Dokumentacja inżynierii wymagań

Przedmowa

Celem naszego projektu było stworzenie prostej gry (istnieje wiele gier tego typu ale chcemy spróbować naszych sił w stworzeniu własnej wersji) , która dostarcza dobrą zabawę, zwiększa efektywność uczenia się i zapamiętywania. Gra typu „Memory” polega na odkrywaniu i zapamiętywaniu kart w celu zdobycia pary takich samych obrazków. Gra pozwala równocześnie na relaks jak i trening umysłu.

Słownik pojęć technicznych

- **UI:** Komponent umożliwiający interakcję użytkownika z systemem za pomocą przeglądarki internetowej, zaimplementowany przy użyciu HTML, CSS i JavaScript.

Macierz kompetencji:

Kompetencje	Michał Tajak	Michał Żychowicz	Bartosz Bereski
Programowanie JavaScript	**Posiada**	**Posiada(podstawy)**	**Posiada(podstawy)**
Programowanie Node.js	**Posiada**	**Nie posiada**	**Nie posiada**
Programowanie TypeScript	**Posiada (podstawy)**	**Nie posiada**	**Nie posiada**
Programowanie Python	**Posiada**	**Posiada (podstawy)**	**Posiada**
Testowanie oprogramowania	**Posiada (podstawy)**	**Nie posiada**	**Nie posiada**
Programowanie PostgreSQL	**Posiada**	**posiada(podstawy)**	**posiada**
HTML I CSS	**Posiada**	**Posiada**	**Posiada**

Definicja wymagań użytkownika

Działanie gry:

1. Uruchamiamy grę w menu startowym(start gry uruchamia timer).
2. Mamy możliwość zatrzymania gry.
3. Mamy możliwość odkrywania kart każdy użytkownik po jednej próbie.
4. Jeśli gracz odkryje dwie różne karty, zostają one odwrócone do stanu początkowego.
5. Gdy dany gracz odkryje dwie takie same karty w jednej turze otrzymuje punkt.
6. Gra kończy się w momencie braku kart do odkrycia(gdy wszystkie karty są widoczne).
7. Wyświetlane jest podsumowanie gry.

Pytania, które pojawiły się w dyskusji w celu sprecyzowania celu projektu:

Pytanie	Odpowiedz	Uwagi
Ile graczy może wziąć udział w jednej rozgrywce?	Przynajmniej dwóch	Liczba graczy może wzrastać wraz z wielkością planszy
W jaki sposób gracze będą dołączać do rozgrywki?	Na początku każdy gracz będzie podawal swój login i będzie mogl utworzyc lub dołączyć do istniejącego pokoju o konkretnej nazwie	Pokoj nie będzie się wyświetlał w momencie gdy rozgrywka zostanie rozpoczęta
W jaki sposób będą zbierane informacje o rozgrywce?	Po każdej rozgrywce dane z parametrami będą zapisywane do bazy danych	
W jaki sposób będzie można wygenerowac statystki?	W panelu admina będzie możliwość wygenerowania statystyk w postaci wykresów	
Sposób punktowania?	Za każdą zgadniętą parę gracz dostaje 1 punkt	

Szczegóły implementacyjne:

Room	User	Report	
roomID (int unique const)	userID (int unique const)	room name (string)	numOfPlayers (int const
Name (string)	name (string)	game size (string)	correct matches (float -1)

NumOfPlayers (int const)	roomID (int const)	first move (int array [w,d] -> współrzędne)	name Winner (string)
Size (string -> small, mid, big)	score (int)	name user with first move (string)	numOfMoves (int const

MODELOWANIE:

- **Za pomocą tabeli:**
  - Aktorzy -> gracze, analiza danych
  - Opis -> celem projektu jest możliwość zalogowania się, wybrania lub utworzenia pokoju a następnie możliwość zagrania z innymi bądź z botem. Każda rozgrywkę rozpoczyna inny gracz. Po zakończeniu każdy z graczy dostaje wiadomość ze swoim wynikiem i informacje na temat wygranej osoby .
  - Dane -> liczba graczy rozmiar planszy, wynik graczy
  - Wyzwalacz -> po kliknięciu ‘graj’ przez twórcę danego pokoju gra rozpoczyna się jeśli spełnione są wymagania (przynajmniej dwóch graczy w pokoju, liczba graczy dopasowana do planszy, ustawione parametry rozgrywki)
  - Odpowiedz -> po zakończonej rozgrywce wyświetla się plansza informująca i zostanie wygenerowany raport do bazy
  - Uwagi -> każda rozgrywka kończy się w momencie odszukania wszystkich par dla obrazków
  - Gracze -> proces przeprowadzenia rozgrywki -> Analiza danych
- **Diagram użycia:**
  - Zalogowanie się użytkownika
  - Dołączenie lub utworzenie rozgrywki
  - Uruchomienie rozgrywki po spełnieniu wymagań
  - Przeprowadzenie rozgrywki zgodnie z logika gry
  - Losowany jest kolejność w jakiej gracze wykonują ruch
  - Następnie gracz odkrywa dwie losowe karty
  - Jeśli **SĄ** takie same dostaje punkt a karty zmieniają swój kolor i tracą możliwość odkrycia
  - Jeśli nie SĄ takie same zostają ponownie zakryte

- Kolejno wybierany jest następny gracz a czynność zostaje powtórzona
- Gra kończy się w momencie odkrycia ostatniej pary obrazków gdy plansza jest pusta
- **Zakończenie rozgrywki i wyświetlenie wyników oraz wysłanie raportu z rozgrywki do bazy**

- **Diagram sekwencyjny:**
  - Zalogowanie się użytkownika (wygenerowanie danych user w systemie)
  - Dołączenie użytkownika do pokoju (dodanie do danych user id pokoju)
  - Rozpoczęcie rozgrywki (przekazanie danych i parametrów do gry)
  - Przeprowadzenie gry (zmiany w parametrach takich jak liczba ruchów, procent zgadnięć)
  - Zakończenie rozgrywki (przekazanie danych z gry i zwycięzcy do bazy danych)
- **Projekt architektury:**
  - Start – wejście:
    - Użytkownicy
    - Pokoje
    - Gra (wraz z parametrami rozgrywki)
  - Przetwarzanie:
    - Przeprowadzenie rozgrywki zgodnie z logika gry
    - Przetwarzanie zmian w parametrach gry
  - Koniec – wyjście:
    - Informacja o wynikach rozgrywki
    - Generowanie danych z rozgrywki do bazy

Minimum Viable Product (MVP)

Dla gry Memory, znanej także jako Gra Pamięci, to prosty projekt, który zawiera podstawowe funkcje potrzebne do gry w tę popularną grę logiczną. Oto lista funkcji, które można uwzględnić w MVP gry Memory:

1) Interfejs użytkownika:
   - Ekran startowy z opcją rozpoczęcia gry.
   - Plansza do gry o określonym rozmiarze (4x4).
   - Karty do pary, które są ukryte na początku gry
2) Rozgrywka:
   - Gracz może odkrywać dwie karty na raz, klikając na nie.
   - Logika gry, która sprawdza, czy odkryte karty są takie same (jeśli tak, to zostają odsłonięte, jeśli nie, to zostają zakryte)
3) Funkcje dodatkowe:
   - Możliwość zatrzymania gry w dowolnym momencie
   - Licznik ruchów lub czasu gry (opcjonalnie)
4) Zakończenie gry:
   - Ekran końca gry, który wyświetla wynik gracza (liczbę ruchów lub czas).

- Przycisk do ponownego rozpoczęcia gry lub powrotu do ekranu startowego. Grafika i dźwięk:

5) Grafika i dźwięk
    - Proste grafiki kart (np. obrazki lub symbole).

Sugerowane rozwiązania:

- JavaScript
    - Za pomocą tego języka zostanie stworzone GUI całej strony, zostanie zaprogramowana rozgrywka oraz sposób jej działania
    - Uzasadnienie: Wybraliśmy Javascript ze względu na jego **uniwersalność** dodatkowo każdy uczestnik zespołu był zaznajomiony z jego użyciem i wykorzystaniem. Kolejną zaletą było to że jest to **język skryptowy** i ma wsparcie wielu **bibliotek i frameworków**.
- Node.js
    - Będzie odpowiedzialny za komunikację klient Serwer dodatkowo jako Serwer będzie przetwarzał informacje na temat graczy, pokojów i rozgrywki.
    - Uzasadnienie: **Asynchroniczność** pozwalająca na obsługiwaniu wielu klientów jednocześnie. Dodatkowo z wykorzystaniem Node.js możemy wykorzystać samego Js po stronie serwera i klienta.
- Python
    - Będzie odpowiedzialny za przeprowadzenie analityki danych dostarczonych z rozgrywek i przechowywanych w bazie danych, dodatkowo będzie generował raporty w postaci wykresów
    - Uzasadnienie: Łatwość implementacji i bogate narzędzia ułatwiające analizę danych takie jak: **Matplotlib czy NumPy**
- PostgreSQL:
    - Baza danych będzie odpowiedzialna za przechowywanie podsumowań rozgrywek
    - Uzasadnienie: Znajomość przez każdego członka zespołu, postgreSQL obsługuję **transakcję** i **dobrze wydajne**.