

Sprawozdanie Jan Naklicki

 Created By

[Fly-us.com](#)

[Cele projektu](#)

[Szczegóły techniczne](#)

[Zmiany w trakcie projektu](#)

[Uruchamianie programu](#)

[Interface](#)

[Klasy](#)

[Podsumowanie](#)

Fly-us.com

Cele projektu

Celem projektu było stworzenie programu obsługującego sprzedaż biletów lotniczych.

Szczegóły techniczne

Projekt został wykonany przy pomocy języka c++ i jego biblioteki `sqlite3` pozwalającej na korzystanie z baz danych SQL.

Zmiany w trakcie projektu

Początkowo projekt miał pozwalać na edycję biletów, osób kupujących, pracowników, lotów, lotnisk, samolotów oraz bagaży. Jednak z racji podobnego charakteru tych rzeczy zdecydowałem na zmniejszenie ilość obsługiwanych rzeczy do dwóch: bilet(zawierają zarówno dane osób jak i cechy szczególne dla biletu) oraz lot.

Uruchamianie programu

Aby uruchomić program należy zainstalować bibliotekę `sqlite3` i uruchomić polecenie:

```
g++ main.cpp Database.cpp Interface.cpp -lsqlite3 -o program -std=c++11 && ./program
```

Druga możliwość to uruchomić gotowy program

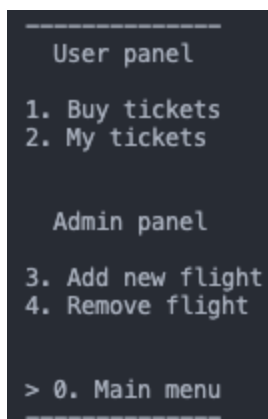
```
./program
```

Interface

Interface graficzny to zwykle wyświetlanie danych w konsoli w przejrzysty dla użytkownika sposób. Z racji ograniczeń bazy danych część rzeczy wyświetlana jest poza interface-m, z braku możliwości zrobienia tego w inny sposób.

W programie wyróżniamy:

- Główne menu
 - Użytkownik
 - Wyświetlanie biletów
 - kupowanie biletów
 - Kupione bilety
 - Admin
 - Dodawanie lotów
 - Usuwaniem lotów



```

User panel
1. Buy tickets
2. My tickets

Admin panel
3. Add new flight
4. Remove flight

> 0. Main menu
```

| FLIGHT | SOURCE | DEST | DEPARTURE | ARRIVAL |
|--------|--------|------|------------------|------------------|
| 1 | GDN | LON | 2020-11-17 15:00 | 2020-11-17 18:10 |
| 2 | GDN | KRK | 2020-11-18 12:00 | 2020-11-18 12:40 |
| 3 | WRO | KRK | 2020-11-18 13:15 | 2020-11-18 13:50 |
| 4 | WRO | KRK | 2020-11-18 13:15 | 2020-11-18 13:50 |
| 11 | KRK | NYC | 2022-06-22 10:00 | 2022-06-22 20:00 |

Do you want to buy a ticket?

Klasy



Interface

```
public:
    ....//UI elements
    ....void displayInterface(string content);
    ....void displayHeader(string content);
    ....void displaySpacer(string above, string below);
    ....//Pages
    ....void mainMenu();
    ....void searchFlight();
    ....void myTickets();
    ....void addNewFlight();
    ....void removeFlight();
};
```

W interface wyróżniamy dwie główne działy:

- UI
- Pages

UI zawiera funkcje pomagające wyświetlać danych. W Pages zawierają się funkcje wyświetlające konkretne widoki, w których wczytywane są dane do dalszej obsługi programu.



Database

```
private:
    ... sqlite3 *DB;
    ... int exit = sqlite3_open("FlyUs.sqlite", &DB);
    ... char *messageError;
    ... static string selectedTable;
    ... static int displayFlights(void *data, int argc, char **argv, char **azColName);
    ... static int displayTickets(void *data, int argc, char **argv, char **azColName);

public:
    ... static string queryForDisplay;
    ... void createDatabase();
    ... static int callback(void *ptr, int argc, char **argv, char **azColName);
    ... void createRecord(string tableName);
    ... void queryFlights();
    ... void queryTickets(string name, string surname);
    ... void insertTicket(string values);
    ... void insertFlight(string values);
    ... void removeFlight(string value);
    ... string generateRandomSeat();
    ... string appendNewValue(string values, bool last, bool date);
};
```

Database zawiera funkcje wywoływane w interfejsie, które wykonują operacje na bazie danych. Przykładowo metody z insert wykonują operacje "INSERT", która dodaje rekord do tablicy.

Są też funkcje pomocnicze takie jak appendNewValue która, tworzy string, który zostanie wklejony np. do funkcji insert tickets.

Podsumowanie

Projekt jest przykładem tego, że nawet w małym projekcie warto jest używać większych bibliotek, które pomagają przy tworzeniu programów. SQL okazał się być bardzo wygodną rzeczą, która bardzo uprościła pracę z danymi.