenshot shows a window titled "Wybierz ilość biletów | TrainTravel" with a close button (X) in the top right corner. The main header is "TrainTicketer". Below it, the section is titled "Ilość biletów i ulgi". The instruction "Podaj liczbę biletów dla każdej kategorii:" is displayed. There are four categories with corresponding numeric input fields:

Kategoria	Liczba biletów
Bilet normalny	0
Student/doktor	1
Dziecko/uczeń	0
Przewóz roweru	0

A "Przejdź dalej" button is located at the bottom right of the form area.

Ukaże nam się ekran z biletem. Aby "zapłacić" za bilet, wprowadzamy do automatu odpowiednie monety, klikając 1, 2 lub 5 na klawiaturze. Program będzie automatycznie zmieniać kwoty wyświetlane na ekranie.

Płatność | TrainTravel

×

# TrainTICKETER

## Płatność gotówką

Włóż monety do otworu na monety, znajdującego się po prawej stronie ekranu. Otwór na banknoty znajduje się poniżej terminala płatniczego.

Pozostało do zapłaty

21.2 zł

Wpłacono

10 zł

Bilet jednorazowy

od: Gliwice

01:45

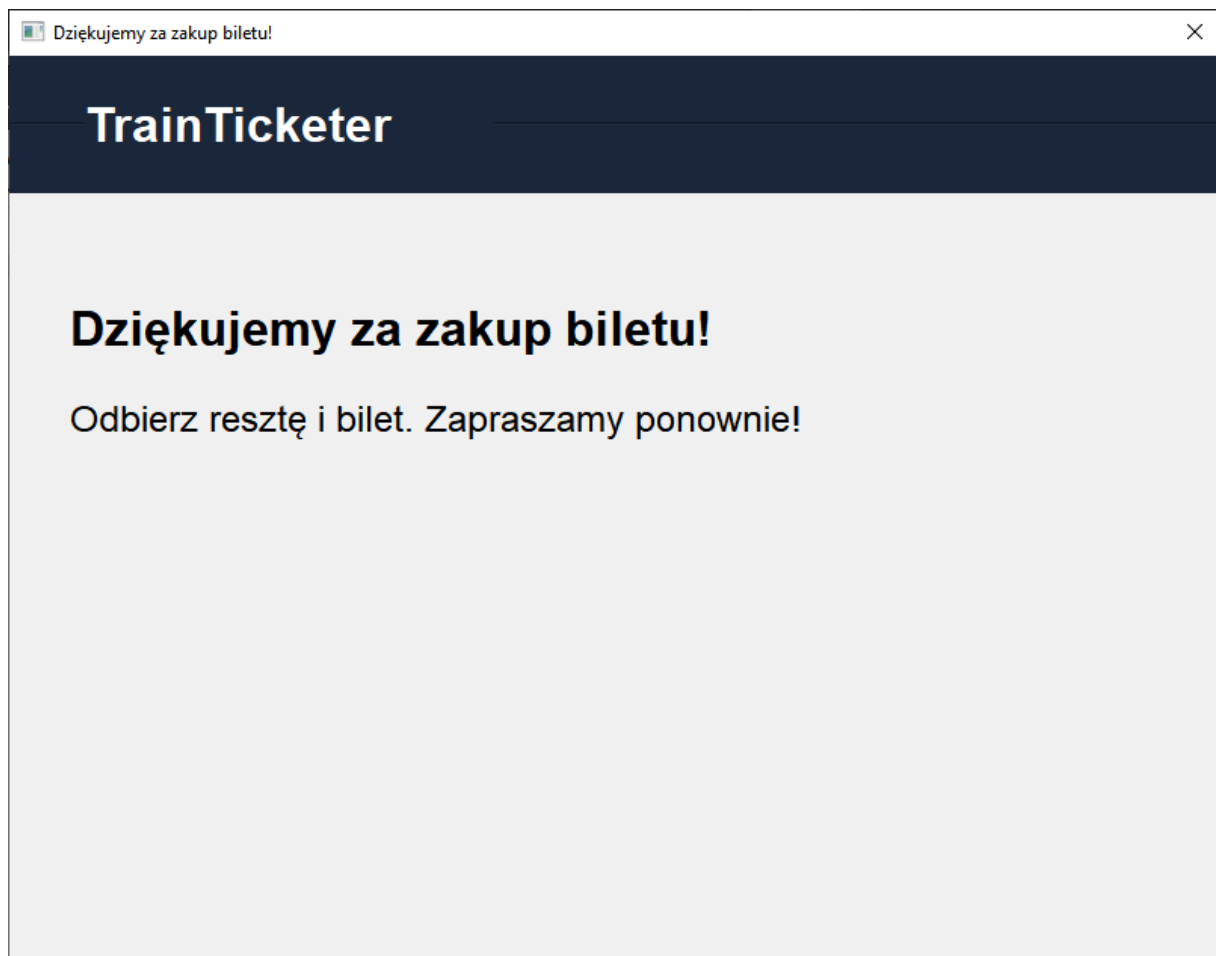
do: Gdynia Główna

08:55

31.2 zł

Klasa 2, przejazd TAM (przedział)

Po osiągnięciu wymaganej kwoty, program pokaże ekran końcowy.



## 5. Podsumowanie i wnioski.

Znaczna większość założeń została zrealizowana. Projekt wczytuje dane z pliku miasta.txt do odpowiedzi, program pobiera realne połączenia ze strony koleo.pl realne połączenia do pliku, działa także podawanie bilonów do automatu. Nie brakuje również operacji I/O na plikach, uruchamiania zewnętrznych aplikacji z poziomu programu, a także operacji na instancjach.

Idea związana z losowaniem zajętości miejsc została porzucona ze względu na założenie, że aplikacja ma być realna, a zajętość mogłaby być zbędną złożonością. Nie udało się zrealizować założenia z generowaniem biletu do pliku .pdf, ze względu na brak zewnętrznych bibliotek do QT pozwalających realizować działanie w sposób responsywny.

Projekt wymaga nieco poprawek wizualnych, które wspomogłyby używalność, w przyszłości mógłby zostać również dodany ekran ładowania, gdy pobierane

są aplikacje, zoptymalizowany scraper. Połączenia mogłyby również posiadać informacje o opóźnieniach.