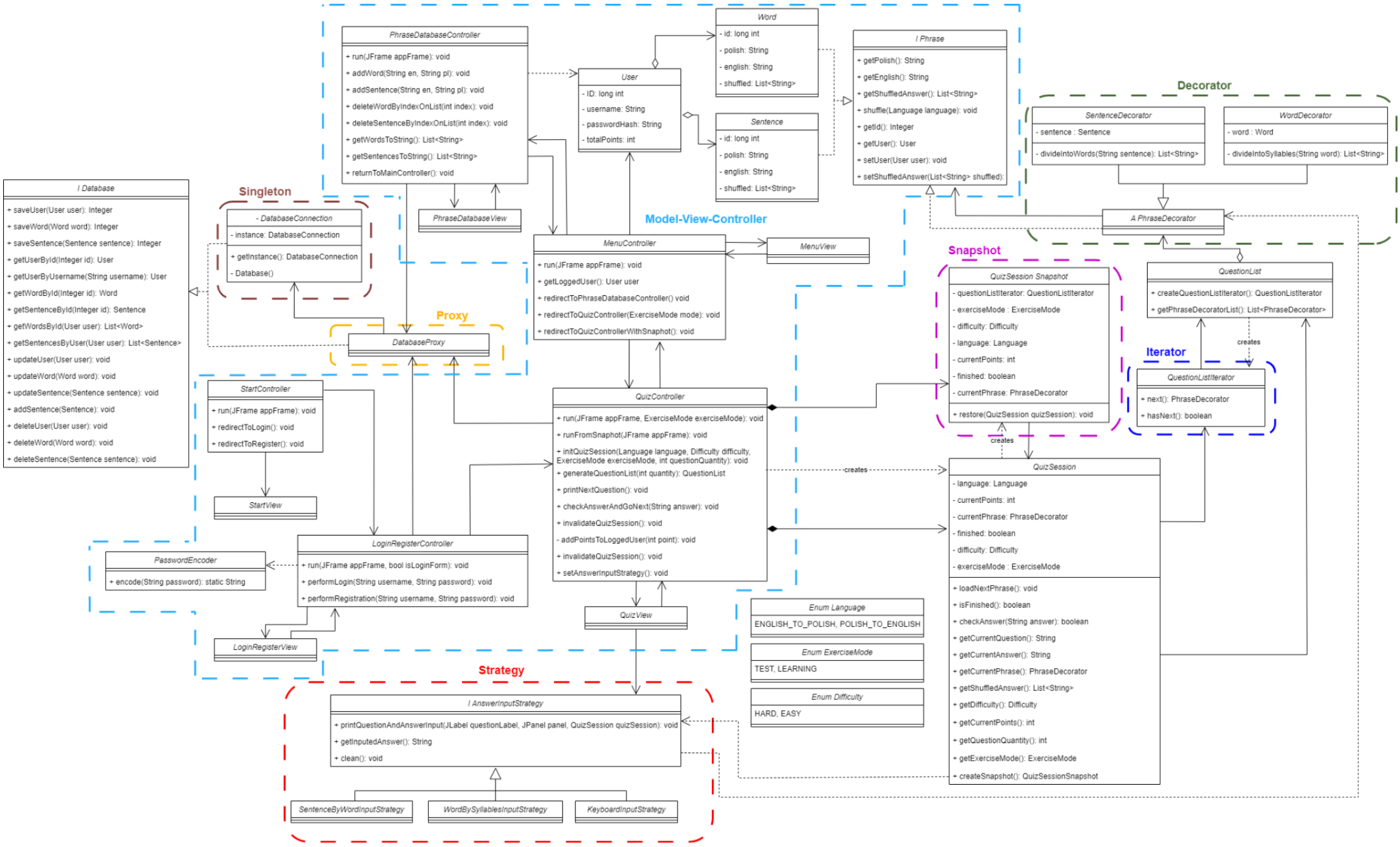
1. Opis projektu

Celem projektu jest stworzenie aplikacji desktopowej do nauki języka angielskiego, wykorzystując przy tym różne wzorce projektowe.
Aplikacja oferuję naukę z polskiego na angielski lub z angielskiego na polski.
Może ona również pracować w trybie nauki lub w trybie testu. Tryb nauki sprawia, że użytkownik nie może przejść do następnego pytania, gdy nie odpowiedział poprawnie (ma nieskończoną ilość szans). Tryb testu sprawia, że użytkownik odpowiada na wszystkie pytania (jedna szansa) i na koniec dostaje wynik, który potem dodaje się do jego sumarycznych punktów na koncie.
Użytkownik może również wybrać poziom trudności: Łatwy - zdania układa się z rozsypanych słów, a słowa układa się z rozsypanych sylab. Trudny - zdania oraz słowa wpisuje się z klawiatury.

Funkcjonalności aplikacji:

- Rejestracja użytkownika:
    - Użytkownik może utworzyć konto, podając login i hasło.
- Logowanie:
    - Zarejestrowani użytkownicy mogą zalogować się wprowadzając odpowiednie dane.
- Strona główna:
    - Po zalogowaniu użytkownik widzi swoją nazwę użytkownika oraz sumaryczną punktację z testów.
    - Użytkownik ma możliwość rozpoczęcia nowego quizu bądź kontynuacji ostatniego niedokończonego quizu.
    - Użytkownik może przejść do bazy fraz.
- Nauka słówek w dwie strony polski-angielski, angielski-polski
- Wybór trybów Quizu:
    - Tryb Nauki - Program nie pozwala przejść do kolejnego pytania jeżeli nie podamy prawidłowej odpowiedzi.
    - Tryb Testu - Program wyświetla tylko podsumowanie na koniec testu z liczbą poprawnych odpowiedzi, co skutkuje przydzieleniem odpowiedniej ilości punktów.
- Poziomy trudności do wyboru:
    - Poziom łatwy - Umożliwia użytkownikowi udzielenie odpowiedzi na pytanie poprzez układanie zdań z podanych słów, bądź układania słów z podanych sylab.
    - Poziom trudny - Umożliwia użytkownikowi udzielenie odpowiedzi na pytanie poprzez wpisanie jej z klawiatury.
- Wyświetlanie kolejnych zdań, słów w trakcie testu.
- Możliwość przerwania testu oraz wznowienie go zachowując dotychczasowy postęp.
- Możliwość edytowania bazy danych fraz, dodawanie/usuwanie słów/zdań.

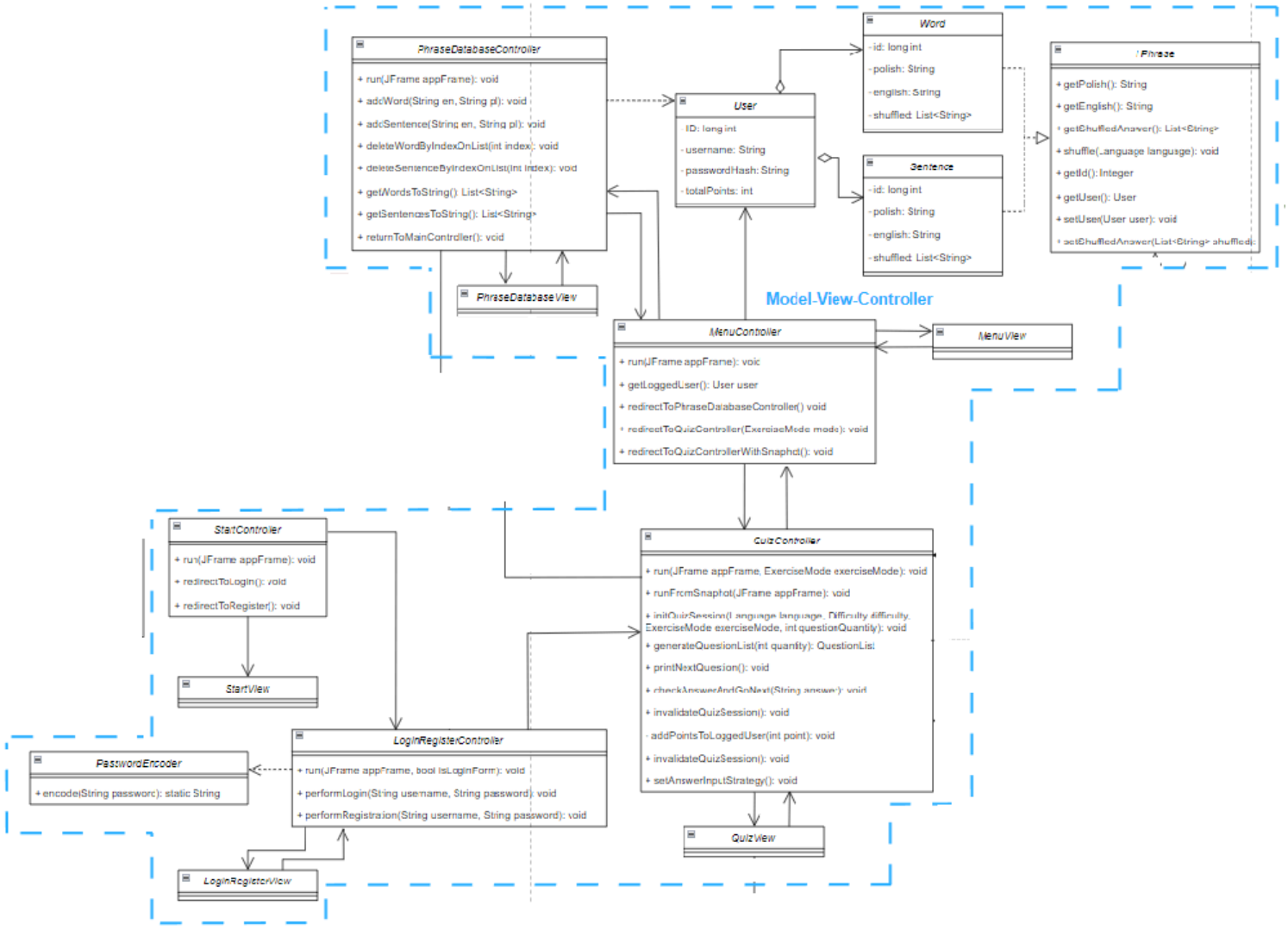
2. Diagram klas (zaimplementowanych w programie)



3. Opis poszczególnych wzorców projektowych

Wzorce Architekturalne:

Model-View-Controller (MVC)



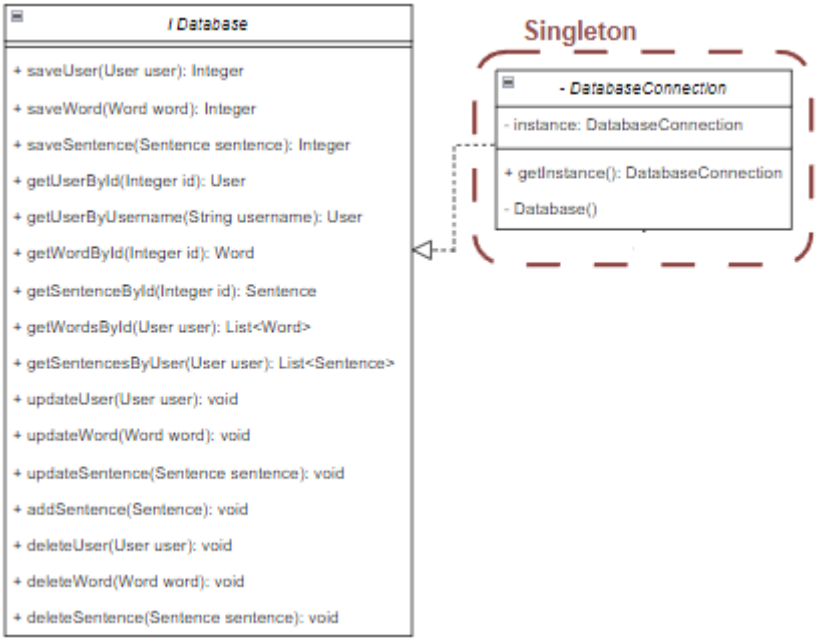
Cel użycia:
Organizacja struktury projektu poprzez oddzielenie od siebie klas danych (model), kontrolerów oraz widoków.
Klasy modelu definiują dane używane w projekcie.
Widoki mają za zadanie tylko wyświetlać przekazane dane w odpowiedniej formie oraz przesyłać dane wprowadzane przez użytkownika.
Kontrolery sterują całym programem. Są odpowiedzialne za przekierowania między modułami, wyświetlanie widoków, przekazywanie danych i wywoływanie wrzystkich operacji.

Lokalizacja w projekcie:

- Klasy modelu: *pl.ztplingo.model*
- Klasy widoków: *pl.ztplingo.view*
- Klasy kontrolerów: *pl.ztplingo.controller*

Wzorce Kreacyjne:

Singleton



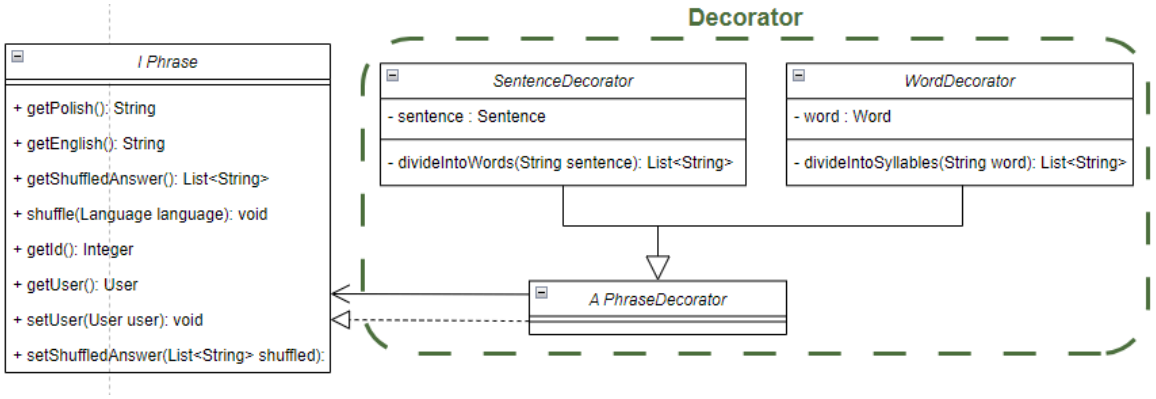
Cel użycia:
Klasa *DatabaseConnection* łączy się z bazą danych i wykonuje operacje SQL. Ważne jest aby aplikacja miała tylko jedno połączenie do bazy danych, ponieważ zwiększa to bezpieczeństwo przed wielowątkowym utworzeniem, ułatwia zarządzanie połączeniem oraz oszczędza zasoby.

Lokalizacja w projekcie:
Klasa *pl.ztplingo.database.DatabaseConnection*.

Użycie:
Dostęp do instancji połączenia z bazą danych mają tylko obiekty klasy pośrednika *pl.ztplingo.databaseDatababaseProxy* (więcej o tym przy opisie wzorca Proxy)

Wzorce Strukturalne:

Dekorator



Cel użycia:
Dekorator implementuję interfejs oraz przechowuję instancję frazy (*Phrase*). Następnie w zależności od ustawień oraz tego, czy fraza jest słowem, czy zdaniem, odpowiednio przygotowuje pytanie. *SentenceDecorator* ma za zadanie przekształcać zdania (*Sentence*), rozdzielając je na pojedyncze wyrazy, *WordDecorator* ma na celu rozbicie wyrazów (*Word*) na sylaby.

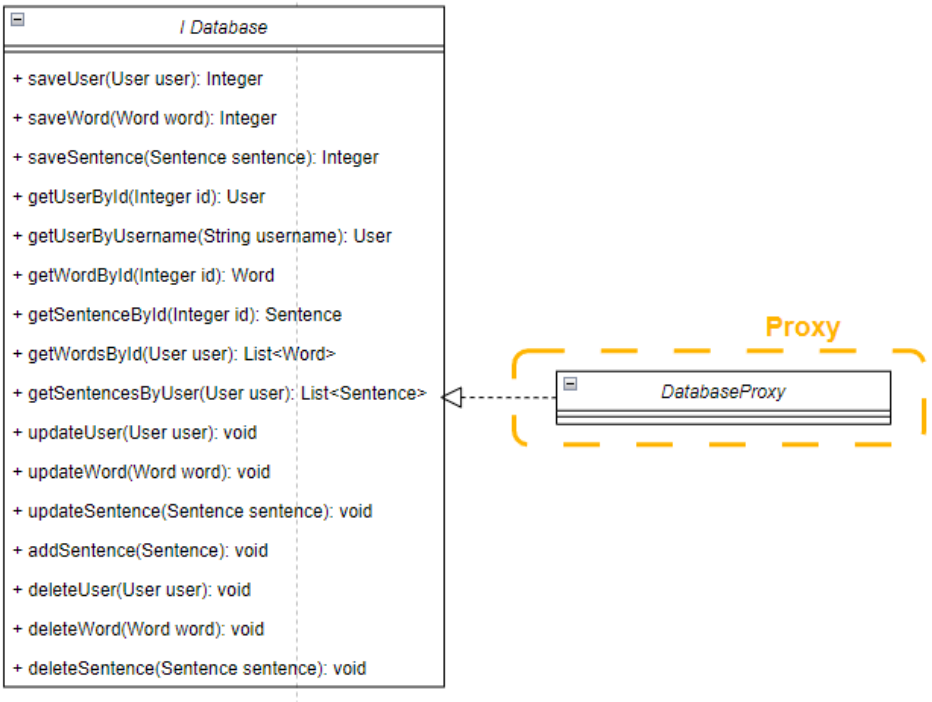
Lokalizacja w programie:
pl.ztplingo.database.DatabaseProxy

Użycie:
Frazy pobierane są z bazy danych i na ich podstawie w klasie QuestionList tworzone są dekoratory (reprezentujące frazę z gotowym pytaniem do quizu), które następnie używane są w klasach QuestionListIterator i QuizSession (co za tym idzie również w QuizSessionSnapshot).

Wektor zmian:
Wektorem zmian dekoratora jest interfejs *Phrase* oraz funkcjonalność tworzonych obiektów.

Możliwość rozbudowy projektu:
Rozbudowa interfejsu *Phrase* oraz dodanie nowych funkcjonalności.

Pełnomocnik (Proxy)



Cel użycia:
Wzorzec w naszym projekcie zakłada stworzenie pośredniczącej klasy o identycznym interfejsie (*Database*) co klasa *DatabaseConnection*. Klasa *DatabaseConnection* wykonuje tylko operacje SQL na bazie danych, natomiast klasa *DatabaseProxy* służy do weryfikacji poprawności żądań przed przekazaniem ich do bazy danych. Gdy pełnomocnik zweryfikuje żądanie pomyślnie, przekazuje je do oryginalnej klasy bazy danych. W ten sposób pełnomocnik działa jako filtr, kontrolując i zarządzając dostępem do bazy danych na podstawie określonych kryteriów np. sprawdzanie, czy użytkownik już istnieje, sprawdzanie, czy słowo lub zdanie jest w poprawnym formacie.

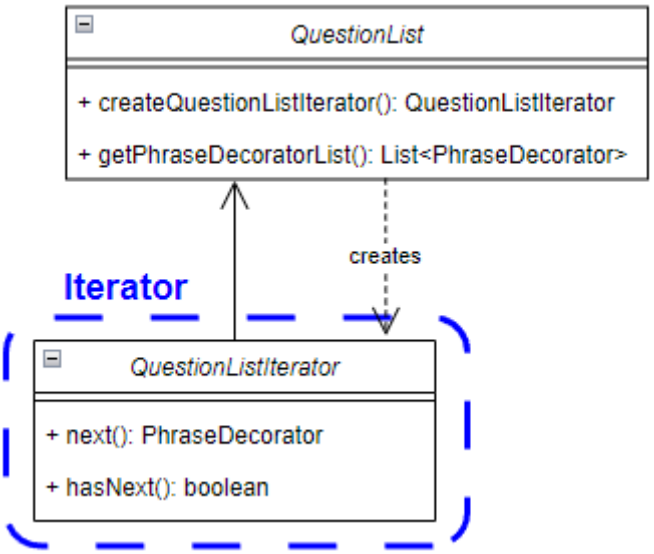
Lokalizacja w programie:
Klasa *pl.ztplingo.database.DatabaseProxy*

Użycie:
Instancję obiektu *DatabaseProxy* przechowują: *PhraseDatabaseController* - w celu zapisu/odczytu fraz do bazy danych, *LoginRegisterController* - w celu zapisu/odczytu użytkowników do bazy danych, *QuizController* - w celu pobierania fraz oraz dodawania punktów użytkownikom.

Wektor zmian:
Wektorem zmian pośrednika jest interfejs *Database* oraz zmiany w sprawdzaniu zapisywanych wartości lub zmiany w dostępie do połączenia z bazą danych.

Wzorce Behawioralne:

Iterator



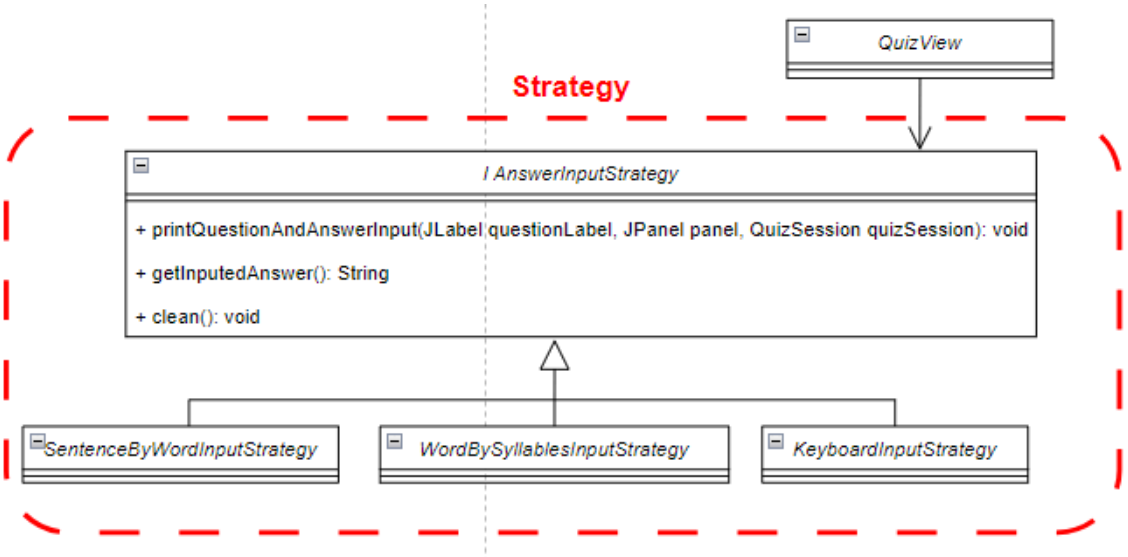
Cel użycia:
Wzorzec zaimplementowany w celu sekwencyjnego przechodzenia po pytaniach do quizu bez konieczności używania konkretnych typów kolekcji obiektów. W projekcie klasa *QuestionListIterator* "przechodzi" po pytaniach w obiekcie klasy *QuestionList.* Ułatwia to pobieranie kolejnych pytań oraz sprawdzania, czy pytania się skończyły.

Lokalizacja w programie:
Klasa *pl.ztplingo.iterator.QuestionListIterator*

Użycie:
Obiekt klasy *QuestionList* tworzy i zwraca "swój" iterator. Obiekt klasy *QuizSession* wywołuje utworzenie oraz przechowuje tylko iterator (bez kolekcji pytań), z którego korzysta do pobierania kolejnych pytań.

Wektor zmian:
Wektorem zmian iteratora jest używana struktura danych do przechowywania listy pytań w klasie *QuesionList.*

Strategia



Cel użycia:
Wzorzec jest odpowiedzialny za interakcję z użytkownikiem podczas procesu wczytywania informacji. Strategia wybierana jest w zależności od poziomu trudności aplikacji oraz od tego, czy ułożyć trzeba słowo (*Word*), czy zdanie (*Sentence*), umożliwiając użytkownikowi składanie zdań z dostępnych wyrazów, układanie słów z sylab bądź wpisywanie odpowiedzi za pomocą klawiatury.

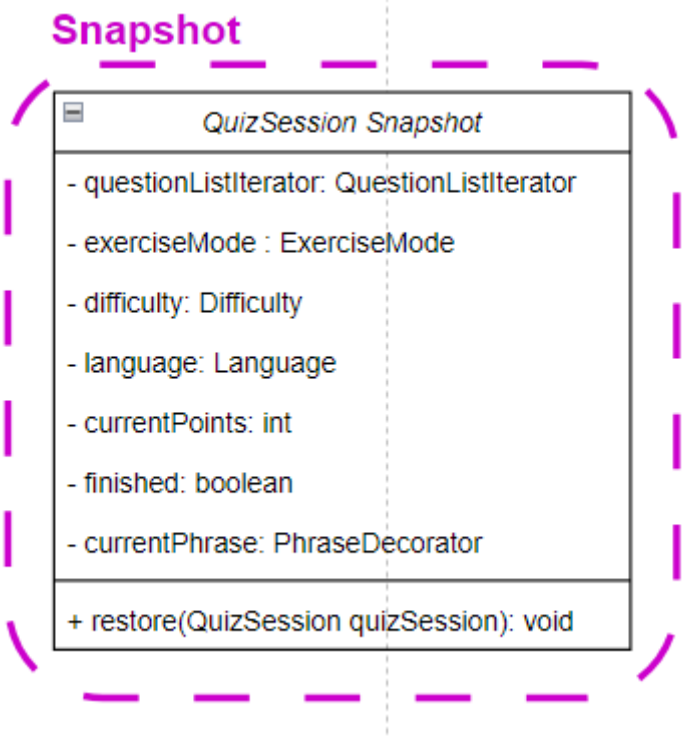
Lokalizacja w programie:
Wszystkie klasy pakietu *pl.ztplingo.strategy*

Użycie:
Instancja *AnswerInputStrrategy* przechowywana jest w widoku *QuizView*. Kontroler *QuizController* każe obiektowi *QuizSession* pobrać kolejne pytanie i na jego podstawie ustawia widokowi jedną z trzech strategii.

Wektor zmian:
Wektorem zmian strategii jest sposób wyświetlania “inputu” i jego interakcja z użytkownikiem.

Możliwość rozbudowy projektu:
Projekt mógłby zostać rozbudowany o dodatkowe możliwości udzielania odpowiedzi (np. wybór jednej z 4 odpowiedzi, uzupełnianie luk we frazach).

Pamiątka



Cel użycia:
Wzorzec został zastosowany w celu umożliwienia użytkownikowi przerwania wybranego quizu. Użytkownik ma potem możliwość wznowić ostatnią przerwaną sesję, zachowując osiągnięty postęp. Może również wykonywać inne quizy, a pamiątka zostanie nadpisana dopiero, gdy użytkownik “przerwie” inny quiz.

Lokalizacja w programie:
Klasa *pl.ztplingo.snapshot.QuizSessionSnapshot*

Użycie:
Pamiątka (Memento): Klasa *QuizSessionSnapshot*
Opiekun (Caretaker): Klasa *QuizController*
Oryginalny obiekt (Originator): Klasa *QuizSession*

Wektor zmian:
Wektorem zmian pamiątki jest budowa klasy QuizSession.

Możliwość rozbudowy projektu:
Zamiast przechowywać jedną pamiątke można sprawić, aby przerwany quiz trafiał na listę pamiątek, z które potem będzie można wybrać, którą sesję chcemy wznowić.

4. Opis co najmniej jednego rozwiązania specyficznego dla użytej technologii (Java)

1. Biblioteka javax.swing:

Swing jest biblioteką javy, która dostarcza zestaw narzędzi i komponentów do tworzenia graficznego interfejsu użytkownika (GUI) w aplikacjach graficznych opartych na języku Java. Pakiet javax.swing wprowadza komponenty bardziej zaawansowane niż wcześniejsza biblioteka AWT (Abstract Window Toolkit).
Oto kilka użytych przez nas klas i komponentów dostępnych w pakiecie javax.swing:

- *JFrame* - Klasa reprezentująca główne okno aplikacji. Kontener dla innych komponentów GUI.
- *JPanel* - Komponent służący do grupowania innych komponentów wewnątrz siebie. Może być używany jako obszar roboczy dla innych komponentów.
- *JButton* - Przycisk, który można kliknąć, aby uruchomić określoną akcję.
- *JTextField* - Pole tekstowe, które pozwala użytkownikowi wprowadzać tekst.
- *JLabel* - Etykieta służąca do wyświetlania tekstu lub ikony.
- *JComboBox* - Lista rozwijana, umożliwiająca użytkownikowi wybieranie jednej z dostępnych opcji.

Źródła:

- Oficjalna dokumentacja: https://docs.oracle.com/javase/7/docs/api/javax/swing/package-summary.html
- Tutorial do swinga: https://www.javatpoint.com/java-swing

2. Framework Hibernate:

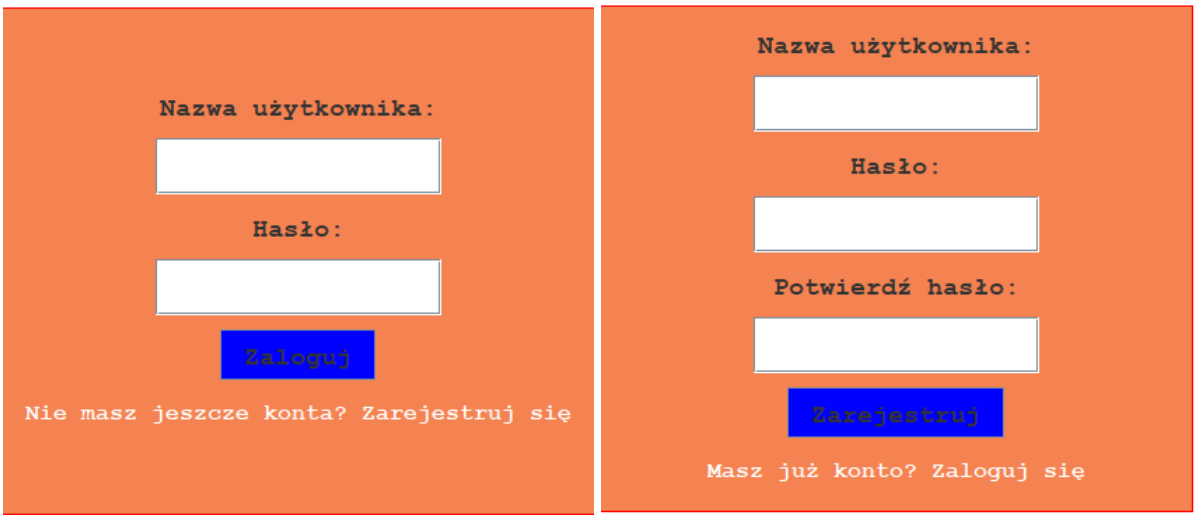
Hibernate jest frameworkiem do mapowania obiektowo-relacyjnego w języku Java. Umożliwia on wygodne i efektywne zarządzanie danymi w aplikacjach. Stanowi jedną z najpopularniejszych implementacji Java Persistence API (JPA). Wykorzystanie Hibernate umożliwiło nam zdefiniowanie tabel, kolumn oraz relacji między klasami *User, Word, Sentence* za pomocą odpowiednich adnotacji, takich jak *@Entity, @Table, @Column, @OneToMany*, co pozwoliło na zapis obiektów tych klas do relacyjnej bazy MySQL.

Źródła:

- Oficjalna dokumentacja: https://hibernate.org/orm/documentation/6.4/ https://docs.jboss.org/hibernate/core/3.3/reference/en/html/
- Strona poświęcona Javie: https://www.baeldung.com/

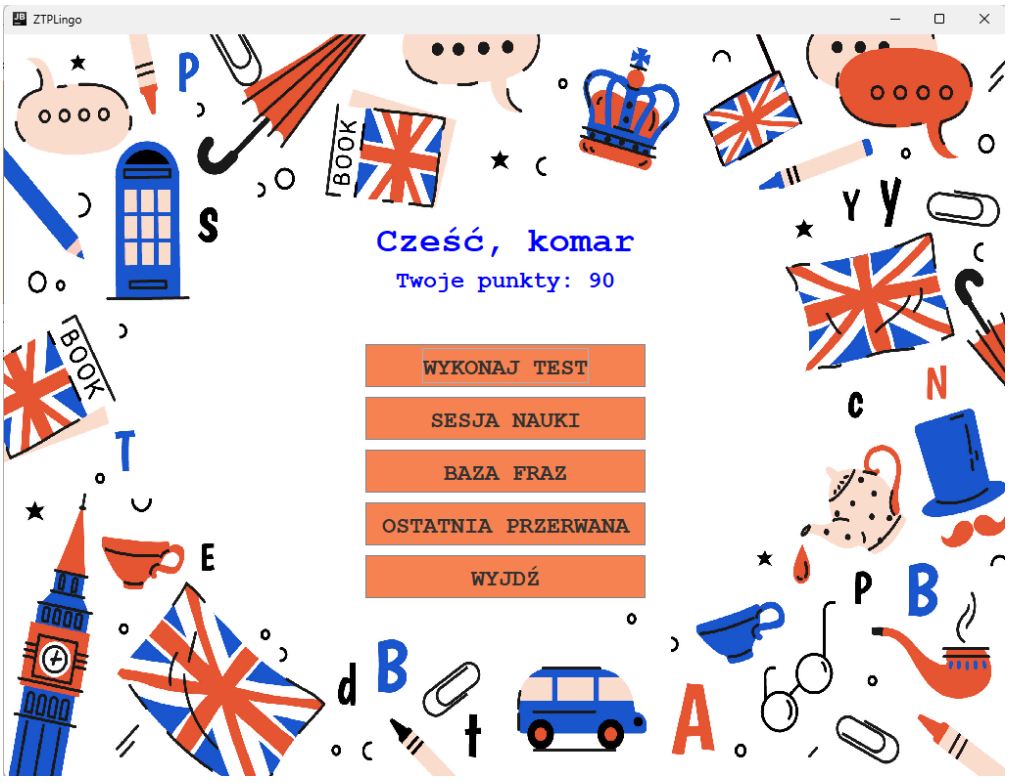
5. Instrukcja użytkownika

Rejestracja, logowanie:



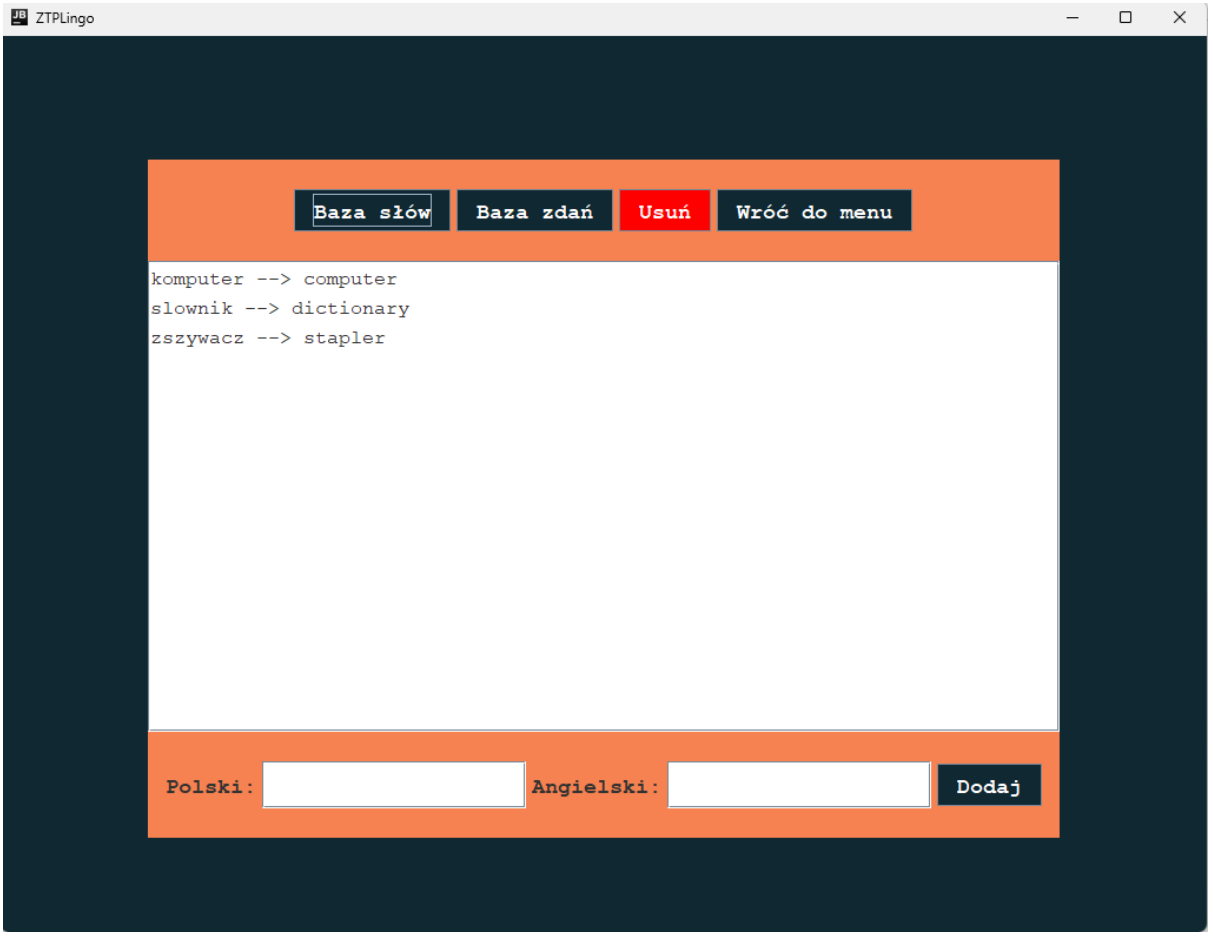
Użytkownik rejestruje/loguje się poprzez podanie nazwy użytkownika i hasła.

Główne menu:



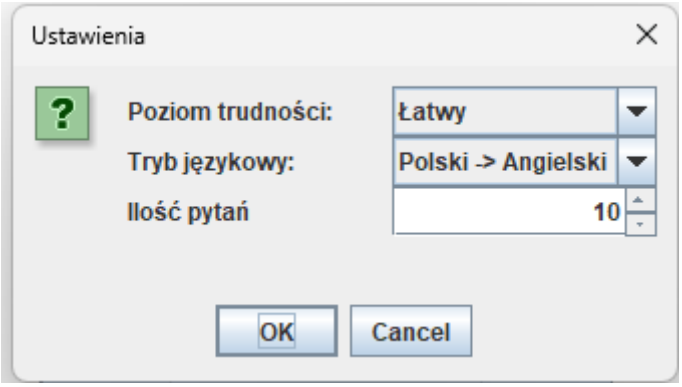
Wykonaj Test - wykonanie quizu w trybie testu
Sesja Nauki - wykonanie quizu w trybie nauki
Baza Fraz - przejście do widoku bazy fraz
Ostatnia Przerwana - powrót do ostatniej przerwanej sesji (nauki lub testu)
Wyjdź - zamknięcie aplikacji
*Twoje punkty - łączna punktacja ze wszystkich wykonanych testów.

Baza fraz:



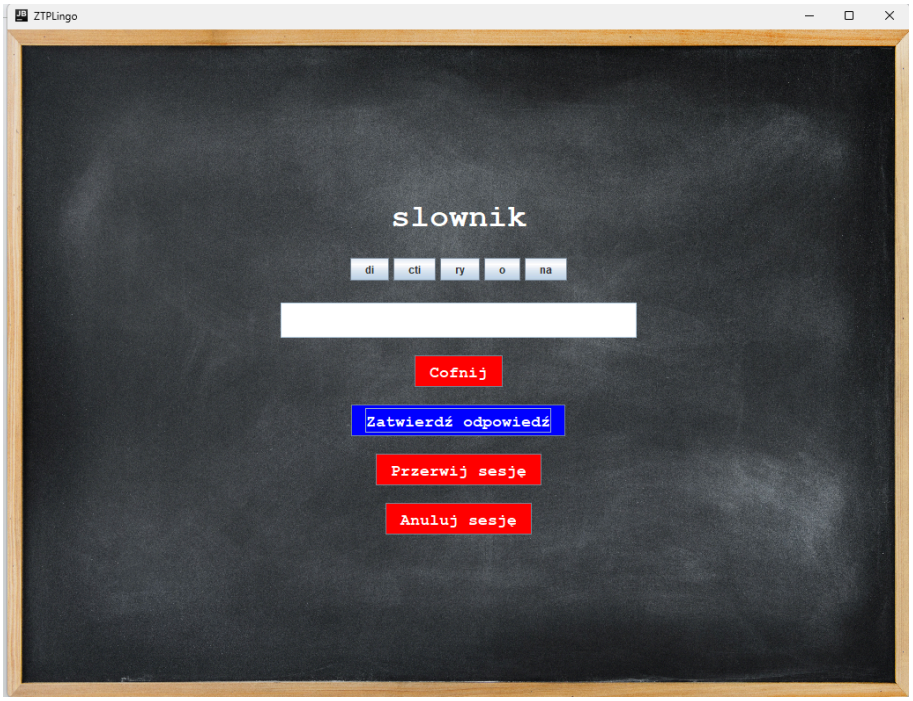
Z poziomu tego widoku użytkownik może dodawać i usuwać frazy - zarówno słowa, jak i zdania. Aby dodać słowo, należy przejść do bazy słów, wpisać polskie słowo i angielskie tłumaczenie i wcisnąć przycisk *Dodaj* (analogicznie z dodawaniem zdań). Aby usunąć frazę, należy wybrać ją myszką i wcisnąć przycisk *Usuń*.

Sesja Quizu:



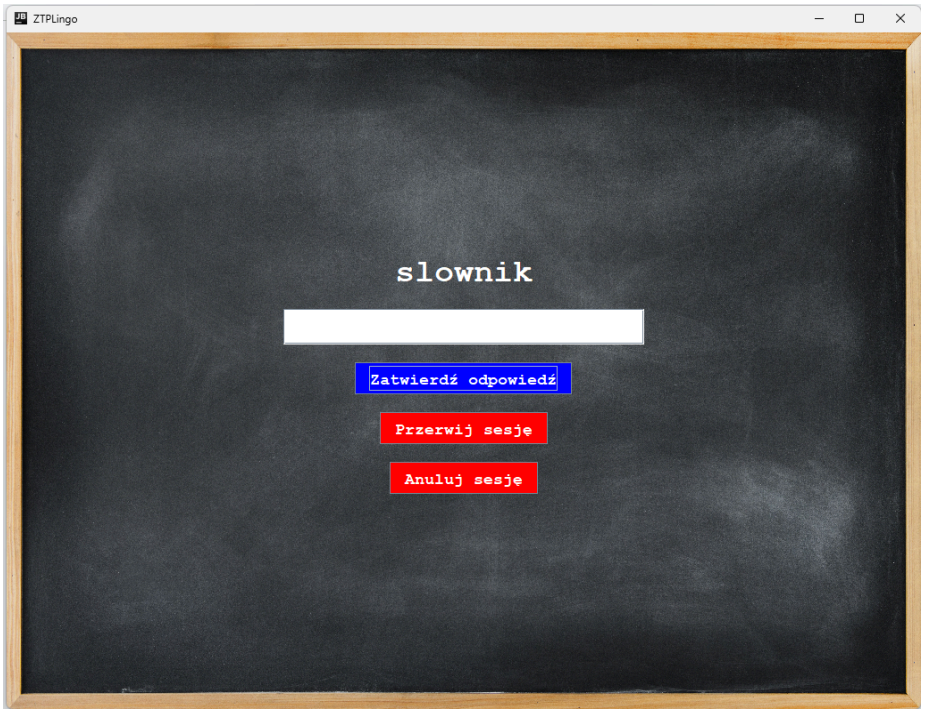
Przed przejściem do quizu (zarówno w trybie testu jak i nauki) użytkownik proszony jest o ustawienie poziomu trudności, trybu językowego oraz ilości pytań.

Quiz o poziomie łatwym:



W trybie łatwym użytkownik składa frazy z rozsypanek poprzez klikanie przycisków myszką. Jeżeli popełni błąd, może usuwać dodane sylaby/słowa przyciskiem *Cofnij*, który cofa kolejno wprowadzane sylaby. Przycisk *Anuluj sesję* wychodzi z quizu i nie zapamiętuje go. Przycisk *Przerwij sesję* wychodzi z quizu i umożliwia wznowienie go z poziomu głównego menu.

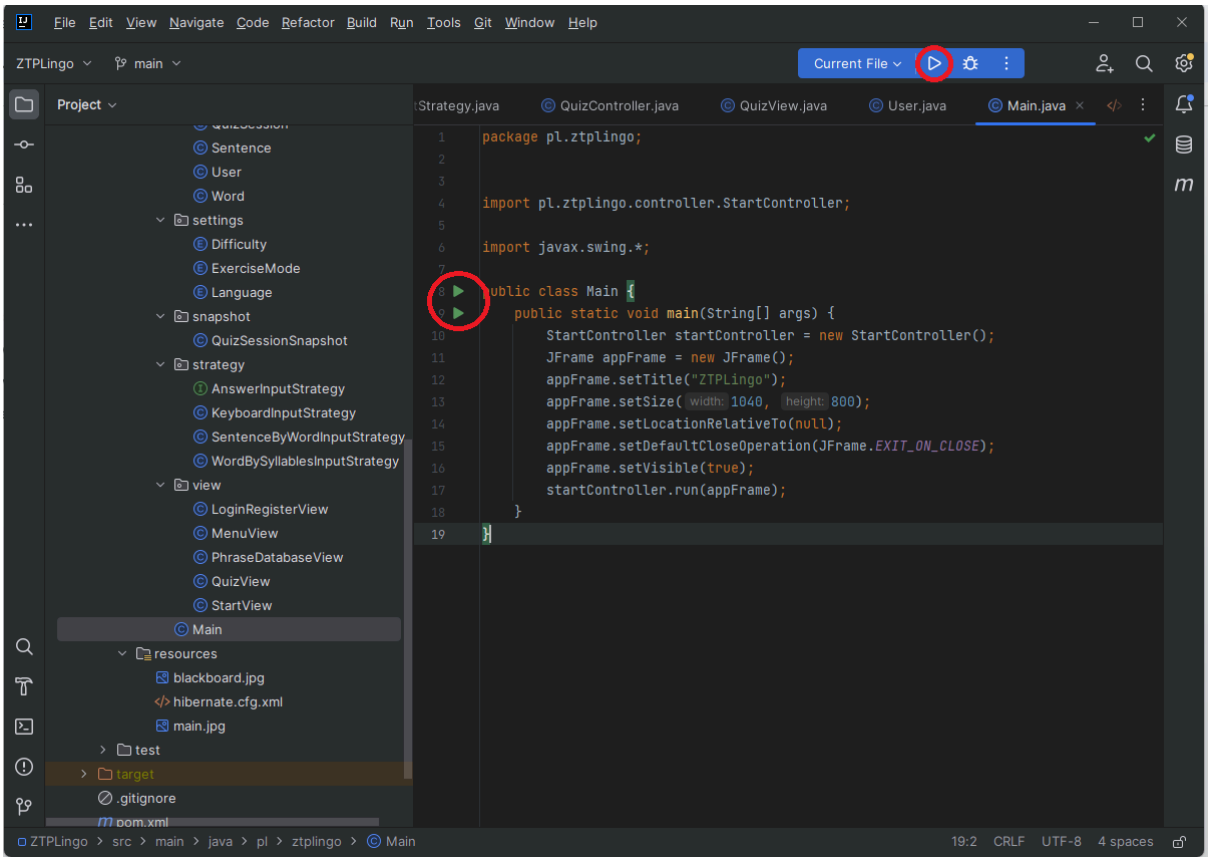
Quiz o poziomie trudnym:



Jedyną różnicą między poziomem łatwym, a trudnym jest sposób odpowiadania na pytania - w tym przypadku, wpisując frazy z klawiatury

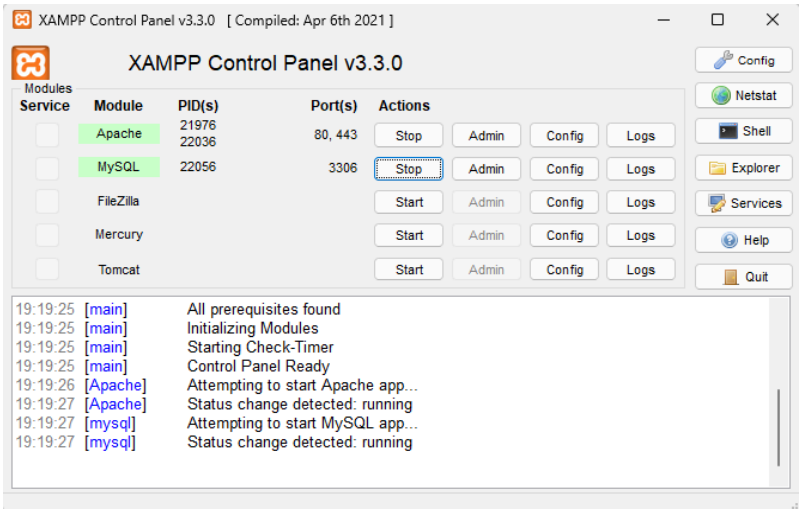
6. Instrukcja instalacji

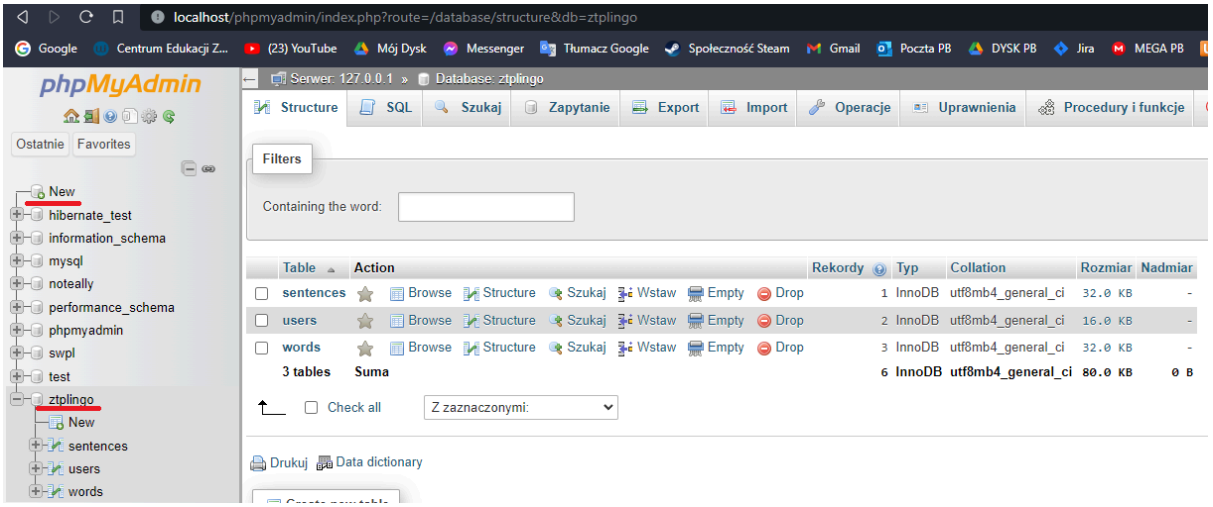
Najlepszym sposobem uruchomienia programu jest otwarcie projektu w środowisku JetBrains IntelliJ IDEA, a następnie uruchominie go z poziomu klasu *Main*.



<u>WAŻNE:</u>

Do działania projektu musi być uruchomiona baza danych MySQL o nazwie “ztplingo” o adresie ip <u>localhost:3306</u>. Program jest przygotowany pod działanie z bazą danych postawioną na serwerze, lecz do celów developerskich użyliśmy bazy na maszynie lokalnej. Aby utworzyć taką bazę najlepiej pobrać program XAMPP i z jego poziomu uruchomić usługę MySQL, a następnie w panelu phpmyadmin utworzyć bazę danych.




localhost/phpmyadmin/index.php?route=/database/structure&db=ztplingo
Google
Centrum Edukacji Z...
(23) YouTube
Mój Dysk
Messenger
Tłumacz Google
Społeczność Steam
Gmail
Poczta PB
DYSK PB
Jira
MEGA PB
phpMyAdmin
Ostatnie
Favorites
New
hibernate_test
information_schema
mysql
noteally
performance_schema
phpmyadmin
swpl
test
ztplingo
New
sentences
users
words
Serwer: 127.0.0.1 » Database: ztplingo
Structure
SQL
Szukaj
Zapytanie
Export
Import
Operacje
Uprawnienia
Procedury i funkcje
Filters
Containing the word:
Table
Action
Rekordy
Typ
Collation
Rozmiar
Nadmiar
sentences Browse Structure Szukaj Wstaw Empty Drop 1 InnoDB utf8mb4_general_ci 32.0 KB -
users Browse Structure Szukaj Wstaw Empty Drop 2 InnoDB utf8mb4_general_ci 16.0 KB -
words Browse Structure Szukaj Wstaw Empty Drop 3 InnoDB utf8mb4_general_ci 32.0 KB -
3 tables Suma 6 InnoDB utf8mb4_general_ci 80.0 KB 0 B
Check all
Z zaznaczonymi:
Drukuj
Data dictionary