# Sprawozdanie

Mateusz Dębski I4.1 INF148216

### **Gra internetowa Warcaby**

### Opis protokołu komunikacyjnego:

Protokół komunikacyjny opiera się na wymianie komunikatów, zawierających aktualny stan planszy, między klientami. Klienci komunikują się ze sobą za pośrednictwem serwera.

Po wysłaniu komunikatu do klienta, serwer oczekuje odpowiedzi potwierdzającej "o.", z wyjątkiem komunikatów oznaczających zakończenie połączenia: f. , v. , l. ,q.

### Rodzaje komunikatów:

- c. udane połączenie
- f. serwer jest pełny
- w. twój kolor to biały
- b. twój kolor to czarny
- o. oczekiwanie na wykonanie ruchu przez przeciwnika albo aż przeciwnik dołączy do gry
- j. przeciwnik dołączył do gry, koniec oczekiwania
- q. przeciwnik wyszedł z gry
- m[plansza]. wykonaj ruch
- v[plansza]. koniec gry wygrałeś
- I[plansza]. koniec gry przegrałeś

Plansza jest reprezentowana w postaci ciągu znaków np. początkowa plansza:

### Oznaczenia w planszy:

- x puste pole,
- w biały pionek,
- b czarny pionek,
- d biała damka,
- u czarna damka

## Opis implementacji:

#### Server

Implementacje serwera można znaleźć w pliku server.cpp. Współbieżność serwera zrealizowana przy pomocy wątków (każdy gracz to osobny wątek).

#### **Klient**

Odpowiada za interfejs(stworzony za pomocą biblioteki PyGame)

Implementacja klienta jest dużo bardziej złożona, składa się z plików:

- board.py
   odpowiada za planszę, znajdują się tam funkcje niezbędne do narysowania planszy,
   walidacja ruchów odbywa się poprzez znalezienie możliwych ruchów i pokazanie ich na
   ekranie użytkownikowi.
- Piece.py
   odpowiada za pionka, znajdują się tam metody odpowiedzialne za jego rysowanie, oraz
   wyliczające współrzędne potrzebne do narysowania
- button.py
  Klasa odpowiadająca za przyciski i komunikaty wyświetlane na ekranie
- client.py
  Łączy wszystko w całość, odpowiada za obsługę servera, wyświetla odpowiednie ekrany
- W folderze assets znajdują się obrazy potrzebne do rysowania pionków oraz przycisków

### Opis uruchomienia:

- 1. Aby uruchomić projekt należy ściągnąć projekt (trzeba zachować strukurę taką samą jak na gitlabie).
- 2. W razie potrzeby zainstalować pythona, oraz bibliotekę PyGame. instalacja:

### pip install pygame

3. Server należy skompilować poprzez polecenie

```
g++ -pthread -Wall server.cpp -o server
```

4. Uruchamianie serwera:

./server

5. Uruchamianie klienta:

python client.py <ip\_servera>

Po uruchemieniu klienta naszym oczom ukaże się ekran z przyciskiem connect.

Po naciśnięciu przycisku rozpoczniemy rozgrywkę.

Aby wykonać ruch należy wybrać pionek (dwukrotnie na niego klikając) a następnie wybrać pole na które chcemy go przesunąć.