PROJEKT ZPR - GRA W STATKI
DOKUMENTACJA KOŃCOWA

Katarzyna Dziewulska
Daniel Giełdowski
Mateusz Morusiewicz

Prowadzący : dr hab. inż. Robert Nowak

1.Temat projektu:

Tematem projektu jest dwuosobowa gra w statki z interfejsem na przeglądarkę, wykonana w architekturze klient-serwer.

2.Dokumentacja użytkownika:

Po uruchomieniu gry w przeglądarce internetowej pojawia się okno rejestracji i logowania. Jeśli użytkownik po raz pierwszy rozpoczyna grę wpisuje login i hasło, których będzie używał w następnych logowaniach, oraz wciska przycisk Register. W następnych logowaniach robi to samo, lecz zatwierdza przyciskiem Login. Następnie pojawia się okno wyboru nowej lub już istniejącej gry, w której użytkownik oczekuje na współgracza. W nim również na samej górze wyświetlają się statystyki (ilość wygranych gier). Po wyborze gry pojawia się okno gry z dwoma planszami. Po lewej stronie znajduję się plansza ze statkami gracza, po prawej zaś plansza z jego strzałami. Jeśli użytkownik oczekuje na współgracza, plansza po prawej jest koloru szarego a po lewej białego i nie można wykonać żadnych akcji do czasu pojawienia się przeciwnika. Kiedy w grze jest dwóch graczy plansza po lewej jest koloru niebieskiego z ustawionymi na niej statkami koloru czerwonego. Kiedy przeciwnik trafi w statek jest to zaznaczone czarny kolorem. Plansza po prawej jest koloru niebieskiego u gracza, który ma aktualnie prawo ruchu. Jego przeciwnik ma zablokowany ruch, a jego plansza ma kolor szary. Wykonanie ruchu "strzału" odbywa się przez kliknięcie myszką w wybrane pole na planszy strzałów (tej po prawej stronie) o ile nie było wcześniej takiego ruchu. Strzały trafione zaznaczane są czerwonym kolorem, natomiast strzały chybione czarnym. Gracz ma określony czas na wykonanie ruchu i jest to 30 sekund. Ilość czasu pozostałego do końca ruchu wyświetla się pod planszami. W przypadku wygrania gry - zestrzelenia wszystkich statków przeciwnika pod planszami wyświetla się komunikat czy gracz wygrał czy

przegrał rozgrywkę oraz statystyki. Jeśli użytkownik nie wykona ruchu w ciągu 30 sekund to przegrywa rozgrywkę.

3.Zrealizowana funkcjonalność:

- Po pierwszym uruchomieniu gry użytkownik zakłada swoje konto (wybiera nieużywany przez innych graczy nick i hasło), w następnych uruchomieniach zgłasza chęć gry logując się na swoje konto.
- Statki ustawiane są automatycznie.
- Po rozpoczęciu rozgrywki podczas ruchu jednego gracza, możliwość ruchu przeciwnika jest zablokowana.
- Niedozwolone ruchy gracza są zablokowane (np. dwukrotne zatopienie tego samego okrętu).
- Czas ruchu gracza jest ograniczony. Po przekroczeniu czasu gracz przegrywa rozgrywkę.
- Jeśli jeden z graczy w trakcie rozgrywki zrezygnuje z gry lub zamknie przeglądarkę to przegrywa rozgrywkę.
- Użytkownik ma możliwość podejrzeć statystyki swoich dotychczasowych rozgrywek.
- Po zakończonej rozgrywce możliwa jest kolejna z tym samym graczem za jego zgodą, bądź rozpoczęcie rozgrywki z nowym graczem.

4.Technologie które zostały użyte:

Klient

Jako że klient wykorzystuje jedynie przeglądarkę www do napisania interfejsu graficznego użytkownika został użyty JavaScript +CSS+HTML5.

Serwer

Serwer działający na systemie Linux oraz Windows został utworzony przy użyciu: lighttpd (serwer www udostępniający stronę z aplikacją), Python (serwer aplikacji), boost::python, C++ (logika aplikacji). Informacje o dotychczasowych rozgrywkach gracza są przechowywane w bazie danych po stronie serwera.

Komunikacja klient - serwer została zrealizowana przy użyciu środowiska bioweb.

System kontroli wersji
Tworzenie aplikacji ułatwiło nam użycie systemu kontroli wersji git w serwisie github (https://github.com/Katarzyna9527/Projekt_ZPR).

5. Uruchamianie i kompilacja

Wszystkie potrzebne biblioteki i komponenty potrzebne do skompilowania i uruchomienia gry są wyszczególnione w pliku README_EN środowiska bioweb.
Aby uruchomić grę należy najpierw skompilować źródła poleceniem scons w wierszu poleceń, znajdując się w katalogu gry, a następnie uruchomić grę komendą scons r=l.

Nie byliśmy w stanie sprawdzić czy gra jest kompilowana i uruchamiana poprawnie na systemie Windows, z powodu braku możliwości zainstalowania oprogramowania lighttpd na tym systemie. Linki do pobrania odwołują się do nieistniejącej strony.

6. Testowanie

Język	Ilość testów	Procentowe pokrycie kodu testami	Komentarz
C++	14	100% funkcji	Logika gry
Python	15 (4 model - kontroler, 11 widok-kontroler)	100% metod publicznych	Serwer i kontroler aplikacji

7. Udział języków:

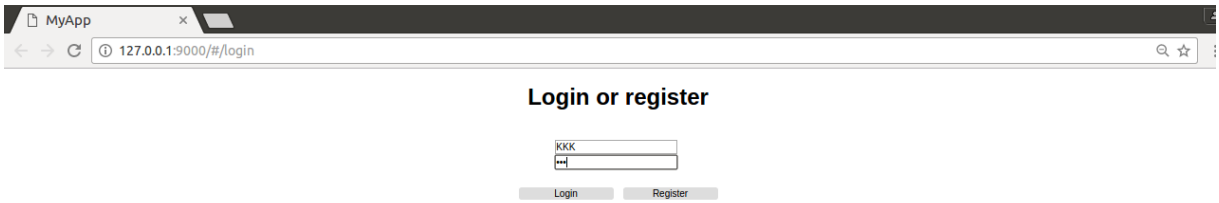
Język	liczba linii kodu
C++	1089 (224 testy)
Python	646 (219 testy)
JavaScript	288
CSS	160
HTML	130

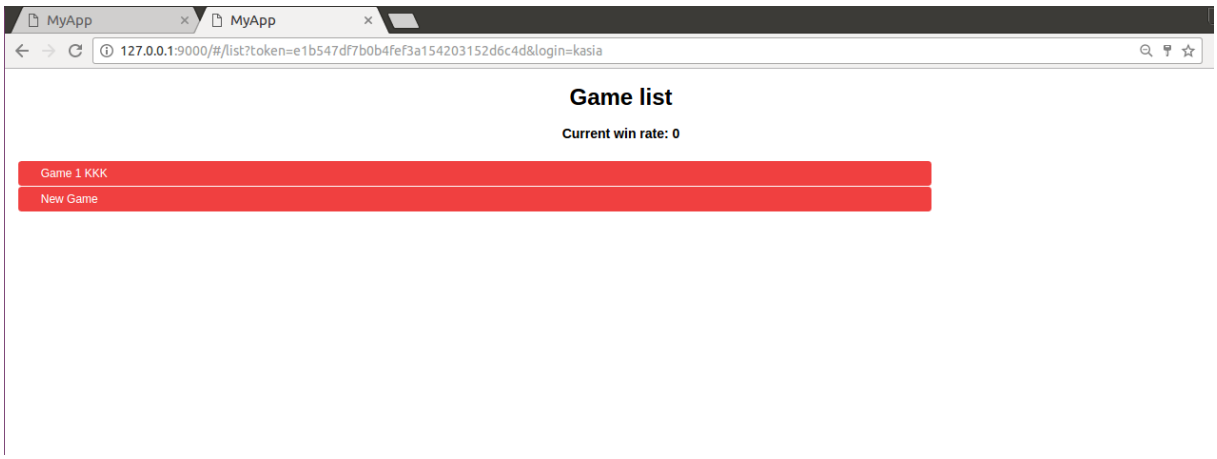
8. Liczba godzin poświęcona na projekt: Każdy z członków zespołu poświęcił na projekt około 200 h.

9. Opis napotkanych problemów i popełnionych błędów

Jednym z problemów było użycie biblioteki boost::python. W szczególności nie wiedzieliśmy jak odczytać dane z vectora vectorów w języku python, jednak doszliśmy do tego metodą prób i błędów. Dużo czasu zajęło nam zdecydowanie za co konkretnie powinien być odpowiedzialny poszczególny moduł gry. Mieliśmy tylko zarys, lecz nie konkretny model aplikacji, co znacznie utrudniło i wydłużyło pisanie projektu.

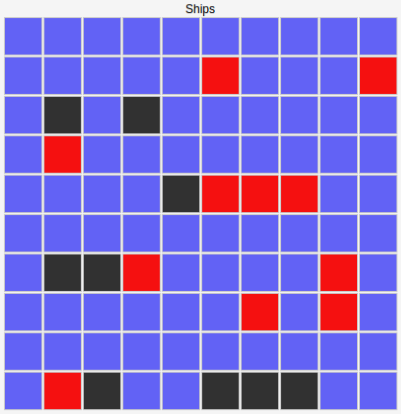
10.Screenshot z gry



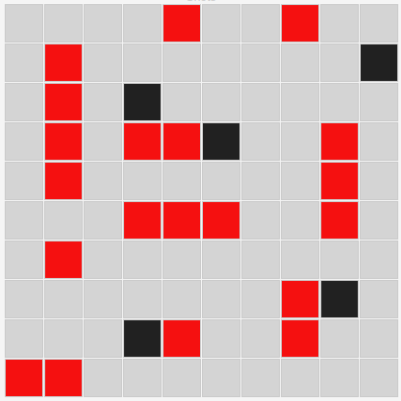


Board Game 2

Ships



Shots



Game won! Congratulations!

Your win rate is 1

Replay