

Technologie internetowe w programowaniu

2020/2021

Kamil Wyżgoł, grupa ISI3

Dokumentacja projektu

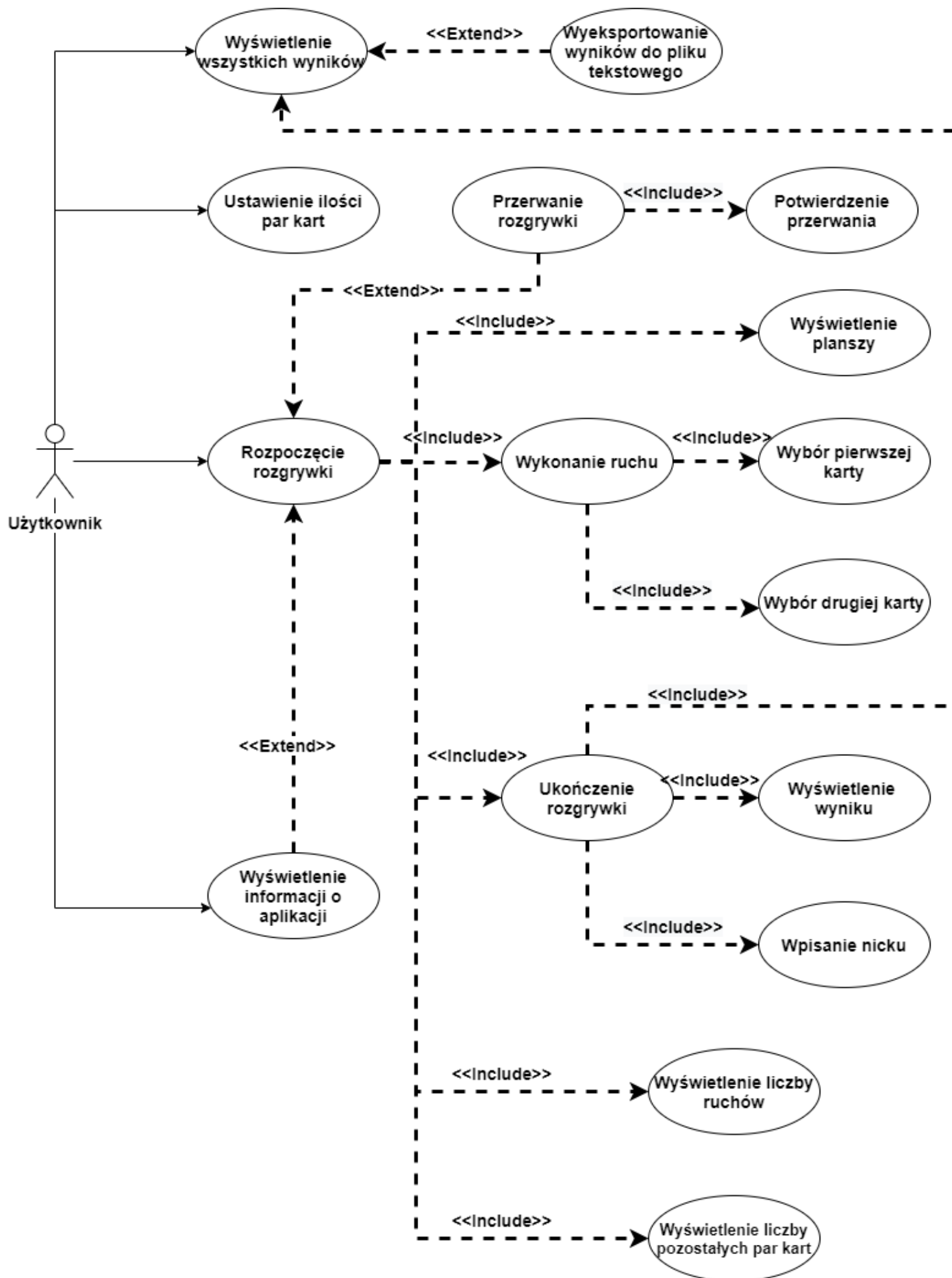
❖ Opis aplikacji

Aplikacja polega na znajdowaniu par takich samych kart przy jak najmniejszej liczbie wykonanych ruchów. Liczbę par można dostosować: minimalna liczba to 2, a maksymalna to 32.

❖ Lista funkcjonalności

- ➔ losowa kolejność kart,
- ➔ możliwość dostosowania ilości par kart (od 2 do 32),
- ➔ rozmieszczenie kart zależne od szerokości okna,
- ➔ obliczanie wyniku według wzoru:
$$\text{Wynik} = (1000 * \{\text{ilość par}\}) + (1000 / \{\text{ilość ruchów}\})$$
- ➔ zapisywanie wyników (Web Storage API – localStorage),
- ➔ wyświetlenie wyników (w postaci posortowanej),
- ➔ animacja odwracania karty,
- ➔ informacja o liczbie wykonanych ruchów,
- ➔ informacja o pozostałej liczbie par kart,
- ➔ możliwość przerywania gry (wymaga potwierdzenia),
- ➔ możliwość odsłonięcia tylko dwóch kart w jednym ruchu,
- ➔ zanikanie znalezionych już par (nie wpływa na ułożenie pozostałych),
- ➔ możliwość wyeksportowania wyników do pliku tekstowego,
- ➔ wyświetlenie informacji o aplikacji.

❖ Diagram przypadków użycia



- ➔ **Wyświetlenie informacji o aplikacji** – tj. nazwa, wersja, opis, autor, informacja o źródle użytych grafik,
- ➔ **Ustawienie ilości par kart** – minimum 2, maksimum 32,
- ➔ **Wyświetlenie wszystkich wyników** – jeżeli są w localStorage (Web Storage API),
 - **Wyeksportowanie wyników do pliku tekstowego** – za pomocą dynamicznie utworzonego elementu <a>,
- ➔ **Rozpoczęcie rozgrywki**,
 - **Wyświetlenie planszy** – losowa kolejność kart, ułożenie zależne od szerokości okna przeglądarki,
 - **Wyświetlenie liczby ruchów**,
 - **Wyświetlenie liczby pozostałych par kart**,
 - **Wykonanie ruchu**,
 - **Wybór pierwszej karty** – zastosowanie odpowiednich animacji oraz oczekiwanie na wybranie drugiej karty,
 - **Wybór drugiej karty** – zastosowanie odpowiednich animacji oraz sprawdzenie poprawności wyboru,
 - **Ukończenie rozgrywki** – wymagane jest odnalezienie wszystkich par,
 - **Wyświetlenie wyniku**,
 - **Wpisanie nicku**,
 - **Przerwanie rozgrywki**
 - **Potwierdzenie przerwania** – za pomocą okna modalnego.

❖ Przykłady fragmentów zastosowanych algorytmów

➔ Losowa kolejność elementów w tablicy

```
for i ← array.length to 0
  do r ← random( i )
    tmp ← array[i]
    array[i] ← array[r]
    array[r] ← tmp
```

➔ Sprawdzenie poprawności wskazania pary kart oraz ewentualne zapisywanie wyniku

```
firstCardValue ← card1.img.alt
secondCardValue ← card2.img.alt
score ← 1000 * numberOfPairs
moves ← moves + 1

if firstCardValue = secondCardValue
  then numberOfPairs ← numberOfPairs - 1
    if numberOfPairs = 0
      then results ← localStorage.getItem( )
        if results = null
          then results ← new Array( )
            score ← score + (1000 / moves)
            score ← round( score )
            nick ← userWrite( )
            results.push( nick, score )
            results.sort( )
            results.reverse( )
            localStorage.setItem( results );
```

Powyższy pseudokod stanowi zapis uproszczony, pominięte zostały m. in. fragmenty dotyczące animacji oraz wyświetlania informacji.