**Piotr Małecki**

**Michał Daniluk**

**Zaawansowane Programowanie w C++**

**Dokumentacja Końcowa**

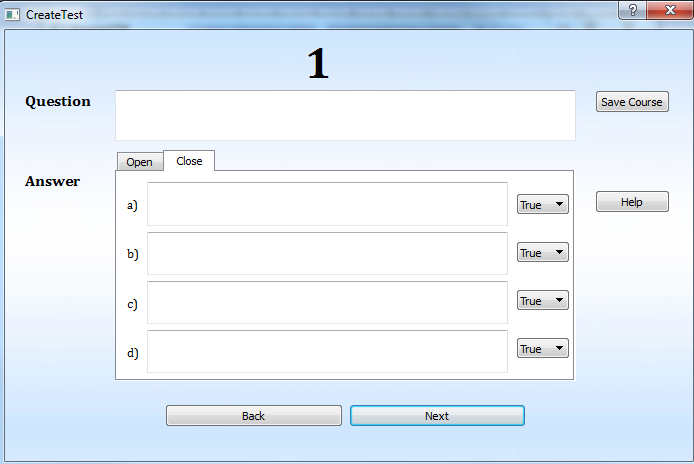
**Temat projektu:** Program wspomagający proces uczenia się w stylu pytanie-odpowiedź.

1. **Dokumentacja funkcjonalna**

Program wspomaga proces uczenia się poprzez odpowiednie dobieranie dat powtórzenia odpowiednich pytań. Założeniem programu jest regularne z niego korzystanie, w przeciwnym wypadku pytania nieodrobione kumulują się. Program wylicza datę powtórki każdego z pytań wg algorytmu ….MOZE TU OPISAC TEN ALGORYTM?

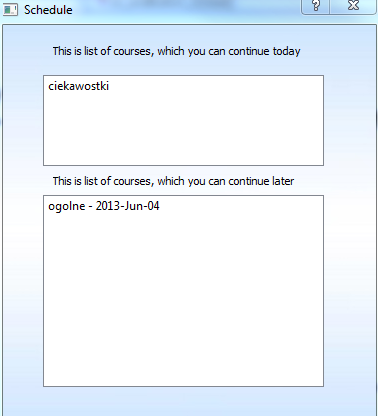
Aplikacja została wyposażona w graficzny interfejs użytkownika. Program został zrealizowany wykorzystując wzorzec projektowy Model Widok Kontroler. Dzięki niemu udało się oddzielić logikę aplikacji od GUI, więc program niezależnie od intensywności wykonywanych obliczeń pozostaje wrażliwy na akcje użytkownika.

**Tworzenie kursu**



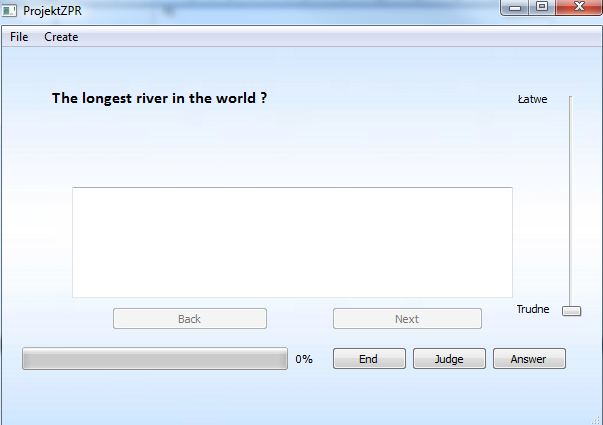
Użytkownik ma możliwość stworzenia własnego kursu poprzez tworzenie zadań mając do wyboru odpowiedź otwartą lub zamknięta. Po wybraniu opcji *Save Course* kurs zapisuje się do pliku XML.

**Plan kursów do przerobienia**

****

Użytkownik po wybraniu odpowiedniej opcji ma możliwość przeglądania listy kursów do przerobienia. Wybierając *Start*  z menu głównego użytkownik ma możliwość zaczęcia kursu od nowa lub kontynuacji.

**Rozpoczęcie nowego kursu**



Podczas robienia kursu, użytkownikowi prezentowane jest pytanie oraz miejsce na udzielenie odpowiedzi w przypadku pytania otwartego lub zestaw odpowiedzi zamkniętych z możliwością wyboru poprawnych. Po udzieleniu odpowiedzi użytkownik powinien nacisnąć przycisk *Answer.* Pojawi się wtedy prawidłowa odpowiedź orazpropozycja oceny, jednak użytkownik ma możliwość zmiany tej propozycji. Jest to uzasadnione w przypadku jeżeli błąd użytkownika to zwykła literówka. Zmianę oceny zatwierdza przyciskiem *Judge.* Po zakończeniu kursu i kliknięciu *End* wyświetlane są statystyki przerobionego kursu i zostaje on poddany serializacji. TU możesz dopisac dwa zdania o serializacji.

1. **Budowa aplikacji**

Program jest zrealizowany w architekturze Model Widok Kontroler. Do stworzenia interfejsu graficznego wykorzystaliśmy bibliotekę QT. Wielokrotnie wykorzystywaliśmy bibliotekę boost. Między innymi do serializacji, odczytu plików XML oraz wykorzystywaliśmy sprytne wskaźniki.

Zestaw użytych klas należących do Widoku:

* + - **View** – reprezentuje główne okno programu
    - **CreateTest** – reprezentuje okno tworzenia kursu
    - **Schedule**– reprezentuje okno zaplanowanych kursów
    - **StartMenu**– reprezentuje okno wyboru kursu
    - **Statistic**– reprezentuje okno statystyk

Zestaw użytych klas należących do Modelu:

* **Model** – klasa reprezentuje całą logikę aplikacji.
* **Course** – klasa reprezentująca logikę tworzenia nowego kursu.
* **Deck**
* **Mark ­**– pomocnicza klasa obliczająca proponowaną ocenę użytkownikowi po udzieleniu odpowiedzi.
* **QuaestionCard**
* **Start –** klasa reprezentuje logikę rozpoczęcia lub kontynuacji kursu

**Controller** – jest to klasa Kontrolera, która umożliwia połączenia widoku oraz kontrolera. W aplikacji wykorzystano mechanizm sygnałów i slotów wykorzystywany w QT. W celu komunikacji z Kontrolerem, Widok emituje sygnały a Kontroler tworzy sloty. Pozostałe okna dialogowe również emitują sygnały, ale wywołują przy tym sygnału Widoku, które następnie wywołują sloty kontrolera.

Dostarczoną dokumentację kodu wygenerowaliśmy używając narzędzia Doxygen.

1. **Napotkane trudności**

Pierwszą napotkaną trudnością było przenoszenie kodu źródłowego pomiędzy członkami zespołu. Aby to rozwiązać zapoznaliśmy się z systemem kontroli wersji GIT i go używaliśmy wykorzystując serwis github.

Następnym problemem było zapisanie obiektu przerobionego kursu ze wszystkimi ocenami w celu późniejszej kontynuacji. Wykorzystaliśmy do tego mechanizm serializacji.