|  |  |  |  |
| --- | --- | --- | --- |
| **Wydział Matematyki Stosowanej**  **Politechnika Śląska** | | | |
| **Kierunek:** | | | Informatyka |
| **Stopień studiów:** | | | pierwszy |
| **Semestr:** | | | IV |
| **Przedmiot:** | | | Programowanie obiektowe i graficzne |
| **Prowadzący zajęcia:** | | | dr inż. Adam Zielonka |
| **Dokumentacja techniczna**  **realizacji projektu zespołowego** | | | |
| **Zespół projektowy** | | | |
| **Lider zespołu:** | | Jakub Rup e-mail | |
| **Wykonawca 1:** | | Ewa Szklanny | |
| **Projekt** | | | |
| **Tytuł projektu:** | MTGDeckCreator | | |
| **Repozytorium:** | <https://github.com/DEVDante/MTGDeckCreator> | | |
| **Cel projektu:** | Celem projektu jest stworzenie programu umożliwiającego konstruowanie poprawnych talii do gry Magic: The Gathering. | | |
| **Opis projektu:** | Projekt składa się z bazy danych zawierającej dostępne karty kolekcjonerskie oraz programu umożliwiającego przeglądanie kart i wybór tych, które mają wchodzić skład talii. Poprawność talii jest weryfikowana przez program. | | |
| **Użyte narzędzia** | | | |
| Nazwa: | Visual Studio 2015 Enterprise | | |
| Licencja: | Dream Spark | | |
| Nazwa: | gitHub | | |
| Licencja | open-source | | |

**Historia zmian dokumentu**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| LP | Opis zmiany | Typ | Data zmiany | Autor | Uwagi |
| 1 | Przygotowanie szablonu dokumentacji technicznej | E | 24.05.2016 | Adam Zielonka | Proszę nie zmieniać wielkości czcionek  w dokumencie |
| 2 | Wypełnienie strony tytułowej | D | 27.05.2016 | Ewa Szklanny |  |
| 3 | Dodanie opisu projektu | D | 30.05.2016 | Ewa Szklanny |  |
| 4 | Dodanie modelu bazy danych | D | 10.06.2016 | Jakub Rup |  |
| 5 | Dodanie klas | D | 15.06.2016 | Jakub Rup |  |
| 6 | Dodanie informacji w punktach 2 i 3 | D | 20.06.2016 | Ewa Szklanny |  |
| 7 | Edycja punktów 2, 3 | EM | 22.06.2016 | Jakub Rup |  |
| 8 | Edycja klas | E | 23.06.2016 | Ewa Szklanny |  |

Typ: E - edycja dokumentu, D - dodanie nowych treści, M - korekta merytoryczna

**1. Opis projektu**

Projekt ma na celu opracowanie aplikacji umożliwiającej przeglądanie bazy danych kart gry kolekcjonerskiej Magic: The Gathering, tworzenie i edytowanie talii do tejże gry oraz możliwość ich eksportu oraz importu do formatów wykorzystywanych przez programy do granie w tą grę.

**2. Wymagania**

2.1 Wymagania funkcjonalne

Program ma umożliwiać użytkownikowi przeglądanie dostępnych w bazie kart, a następnie wybór kart, które będą tworzyć jego talię. Program analizuje poprawność talii i informuje użytkownika o niemożności przeprowadzenia pewnych operacji. Użytkownik ma możliwość śledzenia stanu talii poprzez obserwacje jej statystyk. Dodatkowo będzie możliwy podgląd graficzny karty.

W czasie pracy użytkownik ma możliwość zapisu projektu a następnie otworzenie go do dalszej pracy, a po zakończeniu pracy i uzyskaniu poprawnej talii może wyeksportować ją do formatów używanych przez programy pozwalające na grę daną talią (takich jak: Magic Online, Xmage).

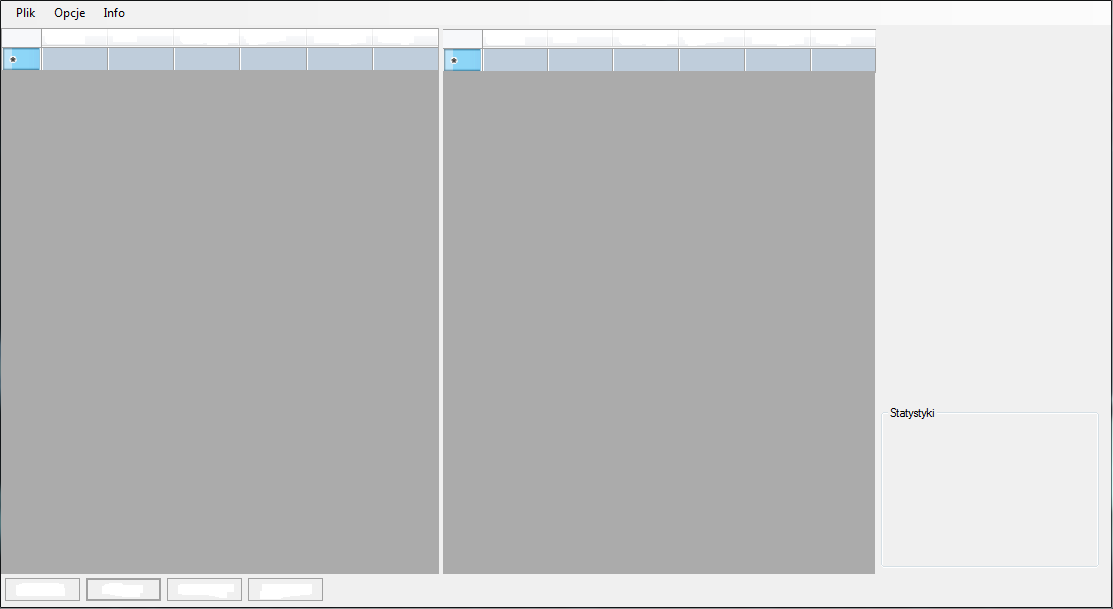
2.2 Wymagania niefunkcjonalne

Aby zmniejszyć obciążenie, obrazek z bazy przy pierwszym otwarciu zapisywany jest w folderze tymczasowym po id karty.

Program komunikuje się z bazą tylko, kiedy potrzebuje otrzymać dane.

Możliwość otrzymania danych o danej karcie tylko znając jej nazwę.

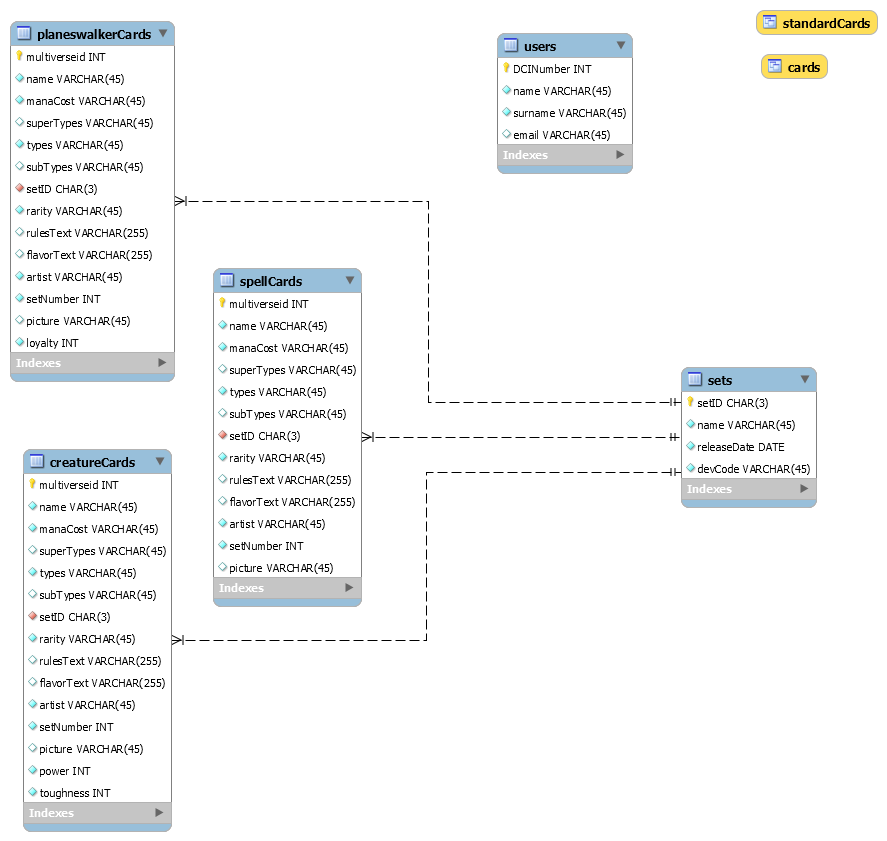
**3 Projekt interfejsu graficznego**



Okno programu będzie podzielona na kilka sekcji. U samej góry znajduje się pasek zadań, na samym dole pasek przycisków funkcyjnych naszego programu (takich jak dodaj, usuń). W środkowej sekcji widzimy dwa ciemnoszare panele. Lewy będzie przechowywał listę wszystkich dostępnych kart, natomiast prawy karty wybrane przez użytkownika, tworzące talię. Jasny panel po prawej stronie okna będzie odpowiedzialny za wyświetlanie statystyk talii, oraz obrazka wybranej karty.

**4. Model obiektowy**

*Schemat UML klas*

**5. Model danych (w przypadku aplikacji bazodanowej)**

**6. Implementacja**

|  |  |
| --- | --- |
| **Klasa** | SpellCard |
| **Właściwości** | |
| string Name | Nazwa karty |
| string Set | Trzy literowy identyfikator dodatku zawierającego kartę |
| string[] SuperTypes | Super typy karty |
| string[] Types | Typy karty |
| string[] SubTypes | Podtypy karty |
| string Rarity | „Rzadkość” karty |
| string RulesText | Tekst z zasadami karty |
| string Flavor | „Flavor” tekst karty |
| string Artist | Imię i nazwisko ilustratora |
| int Number | Numer karty w dodatku |
| int MultiverseID | Unikalny numer id karty |
| string Picture | Ścieżka do pliku z obrazkiem karty |
| char[] ManaCost | Tablica symboli many wymaganej do zagrania karty |
| **Metody publiczne** | |
| byte calculateCMC | Oblicza i zwraca całkowity koszt many karty na podstawie ManaCost |

|  |  |
| --- | --- |
| **Klasa** | CreatureCard |
| **Właściwości** | |
| int Power | Siła potwora |
| int Toughtness | Punkty życia potwora |
| **Metody publiczne** | |

|  |  |
| --- | --- |
| **Klasa** | PlaneswalkerCard |
| **Właściwości** | |
| int Loyalty | Punkty lojalności „planeswalker’a” |
| **Metody publiczne** | |

|  |  |
| --- | --- |
| **Klasa** | DeckFile |
| **Właściwości** | |
| String Filename | Ścieżka do pliku z talią |
| **Metody publiczne** | |
| List<Pair<int, string>> load | Wczytuję talię z pliku i zwraca listę par (ilość karty i jej nazwę) |
| void save | Zapisuje talię przekazaną, jako parametr do pliku Filename |

|  |  |
| --- | --- |
| **Klasa** | DCKDeckFile |
| **Właściwości** | |
| String Filename | Ścieżka do pliku z talią .DCK |
| **Metody publiczne** | |
| List<Pair<int, string>> load | Wczytuję talię z pliku .DCK i zwraca listę par (ilość karty i jej nazwę) |
| void save | Zapisuje talię przekazaną, jako parametr do pliku Filename |

|  |  |
| --- | --- |
| **Klasa** | TXTDeckFile |
| **Właściwości** | |
| String Filename | Ścieżka do pliku z talią .TXT |
| **Metody publiczne** | |
| List<Pair<int, string>> load | Wczytuję talię z pliku .TXT i zwraca listę par (ilość karty i jej nazwę) |
| void save | Zapisuje talię przekazaną jako parametr do pliku Filename |

|  |  |
| --- | --- |
| **Klasa** | Deck |
| **Właściwości** | |
| List<Pair<int, SpellCard>> CardsList | Lista par (ilość karty i jej nazwa) |
| int CardsCount | Zwraca liczbę kart w talii |
| **Metody publiczne** | |
| void addCard | Dodaję kartę SpellCard do talii |
| bool countUpgrade | Zwiększa ilość kopii danej karty w talii |
| void deleteCard | Usuwa kartę z talii |
| List<int> calculateManaCurve | Oblicza ilość kart dla każdego całkowitego kosztu many kart w talii zwracając je, jako listę integer’ów |
| int calculateOtherCount | Oblicza liczbę kart zaklęć, artefaktów |
| int calculateCreatureCount | Oblicza liczbę kart potworów |
| calculateLandCount | Oblicza liczbę kart lądów |

|  |  |
| --- | --- |
| **Klasa** | SQLDataExchange |
| **Właściwości** | |
| string Server | Adres serwera SQL |
| string Database | Nazwa bazy danych |
| string User | Login użytkownika |
| string Password | Hasło użytkownika |
| **Metody publiczne** | |
| List<SpellCard> GetSpellsList | Pobiera z serwera listę kart zaklęć, artefaktów |
| List<CreatureCard> GetCreaturesList | Pobiera z serwera listę kart potworów |
| List<PlaneswalkerCard> GetPlaneswalkersList | Pobiera z serwera listę kart „planeswalker’ów” |
| List<SpellCard> GetCardsList | Łączy funkcjonalność 3 powyższych funkcji |
| SpellCard GetCardInfo | Zwraca informacje o karcie o danej nazwie podanej jako parametr |

**7. Instrukcja użytkownika**