Dokumentacja PWI

Kyrylo Goroshenko *345615* Olaf Surgut (Kucz) Szymon Kuczyński Lidia Podoluk

Piotr Wasielewski Łukasz Janicki *346046* Aliaksandr Luksha

22 stycznia 2024

1 Czego dotyczy projekt

W projekcie stworzona została aplikacja do gry w pokera tajskiego. Główna część dokumentacji dotyczy technicznych aspektów aplikacji i jej obsługi. Zasady gry są opisane na końcu jako część dodatkowa.
Aplikacja składa się z dwóch elementów: serwera oraz klienta.

2 Uruchamianie aplikacji

2.1 Uruchamianie serwera

Należy zainstalować wymagane pakiety wykonując w terminalu komendy:
`sudo apt install python3`
`pip install -r requirements.txt`
Następnie należy zmienić domyślne ustawienia serwera.
W pliku `config.py` należy zmienić wartość pola `"host"` na IP serwera (być może lokalne) oraz `"port"` na dowolny wybrany.
Serwer uruchamia się za pomocą komendy
`python3 server.py`

2.2 Uruchamianie klienta

Klient obecnie nie ma połączenia z interfejsem graficznym, więc aby go uruchomić należy wykonać komendę
`python3 client.py`

3 Dokumentacja techniczna

3.1 Folder `logic`

Folder `logic` zawiera kody opisujące zasadniczą logiczną część gry.

3.1.1 `cards.py`

Plik `cards.py` zawiera słowniki opisujące kolory i figury kart oraz prostą klasę `Card` wykorzystywaną w innych kodach.

3.1.2 `player.py`

Plik `player.py` implementuje klasę `Player`, która posiada szereg atrybutów wykorzystywanych przez serwer oraz klienta i potrzebnych do kontrolowania przebiegu gry.

3.1.3 `deck_of_cards.py`

Plik `deck_of_cards.py` implementuje klasę `Deck`, wykorzystywaną do losowania rozdań i kontrolowania ich przebiegu poprzez pamiętanie rozdanych kart.

3.1.4 `setcheck.py`

Plik `setcheck.py` zawiera zestaw funkcji sprawdzających, czy dany układ znajduje się w danym zbiorze kart. W praktyce tym zbiorem zawsze są karty rozdane graczom w turze, przechowywane w zbiorze `dealt` będącym atrybutem klasy `Deck`.

3.1.5 `bids.py`

Plik `bids.py` zawiera słownik możliwych odzywek licytacyjnych w postaci stringów, którym odpowiadają inne stringi będące wywołaniami funkcji z `setcheck.py`.

3.1.6 `game_start.py` i `table.py`

Pliki te implementują klasę `Table`, która przechowuje informacje na temat graczy, odzywek (*bidów*) i talii kart. Posiada również funkcje takie jak rozpoczęcie gry (wykorzystuje do tego kod z `game_start.py`) lub jej zakończenie, dodanie lub usunięcie gracza, przygotowanie i rozpoczęcie następnej tury, funkcję zwracającą identyfikator obecnego gracza (na potrzeby komunikacji klient-serwer). Oprócz tego klasa `Table` posiada także funkcję do składania bidów, która zawiera kod z `check.py`, oraz funkcje wyświetlające historię odzywek i odzywki dostępne dla gracza.

3.2 Foldery `client`, `server` oraz plik `config.py`

3.2.1 `config.py`

W pliku tym znajdują się jedynie informacje na temat IP hosta oraz portu. Wartości te należy zmienić odpowiednio dla użytkownika, w zależności od tego, gdzie chce się postawić serwer.

3.2.2 `client`

Jedynym elementem zawartości tego folderu jest plik `client.py`. Zawiera on szereg funkcji, które komunikują się z serwerem i zwracają potrzebne informacje z odpowiedzi serwera. Plik ten posiada także prosty interfejs terminalowy służący do testowania komunikacji klient-serwer.

3.2.3 `server`

Jedynym elementem zawartości folderu `server` jest plik `server.py`, który korzysta z Flaska, aby utworzyć serwer do gry. Serwer ten posiada bazę danych graczy i stołów (odpowiednio `PLAYER_DB` i `TABLE_DB`). Korzysta z dekoratorów, aby odpowiednio przekierowywać przychodzące wywołania z klienta. Funkcje w `server.py` dotyczą tworzenia i usuwania stołów, dołączania i odchodzenia od nich, rozpoczynania i kończenia gry oraz składania odzywek. Oprócz tego istnieją również funkcje zwracające informacje na temat stołu (lub stołów), graczy oraz ich nicków, a także funkcja pingująca, która sprawdza połączenie między klientem i serwerem, a w razie potrzeby usuwa nieaktywnych graczy.
W celu realizacji tych funkcjonalności, funkcje serwera korzystają z odpowiednich metod klas `Table` lub `Player`.

3.3 Folder `textures`

Folder `textures` zawiera tekstury wykorzystywane przez interfejs graficzny gry w formacie `.png`. Są to tekstury tła, kart oraz przycisków (np. check). Znajduje się tu też jeszcze folder `kra-files`, który zawiera jedynie plik `all-cards.kra`.

3.4 Inne pliki w głównym folderze projektu

3.4.1 `info_o_grze.py`

Kod ten zawiera jedynie prostą klasę `GameInfo` służącą do przechowywania informacji o stanie gry na potrzeby innych programów.

3.4.2 `zmienne.py`

W tym kodzie znajduje się szereg zmiennych wyznaczających szerokość i wysokość ekranu i stołu, środek i róg stołu oraz niektóre kolory.

3.4.3 `przycisk.py`

Znajduje się tutaj klasa `Przycisk`, która zawwiera metody rysujące przycisk na ekranie oraz przetwarzające interakcje z myszką (kliknięcie).

3.4.4 `menu.py`

Kod ten implementuje prostą klasę `Menu`, która odpowiada za wyświetlanie menu głównego gry.

3.4.5 `pregame.py`

Ten kod odpowiada za wyrenderowanie interfejsu graficznego przed rozpoczęciem gry przy stole. Zawiera klasę `preGame`, która posiada metody umożliwiające rysowanie (i. e. pokazywanie na ekranie) listy stołów, ekranu oczekiwania na start oraz startowania gry, w tym wprowadzania nicku gracza. Oprócz tego w klasie tej zawarte są też metody przetwarzające kliknięcia (oraz puszczenia i ruchy myszką z technicznego punktu widzenia, jednak metody te zawierają jedynie polecenie `pass`).

3.4.6 `dostepne_bidy.py` i `printbids.py`

W tych plikach znajdują się funkcje służące do wyświetlania dostępnych odzywek na ekranie oraz przetwarzania kliknięć w zależności od potrzeby.

3.4.7 `rozdanie.py`

Kod ten implementuje klasę `Rozdanie`, która odpowiada za animację tasowania i rozdawania kart przy stole.

3.4.8 `rozgrywka.py`

Kod zawarty w tym pliku opisuje klasę `Rozgrywka`, której metody odpowiadają za renderowanie obrazu gry, w tym wyświetlanie kart i informacji na temat odzywek licytacyjnych - zarówno złożonych, jak i możliwych do złożenia - oraz za interakcje ruchów myszki lub kliknięć z grą.

3.4.9 `main.py`

Jest to główny kod interfejsu graficznego gry. Zawiera klasę `Gra`, która posiada metody obsługujące wpisywanie tekstu oraz ruchy myszką, zmieniające odpowiednio stan gry. Znajduje się tu też pętla nieskończona, która sprawdza stan gry i wywołuje odpowiednie metody z klas `Rozdanie`, `preGame`, `Rozgrywka` lub innych zaimplementowanych. Klasa ta odpowiada także za komunikację między klientem a interfejsem graficznym gry.

4 Dokumentacja użytkowa

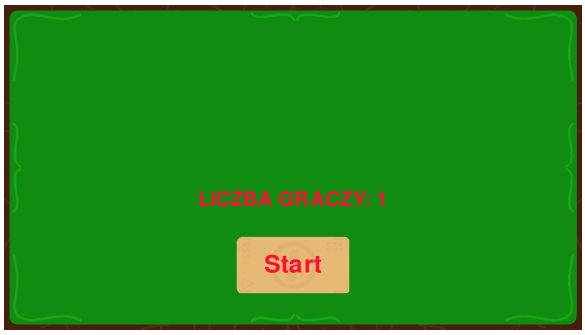
4.1 Ekran startowy

Rysunek 1: Ekran startowy

Ekran startowy zawiera jedynie jakże ujmującą grafikę oraz przycisk Start. Po jego naciśnięciu pojawia się ekran proszący o wpisanie nicku gracza. Po zatwierdzeniu go klawiszem Enter pojawią się opcje “Stwórz” i “Dołącz”.

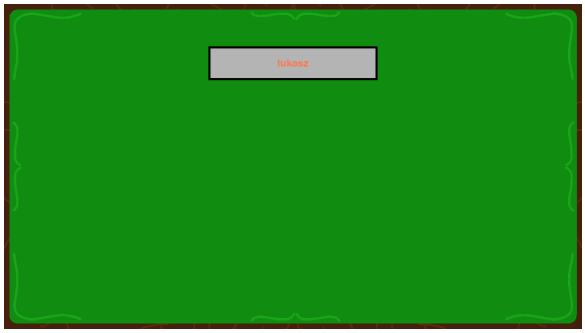
Rysunek 2: Widok po wprowadzeniu nicku

Po wybraniu opcji “Stwórz” pojawi się ekran z liczbą graczy obecnie znajdujących się przy stole oraz przycisk umożliwiający rozpoczęcie gry. Wyświetlana liczba oczywiście zmienia się zależnie od ilości graczy.



Rysunek 3: Ekran oczekiwania z opcją rozpoczęcia

Jeśli wybrano opcję “Dołącz”, pojawi się lista dostępnych na danym serwerze stołów.

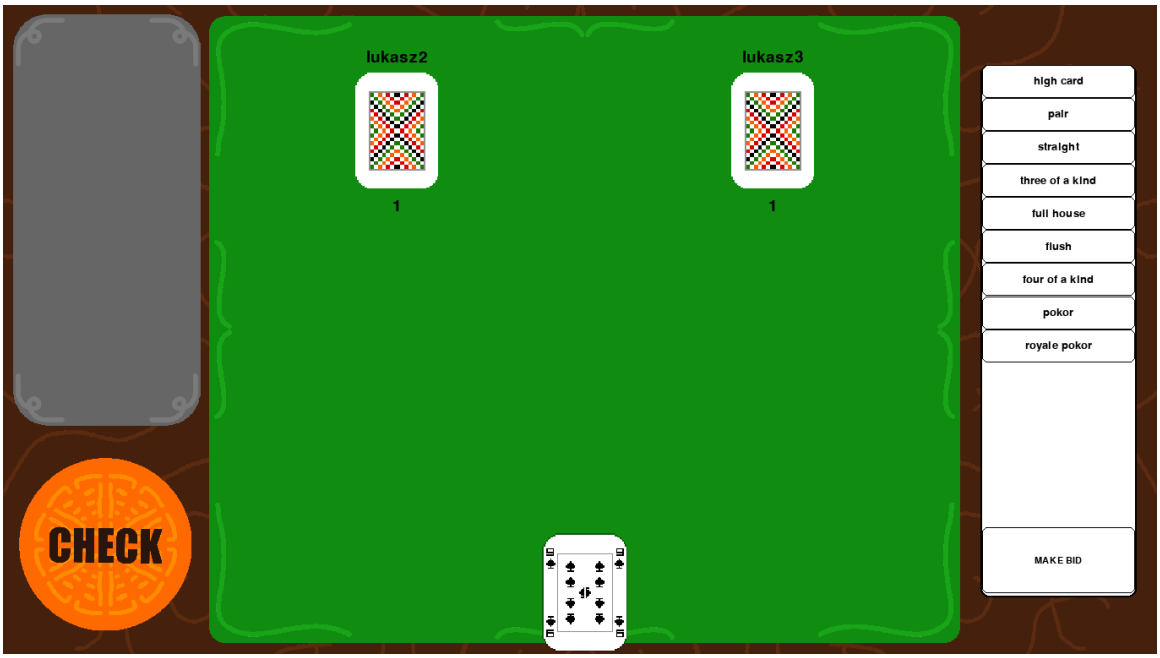


Rysunek 4: Lista dostępnych stołów

Wtedy po wybraniu stołu pojawia się ekran wyświetlający liczbę graczy przy stole i informujący o konieczności czekania.

Rysunek 5: Ekran oczekiwania bez opcji rozpoczęcia

Jeśli gracz, który stworzył stół, kliknie Start, wówczas włączy się animacja tasowania i rozdawania kart, po czym rozpocznie się gra. Interfejs wygląda blisko identycznie niezależnie od tego, czy jest to kolej danego gracza, czy nie - łatwo jednak rozpoznać, czyja jest to kolej, po obecności pomarańczowego przycisku CHECK w lewym dolnym rogu, który na ekranie danego gracza pojawia się tylko wtedy, kiedy jest jego kolej.



Rysunek 6: Ekran gry - kolej danego gracza

W miarę gry, na szarym tle po lewej stronie będą pojawiać się złożone wcześniej odzywki. Umożliwia to śledzenie przebiegu gry.
Gra toczy się, dopóki któryś z graczy nie naciśnie przycisku CHECK, tym samym sprawdzając odzywkę poprzedniego gracza. Przycisk ten nie zareaguje na kliknięcie, jeśli żadna odzywka nie została podczas rundy złożona - w praktyce oznacza to, że gracz rozpoczynający rundę nie może od razu kliknąć CHECK.
Po pomyślnym (tzn. uznanym przez grę jako prawidłowe) sprawdzeniu, gracz, który przegrał, otrzyma komunikat "Przegrałeś Synu!", a reszta - "Przegrał [*nick*]", gdzie *nick* jest, rzecz jasna, nickiem gracza, który właśnie przegrał.

Rysunek 7: Widok przegranej danego gracza

Rysunek 8: Widok przegranej innego gracza

Komunikaty te będą się wyświetlały dopóki nie zostanie złożona odzywka w nowej rundzie. Również liczby kart (cyfry widoczne pod kartami przeciwników) zmienią się odpowiednio, tzn. liczba kart przegranego zwiększy się o 1.
Na Rys. 8 widać również, jak wygląda składanie odzywek - należy kliknąć w typ odzywki, który interesuje gracza, co odsłoni listę dostępnych. Następnie należy kliknąć konkretną odzywkę, która podświetli się na różowo, oraz kliknąć "MAKE BID". Odzywki niedostępne, choć będzie można zobaczyć ich listę, będą wyświetlały się na szaro i nie będą reagować na klikanie. Jeśli gracz się rozmyślił i chce zmienić typ składanej odzywki, musi jeszcze raz kliknąć listę, aby ją najpierw zwinąć.

W momencie *przegranej* któregokolwiek z graczy, czyli jeśli ktokolwiek przekroczy dopuszczalną liczbę posiadanych kart, gracz ten zostaje usuwany z gry - nie są mu już rozdawane karty, choć może nadal przebywać przy stole i obserwować dalszy przebieg rozgrywki. Dla innych graczy jest on nadal widoczny, lecz liczba jego kart widnieje jako 0.

Rysunek 9: Widok gracza który odpadł z gry