# PROGRAMOWANIE OBIEKTOWE (JAVA) - PROJEKT Wydział Elektrotechniki Automatyki i Informatyki Politechnika Świętokrzyska Studia: Informatyka, rok II Członkowie zespołu: Daniel Palak Bartosz Nowotnik Michał Pędzik

**Temat projektu:** System obsługi komunikacji miejskiej

#### Opis tematyki projektu

Obraną tematyką naszego projektu jest System obsługi komunikacji miejskiej. Cele jakie obraliśmy na jej podstawie to dostarczanie informacji klientom tj. pasażerom komunikacji miejskiej na temat stanu komunikacji miejskiej (opóźnienia, odjazdy, przyjazdy) oraz samym pracownikom czyli kierowcom.

## Użyte technologie

#### **Frontend**

W naszym projekcie użyliśmy framework'a Swing do utworzenia szaty graficznej czyli GUI naszego programu.

Biblioteka Java Swing jest to część *Java Foundation Classes* służąca do budowania aplikacji okienkowych w języku Java. Swoje działanie opiera o tak zwany AWT czyli *Abstract Windowing Toolkit API*.

#### **Baza Danych**

Mechanizm dostarczania informacji działa w oparciu o bazę danych obsługiwaną za pomocą Hibernate w oparciu o standard Java Persistance API.

Hibernate ORM jest to framework pozwalający deweloperom na szybsze i łatwiejsze pisanie aplikacji dla których dane żyją w bazach danych.

Java Presistance API jest to oficjalny standard mapowania obiektoworelacyjnego dla języka programowania Java.

Silnik bazy danych to który użyliśmy w naszym projekcie to H2 Database Engine.

H2 Database Engine jest silnikiem bazy danych SQL. Charakteryzuje się szybkością i wieloma funkcjami jak na przykład *In-memory databases*.

#### **Backend**

Całość funkcjonalności projektu została stworzona i napisana w oparciu o standardowe biblioteki.

#### **Testowanie**

Testy aplikacji zostały napisane oraz wykonane przy pomocy JUnit 5.

JUnit 5 jest to *testing framework* którego celem jest prostota oraz przydatność przy budowaniu testów aplikacji działających na maszynie wirtualnej Javy.

## Funkcjonalności projektu



Menu główne aplikacji

Główną funkcjonalnością programu jest dostarczanie informacji na temat aktualnego stanu linii, autobusów oraz przystanków dla dwóch grup użytkowników: kierowców oraz pasażerów.



Menu strefy kierowcy

Podając odpowiednie dane o linii, kierunku podróżny lub przystanku może uzyskać informacje o przejeździe.



Komunikat na temat wybranych danych w menu strefy kierowcy

Po wybraniu odpowiednich danych program wyświetla odpowiedni komunikat informujący kierowcę o tym co wybrał oraz aktualną godzinę, jakie jest opóźnienie i kiedy miał nastąpić odjazd.



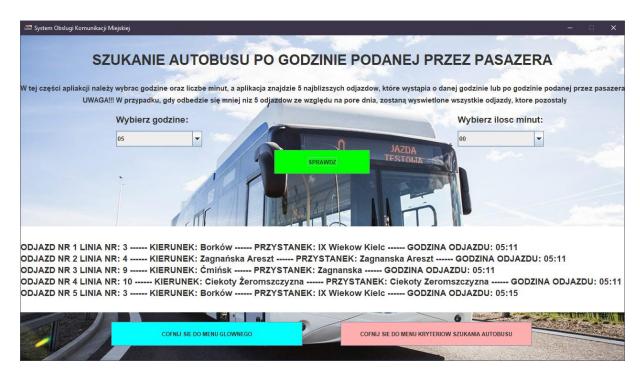
Szukanie autobusu po numerze linii

W przypadku strefy klienta tj. pasażera, podając odpowiednie dane takie jak numer linii i kierunek podróży wyświetla nam się lista odjazdów i odpowiadających im godzin.



Szukanie autobusu po godzinie odjazdu

Pasażer może także skorzystać z opcji wyszukiwania autobusów po podanej godzinie.



Wyniki wyszukiwania po godzinie odjazdu

Jako wynik wyszukiwania dostajemy listę z odjazdami oraz ich godzinami.



Wyszukiwanie po przystanku

Możliwe jest także wyszukiwanie wszystkich dostępnych odjazdów na danym przystanku.

#### Sposób uruchomienia i użycia programu

Użycie programu opiera się o zwyczajne klikanie odpowiednich przycisków i podejmowanie wyborów z rozwijanych list dostępnych w danym menu.

Uruchomienie programu podlega wczesnej kompilacji kodu źródłowego. Należy to zrobić przy pomocy jednego z wybranych IDE najlepiej Ineliji IDEA. Po skompilowaniu kodu można uruchomić go za pomocą przycisku RUN.

### Informacje na temat kodu źródłowego

#### Klasa GUI

Jest to klasa odpowiadająca za cały interfejs graficzny naszego projektu. Dziedziczy ona po *JFrame* i implementuje interfejs *ActionListener*. W postaci pól klasy zawarte są tu wszystkie etykiety, nagłówki itp. Klasa zawiera również wszystkie inne elementy UI takie jak przewijane pola tekstowe, przyciski itp.

#### Klasa Bus

Klasa ta jest traktowana jako encję w bazie danych przedstawiającej wzór informacji o danym autobusie.

## Klasa BusStop

Podobnie jak klasa Bus, przedstawia wzór informacji o przystanku jako encja w bazie danych.

# Klasa BusRepository

Jest to klasa dziedzicząca po GUI. Odpowiada ona za dodawanie autobusów do bazy danych.

# Klasa BusStopRepository

Klasa dziedzicząca po BusRepository. Odpowiada ona za dodawanie przystanków do bazy danych.

# Rozkład pracy na poszczególnych członków

## Daniel Palak - 2ID13A

- Lider zespołu
- Główny programista
- Designer
- Tester
- Debugowanie

#### Bartosz Nowotnik – 2ID13A

- Tester
- Debugowanie
- Designer
- Poboczny programista

# Michał Pędzik – 2ID13B

- Tester
- Redaktor
- Debugowanie
- Poboczny programista