
Eötvös Loránd Tudományegyetem

Informatikai Kar

Algoritmusok és Alkalmazásaik

Tanszék

# Többjátékos kártyajáték keretrendszer

*Témavezető:*
Máriás Zsigmond György
egyetemi óraadó

*Szerző:*
Rubóczki Benedek
programtervező informatikus BSc

*Budapest, 2023*

# Tartalomjegyzék

# 1. fejezet

# Bevezetés

A család valamennyi ember életében központi helyet foglal el. Ezen kötelék közelségének ápolására számos lehetőségünk van, köztük például a közös játék. Így nagyon sokaknak ismerős lehet az ünnepi kártyázás élménye, és az ezzel járó rengeteg saját szabály észben tartásának kihívása.

Sajnos a mai világban egyre nehezebb személyesen összegyűlni. Erre különösen felhívta figyelmünket a 2020-as Covid19-pandémia, melynek hatására kénytelenek voltunk új megoldásokat keresni a mindennapi élettel járó tevékenységek folytatására. A kapcsolattartásra sok megoldással álltak már elő. Több tucatnyi választási lehetőség tárul elénk, ha üzenetváltásról vagy videóhívásról van szó, viszont a közös játékok terén a többség által ismert opciókra vagyunk korlátozva. Könnyen előfordulhat, hogy nem találunk online játszható, többjátékos változatot a kedvelt foglalkozásunkra, vagy ez csak a sajátjainktól különböző szabályokkal érhető el számunkra.

Célom egy olyan webes applikáció elkészítése, mely megoldást nyújt a fentebb említett nehézségekre. Ehhez egy olyan rendszer tervével állok elő, ami lehetőséget biztosít a kedvenc francia kártyával játszott kártyajátékaink elkészítésére és játszására. Fontos szempontként magam elé helyezem a játékélmény folyamatos, gördülékeny menetét, a játékfelület könnyen értelmezhetőségét és használatát. A saját játékok készítése esetén, pedig a szabályrendszer által elegendő lehetőség biztosítását komplexebb szabályok vagy logikai kapcsolatok előállítására.

# 2. fejezet

# Felhasználói dokumentáció

## 2.1. Az alkalmazás rövid ismertetése

A többjátékos kártyajáték keretrendszer egy olyan webes applikáció, ami lehetővé teszi valamennyi francia kártyával játszható kártyajátékok elkészítését és online együtt játszását más játékosokkal.

A játékkészítésre különböző szabályok logikai kapcsolatokkal való összeláncolásával van lehetőség. Az együtt játék pedig bejelentkezés után történhet virtuális szobákban, ahol indítás után a szerkesztés során meghatározott szabályok alapján rakhatnak le kártyákat a játékosok. Játék közben lehetőség van az egyszerű kommunikációra chat ablak és reakció gombok segítségével.

## 2.2. Rendszerkövetelmények

A szerver szoftveres telepítésének követelménye a Node.js[1] futtatási környezet 18.15.0-ás vagy újabb major verzión[2] belüli kiadása.

Ajánlott minimum rendszerkövetelmény:

- Windows operációs rendszeren:
    1. Windows 10 vagy újabb (64 bit).
    2. Intel Core i3 processzor vagy újabb.
    3. 4GB Rendszermemória

[1] A Node.js egy nyílt forráskódú cross-platform JavaScript futtatási környezet.
[2] Jelen esetben ez bármelyik 18.X.X verzió.

- MacOS operációs rendszeren:
    1. macOS Ventura 13 vagy újabb.
    2. Intel Core i5 processzor vagy újabb.
    3. 4GB Rendszermemória

Az applikáció böngészőből elért részéhez ajánlott böngészők: Google Chrome, Safari, Microsoft Edge, Firefox. A megfelelő felhasználói élmény biztosítására az előbb említett böngészők 2020 végén, vagy 2020 után megjelent verziói használatosak.

## 2.3. Telepítés és futtatás

A többjátékos kártyajáték keretrendszer teljes telepítése 5 fő lépésből áll:

1. Supabase projekt felállítása
2. Adatbázis konfigurálása
3. Futtatási környezet telepítése
4. A szerverprogram telepítése
5. Az API-hoz szükséges adatok megadása

Ha az általam készített Supabase projekthez csatlakozunk ez 2 lépésre csökken:

1. Futtatási környezet telepítése
2. A szerverprogram telepítése

### 2.3.1. Supabase projekt felállítása

A supabase lehetőséget biztosít projektek online és lokális futtatására is. Az egyszerűség és erőforrás-takarékosság kedvéért az előbbit választom.

Egy új projekt készítéséhez először egy organizációt[1] kell létrehozni. Itt egy név megadása után egyből átirányít az új projekt oldalra. Az új projekt oldalon szintén egy név megadására, és egy adatbázis jelszó generálására kerül sor. Ezek után létrehozható az új projekt.

### 2.3.2. Adatbázis konfigurálása

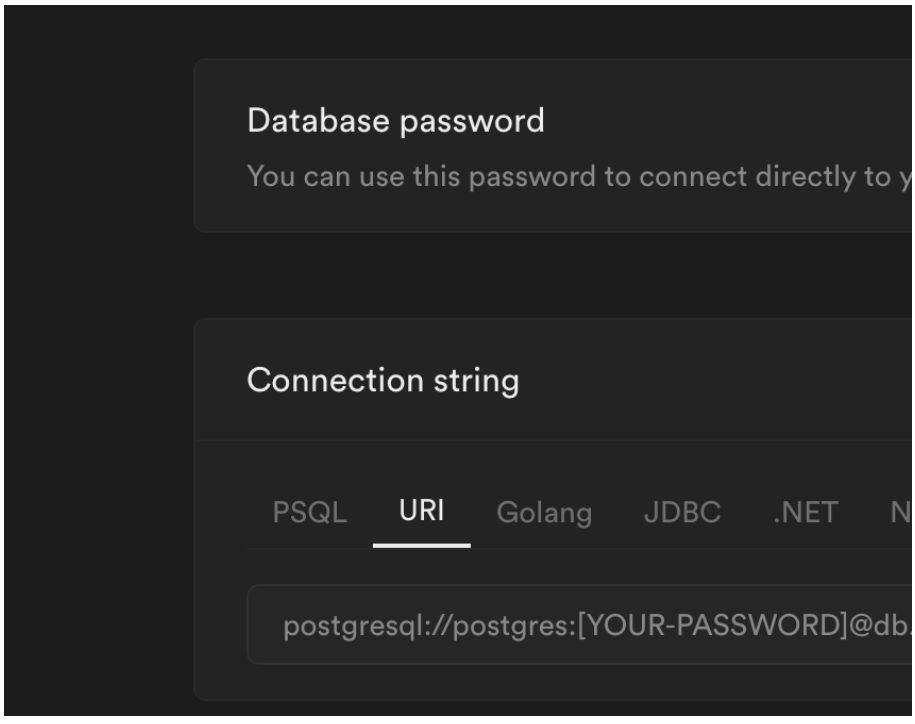
Az adatbázis konfigurálásának követelménye a PostgreSQL telepítése, ez elérhető a következő hivatkozáson: `https://www.postgresql.org/download/`.

A konfiguráció a parancssoron és a projekt Dashboard-on keresztül végezhető el. A táblák elkészítésére és adatokkal való feltöltésére egy psql[2] parancs futtatása ad lehetőséget. Ennek folyamata a tobbjatekos-kartyajatek-keretrendszer mappából futtatott következő parancs:

```
1 psql --single-transaction --variable ON\_ERROR\_STOP=1 --file roles
    .sql --file schema.sql --file data.sql --dbname "SAJAT ADATBAZIS
     URI"
```

2.1. forráskód. Adatbázis elkészítése

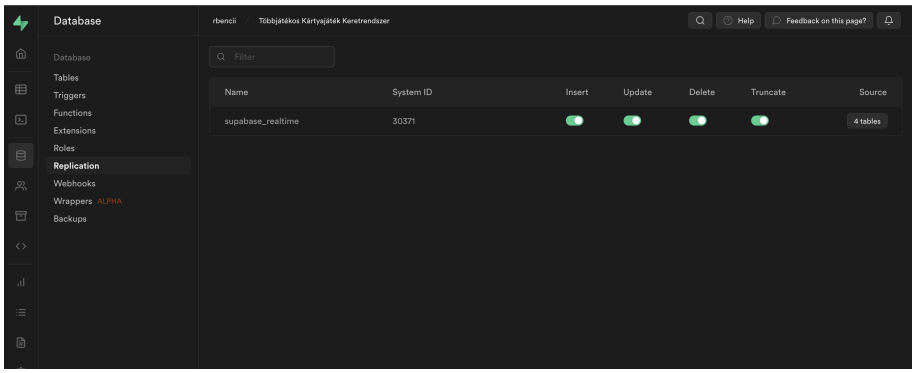
A saját adatbázis URI a `https://supabase.com/dashboard/project/_/settings/database` hivatkozáson megjelelő "Connection String" ablak "URI" gombjának megnyomásával érhető el. Ennek a "[YOUR-PASSWORD]" részét ki kell cserélni az adatbázis jelszavára.



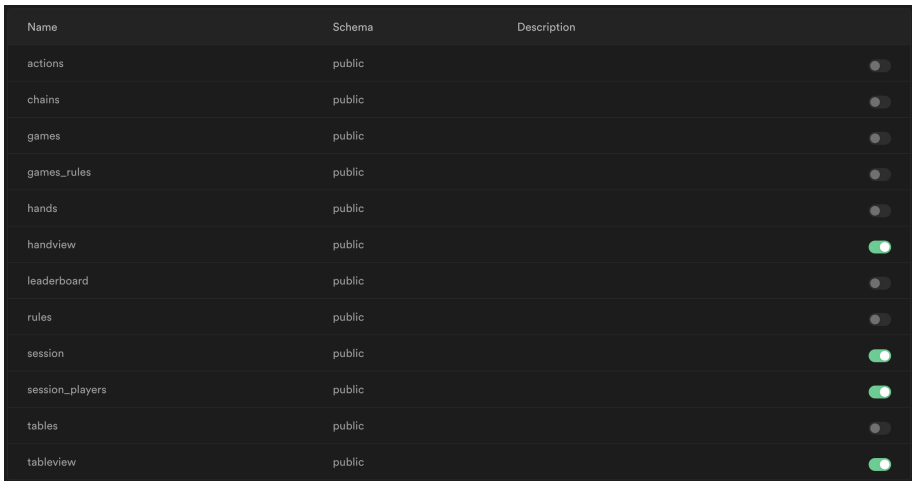

2.1. ábra. Supabase SQL szerkesztő

A táblák létrehozása után a valós idejű funkciók támogatását[3] engedélyeznünk kell. Ehhez teendő lépések:

1. supabase_realtime jobb oldalán lévő "source" megnyitása
2. A következő táblázatok melletti jelölőnégyzetek engedélyezése:
   - handview
   - session
   - session_players
   - tableview




(a) Supabase replication menü




(b) supabase_realtime tábla forrásai

2.2. ábra. Valós idejű funkciók engedélyezése

### 2.3.3. Futtatási környezet telepítése

A Node.js előző verziói a `https://nodejs.org/en/download/releases` hivatkozáson érhetőek el. Az operációs rendszernek megfelelő telepítő kiterjesztéssel rendelkező fájl segítségével telepíthető.

A telepítés után a verzió ellenőrzésére a "node -v" parancssori paranccsal lehetséges.

### 2.3.4. A szerverprogram telepítése

A szerverprogram függőségeinek telepítése a Node Package Manager[3] használatával történik. Ennek folyamata a tobbjatekos-kartyajatek-keretrendszer mappába navigálás utáni "npm i" parancssori parancs futtatása.

### 2.3.5. Az API-hoz szükséges adatok megadása

Ahhoz, hogy a szerverprogram csatlakozni tudjon a Supabase projekthez, definiálni kell a megfelelő környezeti változókat. Az ehhez szükséges adatok a projekt beállítások API[4] részén elérhetőek.

A környezeti változók fájljának elérési útja a "tobbjatekos-kartyajatek-keretrendszer/.env.local". Itt a következőképpen kerülnek definiálásra a változók:

- NEXT_PUBLIC_SUPABASE_ANON_KEY legyen egyenlő a beállítások oldal "Project URL" részén megjelenő "anon public" kulccsal.
- NEXT_PUBLIC_SUPABASE_URL legyen egyenlő a beállítások oldal "Project API keys" részén megjelenő URL-lel.
- PRIVATE_SERVICE_KEY legyen egyenlő a beállítások oldal "Project API keys" részén elrejtett "service_role" kulccsal.

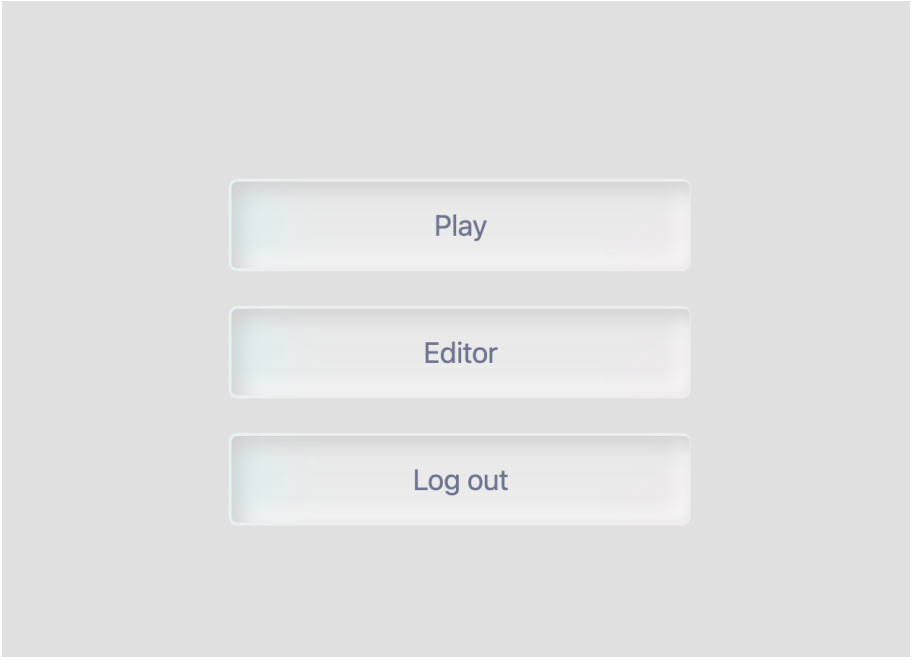
### 2.3.6. A szerver futtatása

A szerver futtatásához a tobbjatekos-kartyajatek-keretrendszer mappában kell végrehajtani két parancssori parancsot. Az első az "npm run build", ami az alkalmazás egy optimalizált változatát állítja elő. A második pedig az "npm run start", ami ezt a változatot futtatja.

[3] A Node Package Manager (npm) a Node.js alapértelmezett csomagkezelője.

## 2.4. Felépítés

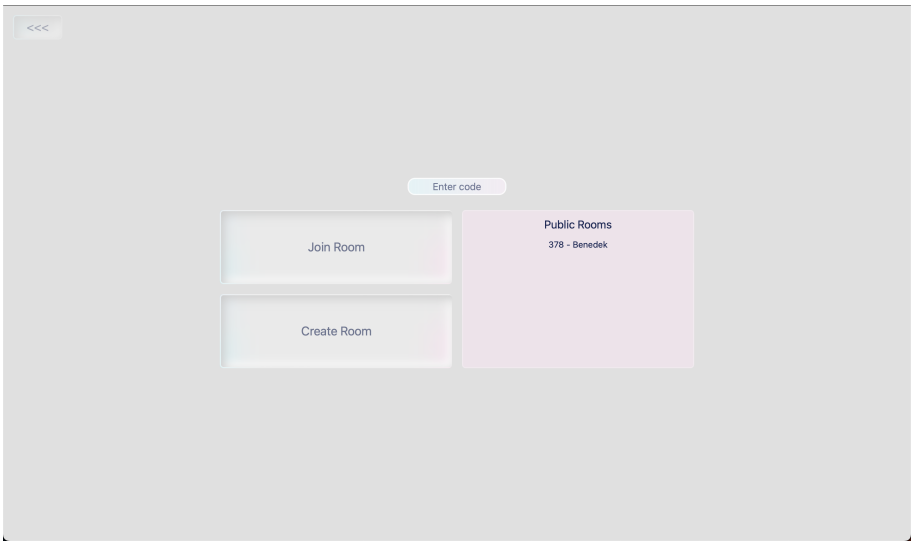
Az applikáció működés szerint két főbb részre osztható, a játékfelületre és a szerkesztőfelületre. Ezek bejelentkezés után érhetőek el.




2.3. ábra. Főmenü

## 2.5. Játékfelület

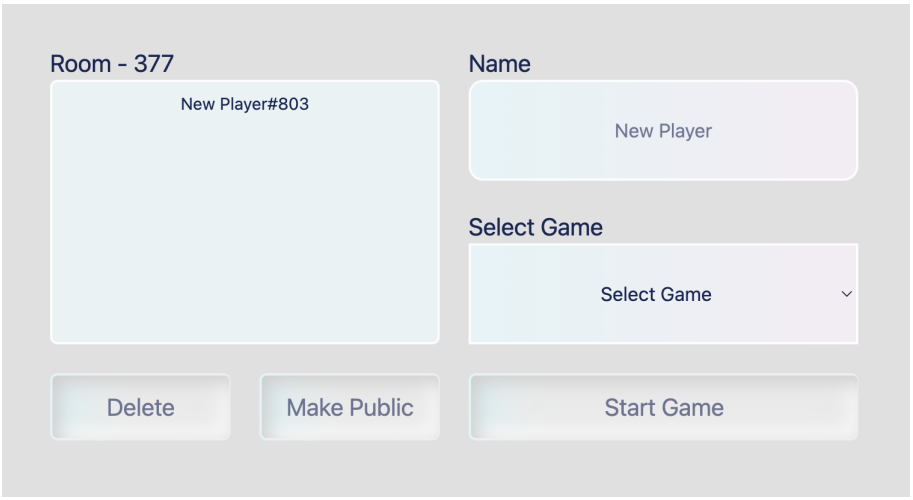
A játékfelületre kerülés módja a bejelentkezés utáni főmenü "Play" gomb megnyomása. Itt, ha a felhasználó már részese egy futásban lévő játéknak, akkor a rendszer felismeri és tovább viszi az asztalhoz. A játékokra úgynevezett szobákban kerül sor.




2.4. ábra. "Play" utáni képernyő

### 2.5.1. Szoba készítése

Új szoba a "Create Room" gomb megnyomásával készíthető. A szobában a készítőnek lehetősége van a neve megváltoztatására, a szoba nyilvánossá tételére és törlésére. A név változtatása a szöveg begépelése során automatikusan történik. A játék indításához kiválaszt egy hivatalos vagy saját játékot és megnyomja a "Start Game" gombot. Ha a játékkonfigurációnak megfelelő számú játékos van jelen a szobában akkor a játékmenet elindul.




2.5. ábra. Készítő játékos szobanézete

### 2.5.2. Belépés egy szobába

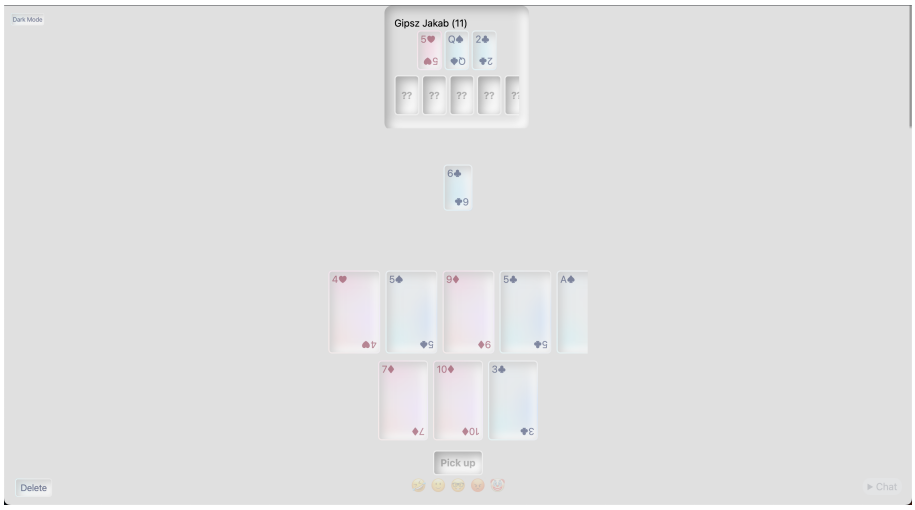
A szobákba a hozzájuk tartozó kód megadásával lehet belépni. A felhasználó kiválasztja a nyilvános szobák egyikét, vagy elkéri a társa által készített szoba kódját, és beírja az "Enter code" mezőbe, aztán enter-t nyom, vagy megnyomja a "Join Room" gombot.

A szobában lehetősége van a neve megváltozására vagy a szoba elhagyására. Az egyes játékosok nevei mellett lévő kupa-szám párosok a kiválasztott játékban szerzett győzelmeinek számát jelzik.


2.6. ábra. Belépő játékos szobanézete

### 2.5.3. Játék folyamata

Játékmenet során a felhasználó felül a többi játékos kártyáit láthatja, rejtve vagy megjelenítve. Középen az asztalon lévő kártyákat, lent pedig a sajátját. Egyszerre csak két sornyi kártya jelenik meg, illetve a gombként definiált kártyák egy harmadik sorban. A kártyákon kívül az ablak bal oldalán egy "Dark Mode" gomb és a szoba készítője esetén a "Delete" gomb látható.

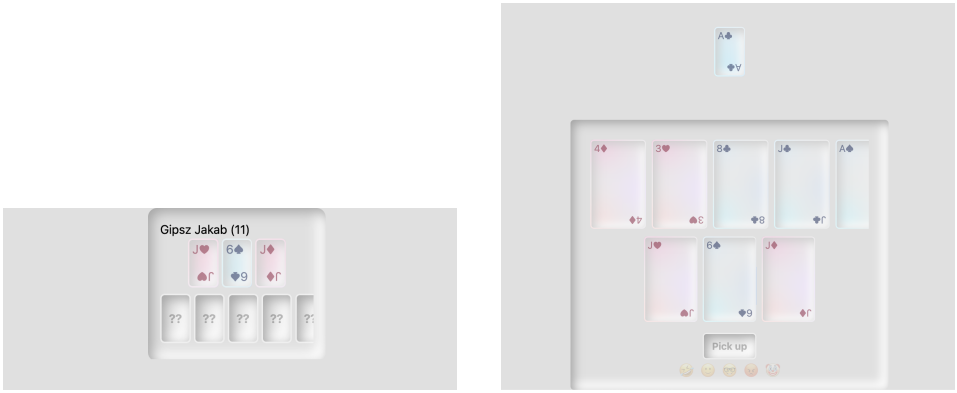



2.7. ábra. Játékmenet vizualizációja

Kommunikációra a képernyő legalján lévő reakciógombok, vagy a jobb alsó sarokban elhelyezkedő "Chat" gombbal kinyitható beszélgetés ablak ad módot.

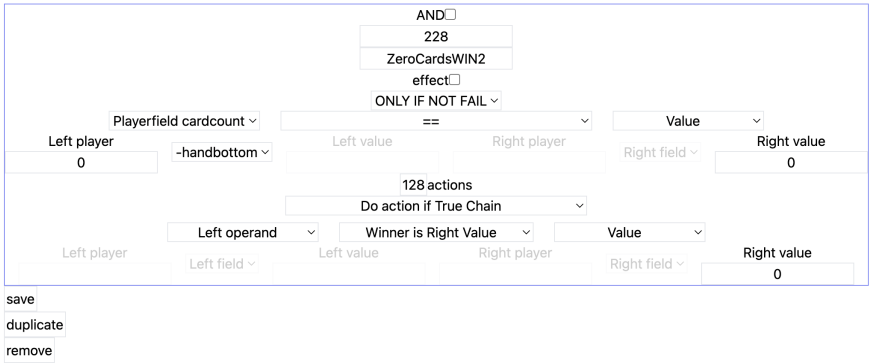

2.8. ábra. Beszélgetés ablak és reakciók

A játékmenet általánosan egy kártya lerakásából és a következő játékos beállításából áll. A kártya lerakása akkor lehetséges, ha a definiált szabályok láncainak valamelyike igaz, vagy van meghatározva hiba akció (lásd: 2.1 Táblázat "On Fail Do action" része). A soron következő játékost a kártyái csoportját körbevevő keret jelzi. Egy sorban csak korlátozott számú kártya fér ki teljes egészében, a többi görgetés után közelíthető meg. Ha a nyerés akció végbemegy, a játék pár másodpercen belül véget ér, és a győztes játékos nyert játékainak száma eggyel nő. A résztvevők mindannyian bekerülnek egy új szobába.




(a) Más játékos következik (b) Saját felhasználó következik

2.9. ábra. Soron lévő játékos jelzése




2.10. ábra. Nyerés akció példája

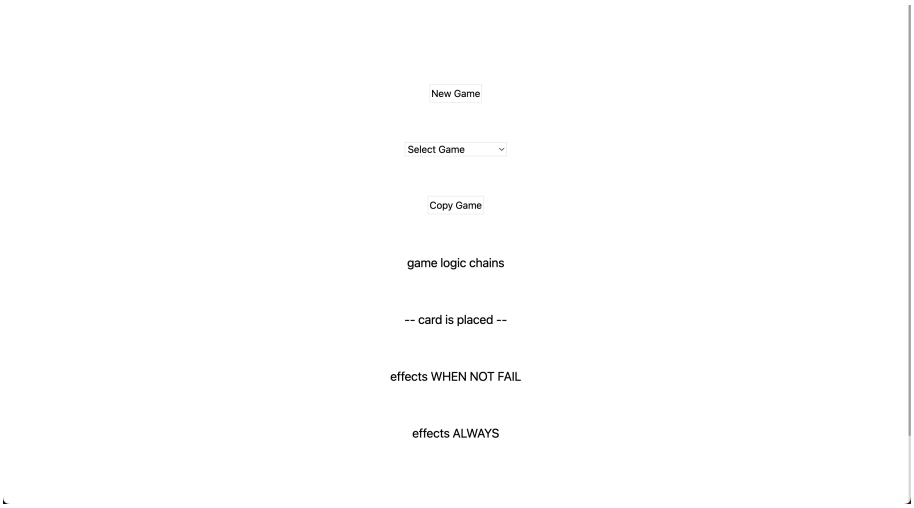
### 2.5.4. Játék szerkesztése

A szerkesztőfelület segítségével a felhasználók lemásolhatnak egy már meglévő hivatalos játékot, vagy készíthetnek egy újat. Új játék készítése során 6 adatot kell megadni:

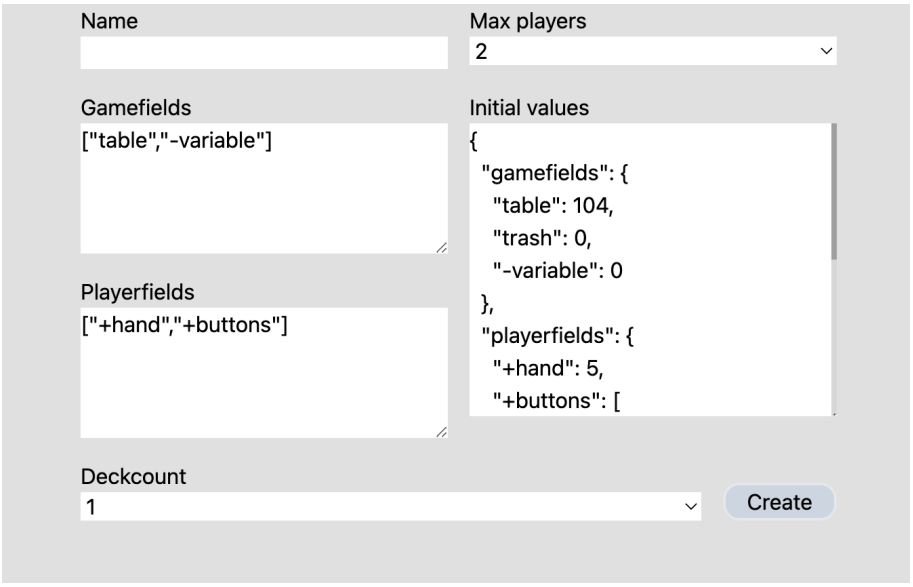
1. "Name" a játék neve.
2. "Max player" a játékosok maximális engedélyezett száma.
3. "Gamefields" az asztal változóinak indexei. A '-' prefix segítségével rejtett kártyacsoport hozható létre. A "table" létezése kötelező.
4. "Playerfields" egy játékos változóinak indexei. A '-' prefix segítségével mindenki számára rejtett, a '+' prefix használatával mindenki számára látható, prefix használata nélkül csak saját felhasználó által látható kártyacsoport hozható létre.

5. "Initial values" az asztal és játékosok kártyáinak generálásakor figyelembe vett értékek. Egy szám érték a pakliból kiosztott kártyák számát jelenti. Lehetséges nem véletlenszerű értékeket is megadni a megfelelő JSON tömb-object használatával.

6. "Deckcount" a játékhoz használt paklik[4] száma.




2.11. ábra. Szerkesztőfelület




2.12. ábra. Új játék készítése

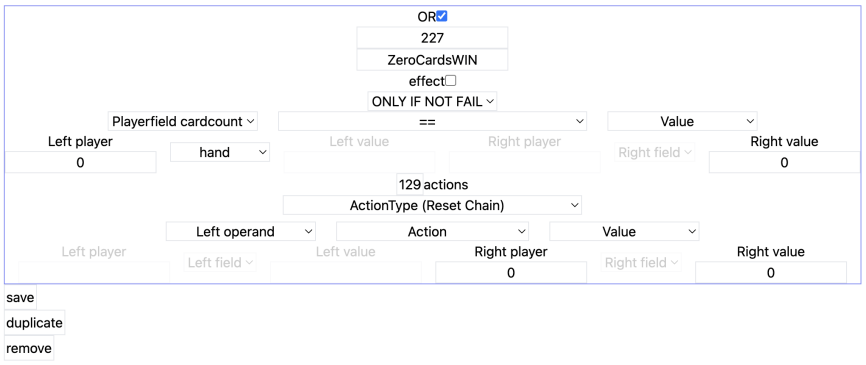
[4]Joker nélküli 52 lapos francia kártyapakli

Játék készítése után annak kiválasztásával elkezdhető a szerkesztés folyamata. A szerkesztés során kétféle szabályt különböztetek meg. Ezek a lerakás módját meghatározó kötelező szabályok, és a lerakás után megvizsgált, általában akciókkal[5] párosult effektus szabályok. Az előbbi jelölése a sárga, az utóbbié a kék keret.

A szabályok rendelkeznek egy AND/OR tulajdonsággal, ami alapján csatlakoznak az előttük lévő szabályokhoz, illetve értékelik ki a játszhatóság igazságértékét. Ha a kapcsolat első a csoportjában, AND esetén igaz, OR esetén pedig hamis értékkel indul a játszhatóság. A vízszintes[6] kapcsolatokon kívül lehetőség van függőleges kapcsolatok előállítására is, a láncok segítségével. A láncok egy lánc kezdetét és végét jelző szabályból, és az előző lánchoz vett AND/OR kapcsolatukból állnak. Egy lánc a kezdő és záró szabályt, illetve a köztük elhelyezkedő kötelező szabályokat foglalja magába.

Az effektusok a felületen helyes lerakást és helytelen/mindig bekövetkező effektusok csoportjaiba különülnek el. Kiértékelés során viszont az azonosítójuk, azaz a készítésük sorrendje alapján jönnek sorba.


2.13. ábra. Kötelező szabály




2.14. ábra. Effektus szabály

[5] Az akciók különböző logikai vizsgálatok után történő változtatások az asztalon, egy játékos kártyáin, vagy a játékmenet egyéb attribútumain.

[6] A "game logic chains" cím alatt ilyen irányban jelennek meg.

| **Szabályok felépítése felülről lefele, balról jobbra** | |
|---|---|
| *Input mező* | *Magyarázat* |
| *AND/OR jelölőnégyzet* | A vízszintes kapcsolatért felelős mező. Ez alapján értékeli ki a játszhatóságot. Ha az előző hamis volt, és itt AND szerepel, akkor a játszhatóság hamis. |
| *ID szöveg bemenet* | A szabály azonosítója. Új esetén -1, másolat esetén pedig -2. |
| *Név szöveg bemenet* | A szabály neve. Ha egy kötelező szabály neve megegyezik az előtte lévő szabály nevével, akkor az akciói az előző szabály akciófutása szerint futnak le, illetve a játszhatóság igazságértéke az előző szabály játszhatóságával lesz egyenlő, azaz nem értékelődik újra.<br>Nem kötelező szabályok esetén, ha az előző szabály játszhatósági értéke igaz volt, akkor lefut az akció és a játszhatóság az előzőre állítódik vissza. |
| *Kötelező/effektus jelölőnégyzet* | Ez határozza meg a szabály típusát. |
| *Bal oldali operandus bemenet* | Az összehasonlítás bal oldali értéke vagy értékének típusa. |
| *Összehasonlító operátor bemenet* | Az összehasonlítás igazságértékét képző operátor. |
| *Jobb oldali operandus bemenet* | Az összehasonlítás jobb oldali értéke vagy értékének típusa. |
| *Bal oldali játékos bemenet* | Megadja a bal oldali játékost az épp soron lévő játékoshoz relatívan. Bizonyos operandusok esetén átalakul, például "Gamefield cardcount"-nál. Itt azt jelzi, hogy a paklit vagy az asztalon lévő kártyákat veszi-e figyelembe. |
| *Bal oldali mező bemenet* | Megadja, hogy a bal oldali operandus melyik mezőben helyezkedik el. |
| *Bal oldali érték bemenet* | A bal oldali operandus egy index vagy érték megadására használatos bemenete. Kártya érték esetén az értékekből le kell vonni kettőt, mert indexként értendő. Így a 2-es kártya értéke 0-val egyenlő. |

| *Input mező* | *Magyarázat* |
|---|---|
| *Jobb oldali játékos bemenet* | Megadja a jobb oldali játékost az épp soron lévő játékoshoz relatívan. Bizonyos operandusok esetén átalakul, például "Gamefield"-nél. Itt azt jelzi, hogy a paklit vagy az asztalon lévő kártyacsoportot veszi-e figyelembe. |
| *Jobb oldali mező bemenet* | Megadja, hogy a jobb oldali operandus melyik mezőben helyezkedik el. |
| *Jobb oldali érték bemenet* | A jobb oldali operandus egy index vagy érték megadására használatos bemenete. Kártya érték esetén az értékekből le kell vonni kettőt, mert indexként értendő. Így a 2-es kártya értéke 0-val egyenlő. |
| *Akció hozzáadása gomb* | Ezzel adható a szabályhoz akció. Akciók esetén többnyire ugyanazok a bemenet magyarázatok érvényesek. |

| *Input mező* | *Magyarázat* |
|---|---|
| *Akció típusa bemenet* | Ez befolyásolja az akciólefutás eldöntésének módszerét.<br>• ActionType (Reset Chain)<br>Az akció végrehajtása a szabály két operandusának összehasonlítása szerint történik. Effektus szabály esetén a játszhatóság igazságértékét beállítja a szabály összehasonlításának eredményére.<br>• On Fail Do action<br>Az akció akkor kerül végrehajtásra, ha a szabály összehasonlítása és előző szabállyal vett vízszintes kapcsolata hamis, de az előtte lévő játszhatóság igazságértéke igaz volt.<br>• Always fail<br>Az akció végrehajtása alap módon történik, viszont a kártya nem kerül le az asztalra.<br>• Do action if True Chain<br>Az akció végrehajtása a játszhatóság igazságságértéke szerint történik.<br>• (Do action and Don't place card) if True Chain<br>Ha a játszhatóság igaz, az akciót végbemegy, de nem kerül le a kártya az asztalra. |

2.1. táblázat. Szabályok mezőinek magyarázata

| **Operandusok magyarázatai** | |
|---|---|
| *Operandus* | *Magyarázat* |
| *Placed card value* | A lerakni kívánt kártya értékének indexe, kivéve "Move X from Left to Right" és "Set Left Card to Right" esetén, ilyenkor a lerakni kívánt kártya. |
| *Placed card suit* | A lerakni kívánt kártya színe. |
| *Init playerfield* | A játékosmezők kezdeti értékei egyike. |
| *Init gamefield* | A játékmezők kezdeti értékei egyike. |
| *Card origin index* | A lerakni kívánt kártya eredetének játékosmező vagy játékmező indexe. |
| *Card origin field* | A lerakni kívánt kártya eredetének kártyacsoportja[7]. |
| *Playerfield card value* | Egy játékosmező adott indexű kártyájának értéke. |
| *Playerfield card suit* | Egy játékosmező adott indexű kártyájának színe. |
| *Playerfield cardcount* | Egy játékosmező adott indexű kártyacsoportjában lévő kártyák száma. |
| *Gamefield card value* | Egy játékmező fentről számolt indexű kártyájának értéke. |
| *Gamefield card suit* | Egy játékmező fentről számolt indexű kártyájának színe. |
| *Gamefield cardcount* | Egy játékmező adott indexű kártyacsoportjában lévő kártyák száma. |
| *Playerfield* | A játékos egy kártyacsoportja. |
| *Playerfield index* | Egy játékosmező index. |
| *Gamefield* | A játék egy kártyacsoportja. |
| *Gamefield index* | Egy játékmező index. |
| *Placed card position* | A lerakni kívánt kártya indexe. |
| *Value* | Szám érték. |
| *Next* | A következő játékos távolsága a jelenleg soron lévőhöz képest. |

2.2. táblázat. Operandusok magyarázata

[7]Játékmező vagy játékosmező

| **Operátorok magyarázatai** | |
|---|---|
| *Operátor* | *Magyarázat* |
| == | Bal oldal egyenlő a jobb oldallal. |
| *!=* | Bal oldal nem egyenlő jobb oldallal. |
| >= | Bal oldal nagyobb, vagy egyenlő a jobb oldallal. |
| <= | Bal oldal kisebb, vagy egyenlő a jobb oldallal. |
| > | Bal oldal nagyobb, mint a jobb oldal. |
| < | Bal oldal kisebb, mint a jobb oldal. |
| *Number of Right (Single) in Left (Multiple) == lv* | A jobb oldali kártyaértékkel vagy kártyaszínnel[8] rendelkező kártyák száma a bal oldali kártyacsoportban megegyezik a "Left value" bemenet értékével. |
| *Number of Right (Single) in Left (Multiple) >= lv* | A jobb oldali kártyaértékkel vagy kártyaszínnel[9] rendelkező kártyák száma a bal oldali kártyacsoportban nagyobb, vagy egyenlő a "Left value" bemenet értékével. |
| *Number of Right (Single) in Left (Multiple) <= lv* | A jobb oldali kártyaértékkel vagy kártyaszínnel[10] rendelkező kártyák száma a bal oldali kártyacsoportban kisebb, vagy egyenlő a "Left value" bemenet értékével. |
| *Left (Multiple) has Right (Single)* | Bal oldali csoport tartalmazza a jobb oldalt. |

2.3. táblázat. Operátorok magyarázata

[8]Operandustól függően
[9]Operandustól függően
[10]Operandustól függően

| **Akciók magyarázatai** | |
|---|---|
| *Akció* | *Magyarázat* |
| *Fill Left from Right* | A bal oldali kártyacsoportot feltölti a kezdeti értékekben megadott méretre a jobb oldali kártyacsoportból. |
| *Move X from Left to Right* | A bal oldali kártyacsoportból "Left value" mennyiségű kártya átmozgatása a jobb oldali kártyacsoportba. Ha a bal operandus "Placed card value" akkor a lerakni kívánt kártya kerül áthelyezésre. A -1 érték a kártyacsoportban lévő összes kártyát jelenti. |
| *Next Player* | A következő játékos beállítása a jelenleg soron lévő játékoshoz relatívan. |
| *Set Left Card to Right* | Bal oldali kártya beállítása a jobb oldalra. Ha a jobb operandus "Value", akkor az új kártya színe "Right Player" számértékének[11] megfelelő index, az értéke a "Right value" szerinti index, a sorrendelő értéke pedig a "Left value" bemenet értéke. |
| *Set Left Card Value to Right* | Bal oldali kártyacsoport "Left value" bemenet értéke szerint indexelt kártya értékének jobb operandusra állítása. Ajánlott jobb operandus a "Value" |
| *Set Left Card Suit to Right* | Bal oldali kártyacsoport "Left value" bemenet értéke szerint indexelt kártya színének jobb operandus szerint indexelt színre állítása. Ajánlott jobb operandus a "Value" |
| *Set Round Direction to Right* | A kör irányának megváltoztatása. 0 esetén megváltoztatásig az eddig soron lévő játékos fog folyamatosan következni. |
| *Break Chain* | A játéklogika értelmezésének megszakítása. |
| *Continue Chain* | A következő láncra ugrás. |
| *Winner is Right Value* | A nyertes a "Right value" bemenet által megadott jelenleg soron lévő játékoshoz nézve relatív játékos. |

2.4. táblázat. Akciók magyarázata

[11] 0: káró, 1: kör, 2: pikk, 3: treff

A komplikáltabb játékok készítésére példát nyújt az általam készített supabase projektben elérhető holland kocsma nevű játék.

Az könnyű értelmezés érdekében egy egyszerű példán keresztül bemutatom a szerkesztési folyamatot. A játékban két felhasználó vehet részt, mindketten az összes színű ász kártyákkal indulnak. Kiválasztanak egy kártyát, aztán az asztalra kerül egy lap a pakliból. A résztvevők célja az, hogy az asztalra kerülő lap színét eltalálják. Az nyer, akinek a játék végére többször sikerült ez.

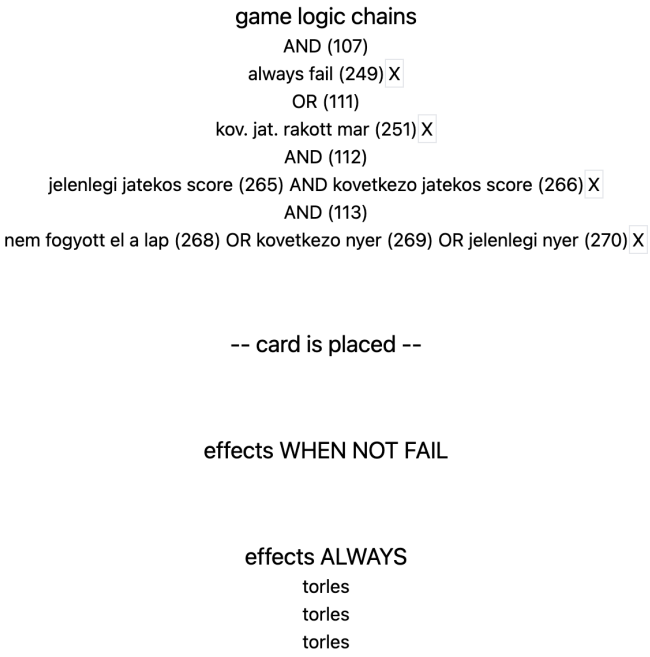
Játékadatok megadása:

| **Bemenetek értékeinek megadása** | |
|---|---|
| *Input mező* | *Megadott érték* |
| *Name* | Guessing game |
| *Max players* | 2 |
| *Gamefields* | [ "table", "-trash" ] |
| *Playerfields* | [ "+chosen", "hand", "-score" ] |
| *Initial values* | {<br>"gamefields": {<br>"table": 10,<br>"-trash": 20<br>},<br>"playerfields": {<br>"hand": [<br>{ "suit": "Diamonds", "value": "A", "sorter": 1 }, { "suit": "Hearts", "value": "A", "sorter": 2 }, { "suit": "Spades", "value": "A", "sorter": 3 }, { "suit": "Clubs", "value": "A", "sorter": 4 } ],<br>"-score": 0,<br>"+chosen": 0<br>}<br>} |
| *Deckcount* | 1 |

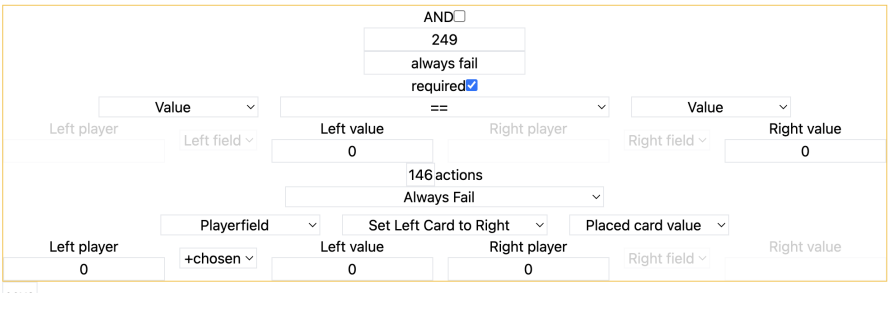
2.5. táblázat. Példa játék adatai

Szabály törlésekor az oldalt frissíteni kell! Új szabály mentése után, a láncképzés megfelelő működése érdekében, a szabályok listáját frissíteni kell a "Refresh" gombbal.

A szerkesztőfelületen megadandó szabályok, effektusok és láncok:

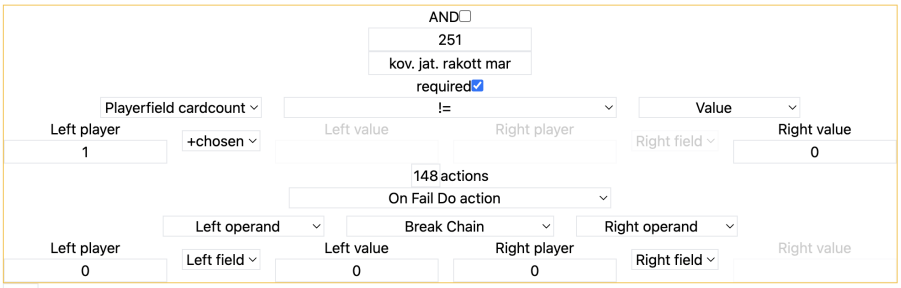



2.15. ábra. Példa játék kapcsolatai



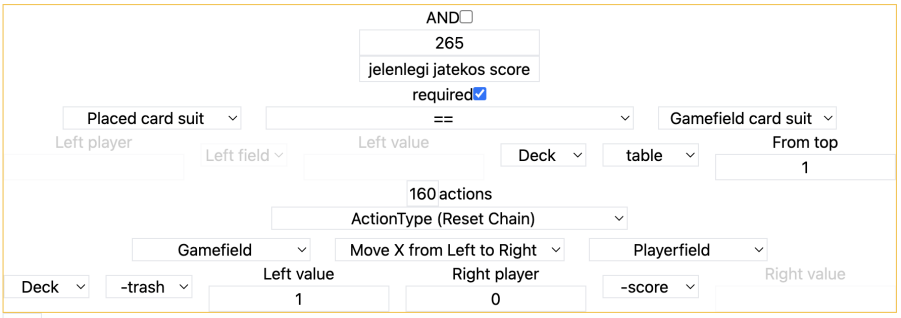

2.16. ábra. Első kötelező szabály

A lerakni kívánt kártya ne kerüljön le az asztalra, illetve bármilyen esetben engedélyezett legyen a mozdulat.



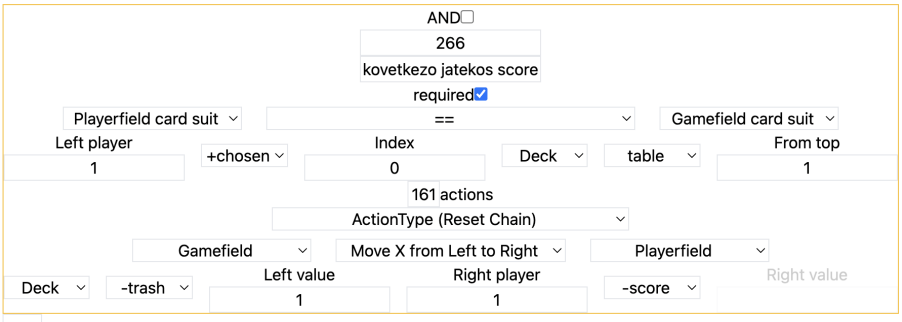

2.17. ábra. Második kötelező szabály

Ha még nem választott színt a másik játékos, akkor megszakítja a lánc kiértékelését.



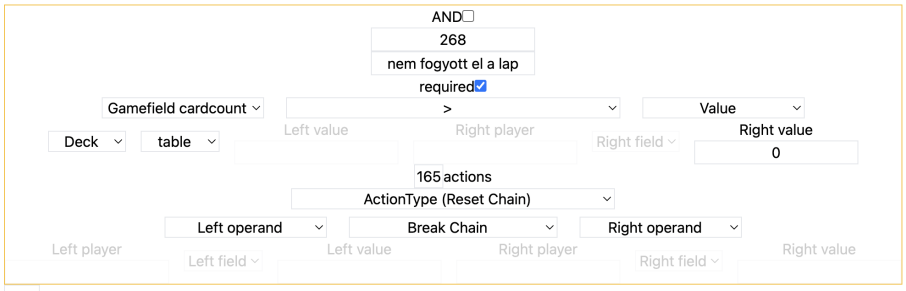

2.18. ábra. Harmadik kötelező szabály

Ha a jelenlegi játékos eltalálta az asztalra kerülő lap színét, akkor növeli eggyel a pontszámláló kártyacsoportját.



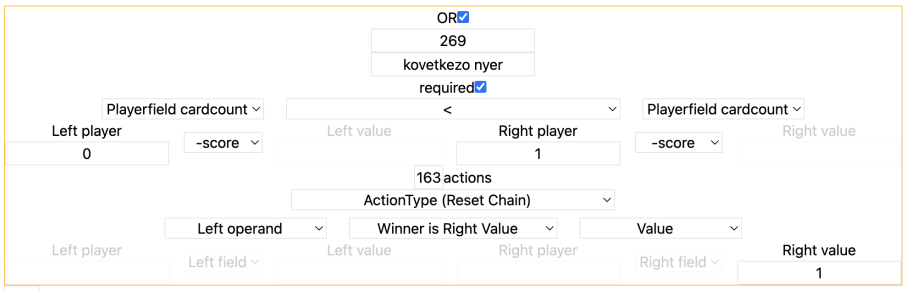

2.19. ábra. Negyedik kötelező szabály

Ha a másik játékos eltalálta az asztalra kerülő lap színét, akkor növeli eggyel a pontszámláló kártyacsoportját.



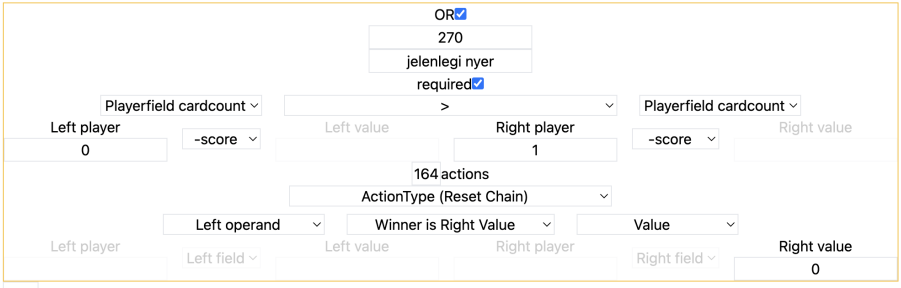

2.20. ábra. Ötödik kötelező szabály

Ha a pakliban lévő kártyák száma nagyobb mint 0, akkor megszakítja a lánc kiértékelését.




2.21. ábra. Hatodik kötelező szabály

Ha a másik játékos pontszámláló kártyacsoportja több kártyát tartalmaz, ő nyer.




2.22. ábra. Hetedik kötelező szabály

Ha a jelenleg soron lévő játékos pontszámláló kártyacsoportja több kártyát tartalmaz, ő nyer.


(a) Első effektus szabály


(b) Második effektus szabály


(c) Harmadik effektus szabály

2.23. ábra. A törlés szabályra épülő három akció

Ha a mindkét játékos választott színt a kártyák segítségével, akkor ezeket töröljük, és a pakliból felfedjük a megtippelt kártya színét.

# 3. fejezet

# Fejlesztői dokumentáció

## 3.1. Tervezés

A cél egy olyan webes applikáció létrehozása, amely lehetőséget biztosít francia kártyával játszott játékok elkészítésére és ezek online többjátékos játékára. Hangsúlyt kell helyezni a csalás meggátlására, egyéb tevékenységek biztonságára, és a reszponzív megjelenésre. A játéklogika szerveroldalon történik, a megjelenítés kliensoldalon. A csalás elleni és biztonság érdekében tett igyekezetek pedig a szerveroldalon, kliensoldalon és az adatbázis szintjén történnek.

Funkcionális követelmények:

- Játék szerkesztése
  - Saját játék szerkesztése, hivatalos játék megtekintése
  - Új játék készítése
  - Meglévő, admin által adatbázisban hivatalossá nyilvánított játékok lemásolása
  - Szabályok definiálása
  - Akciók létrehozása, és ezek szabályokhoz kötése
  - Szabályok szoros összekötése, csoportosítása láncok segítségével
- Játékok játszását biztosító szobák funkciói
  - Szoba készítése
  - Szoba nyilvánossá tétele
  - Szoba törlése
  - Játék kiválasztása

  - Játék indítása
  - Belépés szobába
  - Kilépés a szobából
  - Név megváltoztatása
  - Győzelmek számának megjelenítése

- Játékok játszása
  - Saját, és egyéb játékosok lapjának megjelenítése láthatóság szerint
  - Az asztalon lévő lapok megjelenítése
  - Kártya asztalra helyezése szabályok menete szerint, lehelyezés következménye
  - Soron lévő játékos jelzése
  - Szoba készítője által játék megszakítása törlés gombbal
  - Beszélgetés ablak
  - Reakciók gombok segítségével

### 3.1.1. Játék terve

A játékok készítéséhez át kell gondolni, hogy milyen mezők segítségével lehet eléggé leírni egy kártyajáték tulajdonságait.

Először is az általános, játék menetét túlságosan nem befolyásoló tulajdonságok:

- Játék készítője
- Játék neve
- Maximális játékosok száma
- Hivatalosság
- Használt paklik száma

Ezután játékmenet tulajdonságai következnek. Egy kártyajátékban többnyire az asztalon és játékosonként jönnek létre kártyacsoportok, különböző kezdeti értékek alapján:

- Asztalon belül kialakuló csoportok
- Játékosonként kialakuló csoportok
- Az egyes csoportok kialakítását szabályzó kezdeti értékek

## 3.2. Tételek, definíciók, megjegyzések

### 3.2.1. Szobák terve

Annak érdekében, hogy egyszerre több játék is folyamatban lehessen, a játékok szobákba különülnek.

Szobák tulajdonságai:

- Szoba készítője
- Szobában játszott játék azonosítója
- Folyamatban van-e a játék
- Szoba nyilvánossága

### 3.2.2. Szabályok, akciók és láncok terve

A játékmenet leírását a szabály-akció párosok és ezek összeláncolása teszi lehetővé.

Egy szabály felépítése:

- Szabály neve
- Összehasonlítás bal oldalának típusa
- Összehasonlítás jobb oldalának típusa
- Összehasonlítás oldalainak adatai (játékos vagy asztal indexek, mezők, értékek)
- Szabály típusa
- Effektus szabály esetén lefutást szabályozó kapcsoló
- Előző szabályhoz vett kapcsolat
- Összehasonlítást végző operátor
- Szabályhoz tartozó akció

Egy akció felépítése:

- Akció lefutásának típusa
- Akció bal oldalának típusa
- Akció jobb oldalának típusa
- Akció oldalainak adatai (játékos vagy asztal indexek, mezők, értékek)
- Akció típusa

A szabályok összekötésére használt láncok egy kezdést és véget jelző szabályból, illetve előző lánchoz vett kapcsolatukból állnak.

### 3.2.3. Kártyacsoportok terve

A kártyák ábrázolására egy kártyacsoportot kétféleképpen kell reprezentálni. Egy működés szempontjából fontos, játék állapotát bemutató reprezentációval, és egy felhasználók általi láthatóság szerinti reprezentációval. Ezzel biztosítva a lefordított kártyák rejtettségét.

Nem játékos által birtokolt kártyacsoport esetén ez a paklit és az asztalra letett kártyák csoportját jelenti.

## 3.3. Alkalmazott technológiák

### 3.3.1. TypeScript

A TypeScript[5] a JavaScript típusokkal való kibővítése. Ezt statikus típus ellenőrzéssel teszi. Emellett kiegészíti a szintaxist, ezzel segíti a hibák mihamarabbi felismerését.

### 3.3.2. Next.js

A Next.js[6] egy React[7] alapú keretrendszer webes applikációk készítésére. Segítségével könnyen elérhető a dinamikusság, interaktivivtás és gyorsaság. Ezt különböző optimalizációkkal és további struktúrák és funkciók biztosításával teszi meg.

A Többjátékos kártyajáték keretrendszer szempontjából leghasznosabb funkciói:

- Egyszerű oldalstruktúra és ezen belüli útvonal intézés
- Stílusozás átfogó támogatása

- TypeScript támogatás
- Saját API végpontok készítése

Next.js használt verziója: 13.2.4.

### 3.3.3. Tailwind CSS

A Tailwind CSS[8] egy CSS keretrendszer. Segítségével különböző CSS stílusok előre megadott HTML osztály megfelelőit használva, gyorsan és egyszerűen lehet kialakítani egy weboldal megjelenését.

### 3.3.4. Supabase

A Supabase[9] egy nyílt forráskódú Firebase[10] alternatíva. Egy olyan fejlesztési platform ami segítségével könnyen lehet hatékony Postgres[2] backend-eket készíteni.

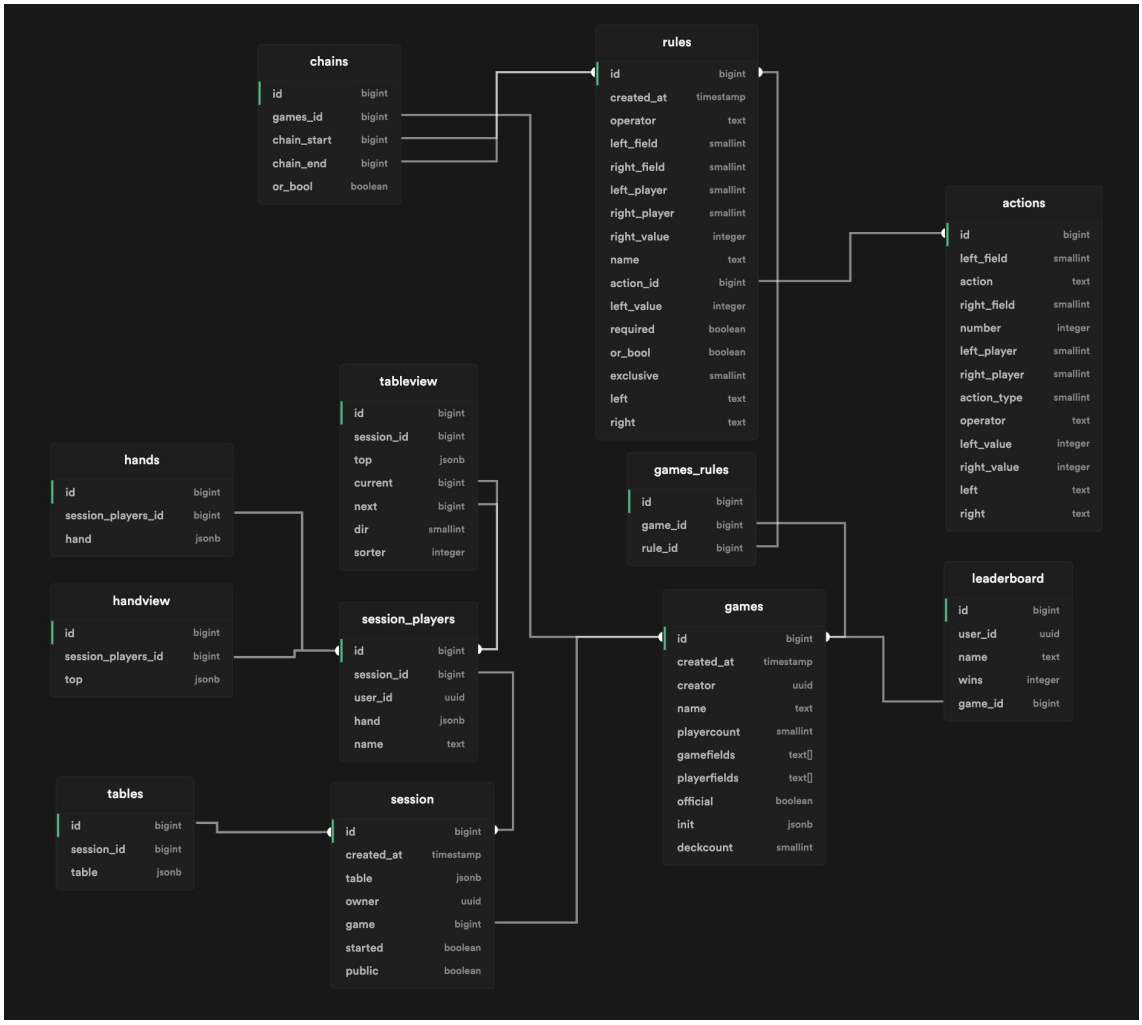
A Többjátékos kártyajáték keretrendszer szempontjából leghasznosabb funkciói:

- Ingyenes hosting
- Postges adatbázis kezelése
- Fejlett, könnyen használható API
- Realtime csatornák
- Autentikáció
- Auth UI[11] kész komponens
- Next.js Auth Helpers[12]

### 3.3.5. PostgreSQL

A PostgreSQL[2] egy relációs adatbáziskezelő. Több, mint 35 éve fejlesztik, ezáltal hatékony és megbízható választás.

## 3.4. Adatbázis felépítése




3.1. ábra. Adatbázis generált[13] diagramja

Az írás és olvasási jogok védelme főként Row-Level Security[1] használatával történik, kivéve pár különleges esetben, ahol a Next.js backend kezeli.

Táblánkénti RLS szabályok:

- actions
  - Olvasás az összes felhasználó számára
  - Beillesztés csak bejelentkezett felhasználók számára
  - Frissítés csak a saját akciók esetén

[1]Postgres sor alapú védelme

- chains
  - Olvasás az összes felhasználó számára
  - Beillesztés csak bejelentkezett felhasználók számára, ha saját játékukról van szó és a szabály a játékhoz tartozik
  - Törlés csak a saját lánc esetén
- games
  - Olvasás az összes felhasználó számára
  - Beillesztés csak bejelentkezett felhasználók számára
- games_rules
  - Olvasás az összes felhasználó számára
  - Törlés csak a saját kapcsolatok esetén
- handview
  - Olvasás a saját látható kártyák esetén
- leaderboard
  - Olvasás az összes felhasználó számára
- rules
  - Olvasás az összes felhasználó számára
  - Beillesztés csak bejelentkezett felhasználók számára
  - Frissítés csak a saját szabályok esetén
- session
  - Olvasás az összes felhasználó számára
  - Beillesztés csak bejelentkezett felhasználók számára
  - Frissítés csak a saját szobák esetén
  - Törlés csak saját szobák esetén
- session_players
  - Olvasás az összes felhasználó számára
  - Beillesztés csak, ha létezik a szoba és még nincs folyamatban a játék
  - Frissítés csak, ha még nincs folyamatban a játék
  - Törlés csak, ha nem saját szoba

## 3.5. Megvalósítás

A játéklogika és adatbázissal való kommunikáció a Next.js szerveroldalon történik, a válaszként kapott adatok megjelenítése, a beszélgetés ablak és a reakciók kezelése a Next.js kliensoldalon. Az adatbázis, autentikáció és valós idejű kommunikáció a Supabase API-ját alkalmazva történik.

### 3.5.1. Szerveroldali végpontok

A szerveroldali API végpontok funkciók szerint két részre különülnek.
Játékokhoz tartozó végpontok

- "game/addchain"
  - Metódus: POST
  - Várt body tartalma: games_id, chain_start, chain_end, or_bool
  - Viselkedés magyarázata: egy új lánc hozzáadása.
- "game/copygame"
  - Metódus: POST
  - Várt body tartalma: game_id
  - Viselkedés magyarázata: egy játék másolása.
- "game/deletechain"
  - Metódus: POST
  - Várt body tartalma: id
  - Viselkedés magyarázata: adott lánc törlése.
- "game/leaderboard"
  - Metódus: POST
  - Várt body tartalma: game_id, room_id
  - Viselkedés magyarázata: szobában tartózkodó játékosok adott játékhoz tartozó győzelmei száma.
- "game/list"
  - Metódus: GET
  - Viselkedés magyarázata: saját és hivatalos játékok listájának lekérdezése.

- "game/newgame"
  - Metódus: POST
  - Várt body tartalma: name, playercount, gamefields, playerfields, init, deckcount
  - Viselkedés magyarázata: új játék készítése.
- "game/removerule"
  - Metódus: POST
  - Várt body tartalma: game_id, rule_id
  - Viselkedés magyarázata: egy lánc eltávolítása a játéktól.
- "game/rules"
  - Metódus: POST
  - Várt body tartalma: id
  - Viselkedés magyarázata: adott játékhoz tartozó szabályok, láncok és akciók lekérdezése.
- "game/setrules"
  - Metódus: POST
  - Várt body tartalma: rules, game_id
  - Viselkedés magyarázata: szabály készítésének és szerkesztésének kezelése, beleértve az akciók kezelését.

Szobákhoz tartozó végpontok

- "room/delete"
  - Metódus: GET
  - Viselkedés magyarázata: a szoba törlése, ha a felhasználó készítette.
- "room/getfieldorders"
  - Metódus: POST
  - Várt body tartalma: id
  - Viselkedés magyarázata: egy adott játékhoz tartozó játékmezők és játékosmezők sorrendjének lekérdezése.

- "room/join"
  - Metódus: POST
  - Várt body tartalma: id
  - Viselkedés magyarázata: belépés egy adott szobába.
- "room/leave"
  - Metódus: GET
  - Viselkedés magyarázata: kilépés a szobából.
- "room/list"
  - Metódus: GET
  - Viselkedés magyarázata: a még nem indult nyilvános szobák lekérdezése.
- "room/makepublic"
  - Metódus: POST
  - Várt body tartalma: session_id
  - Viselkedés magyarázata: adott szoba nyilvánossá tétele.
- "room/name"
  - Metódus: POST
  - Várt body tartalma: name
  - Viselkedés magyarázata: a játékos nevének megváltoztatása.
- "room/new"
  - Metódus: GET
  - Viselkedés magyarázata: új szoba készítése.
- "room/place"
  - Metódus: POST
  - Várt body tartalma: handidx, cardidx
  - Viselkedés magyarázata: a kártya letevésének folyamata. A jelenleg játszott játékhoz tartozó láncok szabályainak vizsgálata és akcióinak végrehajtása. Ha engedélyezett, a kártya asztalra tétele, majd az effektusok kiértékelése és hozzájuk tartozó akciók futtatása.

- "room/presence"
  - Metódus: GET
  - Viselkedés magyarázata: a szobához és, ha van, a futó játékhoz tartozó adatok lekérdezése.
- "room/setgame"
  - Metódus: POST
  - Várt body tartalma: game_id
  - Viselkedés magyarázata: a szoba játékának beállítása, ha a felhasználó készítette.
- "room/start"
  - Metódus: GET
  - Viselkedés magyarázata: a szoba indítása, kártyák generálása a szobához tartozó játék előírása szerint. A kártyák keverése a Fisher Yates[14] modern algoritmusa szerint történik.

### 3.5.2. Kliensoldalon megjelenő oldalak és komponensek

Elérhető oldalak:

- index
  - Felhasznált komponensek: Auth, MenuButton
  - Magyarázat: bejelentkezett felhasználó esetén a főmenü, egyébként a bejelentkezés és regisztráció megjelenítése.
- editor
  - Felhasznált komponensek: Chainer, MenuButton, Rule
  - Magyarázat: a játékok szerkesztésének felülete.
- newgame
  - Magyarázat: új játék készítésére szolgáló felület.
- play
  - Felhasznált komponensek: Rooms, MenuButton
  - Magyarázat: a szobákat kezelő és játékmenetet megjelenítő felület.

Komponensek:

- Card2
  - Magyarázat: egy kártya megjelenítéséért felelős.
- Chainer
  - Magyarázat: a szabályokat összekötő láncok kezelését biztosítja.
- Chat
  - Magyarázat: a beszélgetés ablak komponense.
- Hand
  - Felhasznált komponensek: Card2
  - Magyarázat: a kártyacsoporotok megjelenítéséért felelős.
- MenuButton
  - Magyarázat: gombok stílusa.
- Reactions
  - Magyarázat: reakciók megjelenítéséért és kezeléséért felelős.
- Rooms
  - Felhasznált komponensek: Hand, Chat, Reactions, MenuButton
  - Magyarázat: a szobák és a játékmenet megjelenítéséért és kezeléséért felelős.
- Rule
  - Magyarázat: szabályok és a hozzájuk tartozó akciók megjelenítéséért és kezeléséért felelős.

## 3.6. Tesztelés

A tesztelést manuálisan végzem a felhasználó szemszögéből, a kód ismeretét figyelembe véve, ezzel fókuszálva a kliensoldal és szerveroldal működésére egyaránt.

| **Tesztelési terv** | |
|---|---|
| *Teszt eset* | *Várt eredmény* |
| *Index oldal megnyitása bejelentkezés nélkül* | A bejelentkezés ablakának megjelenítése. |
| *Bejelentkezés hibás adatokkal* | A bejelentkezés nem történik meg. |
| *Bejelentkezés helyes adatokkal* | A bejelentkezés megtörténik. |
| *Index oldal megnyitása bejelentkezés után* | A főmenü jelenik meg. |
| *"Log out" gomb megnyomása* | A felhasználó kijelentkezik. |
| */editor oldal megnyitása bejelentkezés nélkül* | Egy bejelentkezésre irányító gomb jelenik meg. |
| */editor oldal megnyitása bejelentkezés után* | A szerkesztő oldal jelenik meg. |
| */editor oldal megnyitása bejelentkezés nélkül* | Egy bejelentkezésre irányító gomb jelenik meg. |
| *Játék kiválasztása /editor oldalon* | Az betöltenek. |
| *Lánc melletti X gomb megnyomása /editor oldalon* | A lánc törlődik. |
| *Lánc hozzáadása helyes adatokkal /editor oldalon* | A lánc hozzáadásra kerül. |
| *"add new rule" gomb megnyomása /editor oldalon* | Egy új szabály szerkesztése jelenik meg. |

| *Teszt eset* | *Várt eredmény* |
|---|---|
| *"save" gomb megnyomása egy szabály alatt helyes adatok megadása után* | A szabályt menti a rendszer |
| *"remove" gomb megnyomása egy szabály alatt* | Az oldal frissítése után a szabály már nem látható betöltés után. |
| *"duplicate" gomb megnyomása egy szabály alatt* | A szabály egy másolata megjelenik szerkesztésre. |
| */play oldal megnyitása bejelentkezés nélkül* | Egy bejelentkezésre irányító gomb jelenik meg. |
| */play oldal megnyitása bejelentkezés után* | A szobákat kezelő felület jelenik meg. |
| *Hibás kód beírása aztán "Join Room" gomb megnyomása a /play oldalon* | Nem történik semmi. |
| *Helyes kód beírása aztán "Join Room" gomb megnyomása a /play oldalon* | A játékos belép a szobába. |
| *Helyes kód beírása aztán enter billentyűgomb megnyomása a /play oldalon lévő szöveg bemenetben* | A játékos belép a szobába. |
| *"Leave" gomb megnyomása a /play oldalon* | A játékos kilép a szobából. |
| *"Create Room" gomb megnyomása a /play oldalon* | A játékos készít egy szobát. |
| *Név gépelése a szoba felületen látható szöveg bemenetbe* | A játékos neve megváltozik. |
| *"Make Public" gomb megnyomása* | A szoba megjelenik a /play oldalon lévő listán |

| *Teszt eset* | *Várt eredmény* |
|---|---|
| *"Delete" gomb megnyomása* | A játékos törli a szobát. |
| *A szoba készítője megnyomja a "Start Game" gombot úgy, hogy elegendő játékos van a szobában és egy játék ki van választva* | A játék elindul. |
| *A szoba készítője megpróbál egyedül elindítani egy játékot annak kiválasztása után, vagy egy olyan játékot, amiben túl sok játékos van.* | Nem történik semmi. |
| *A játékos begépel valamit a beszélgetés ablak szöveg bemenetébe és enter billentyűt nyom* | Az üzenet elküldésre kerül. |
| *A játékos megnyom egy reakció gombot* | A reakció elküldésre kerül. |
| *A játékos rányom egy kártyára, ami kielégíti a szabályokat úgy, hogy nem ő következik* | Nem történik semmi. |
| *A játékos rányom egy kártyára, ami kielégíti a szabályokat úgy, hogy ő következik.* | A kártya letétele a játék szerint történik. |
| *A nyerés akció bekövetkezik* | A játékosok bekerülnek egy új szobába és a nyertes győzelmeinek száma eggyel nő. |

| *Teszt eset* | *Várt eredmény* |
|---|---|
| *Olyan hivatalos játékhoz tartozó szabály vagy lánc törlésének próbája, amit nem a felhasználó készített* | Az oldal frissítése után látszik, hogy valójában nem törlődött. |

3.1. táblázat. Tesztelési tervek

A tesztelés során minden az elvárásoknak megfelelően történt.

# 4. fejezet

# Összegzés

A szakdolgozatom témájában egy többjátékos kártyajáték keretrendszer webes applikációt készítettem el. Az applikáció lehetőséget nyújt különböző francia kártyával játszott kártyajátékok készítésére és online többjátékos játszására. A kliensoldal és szerveroldal elkészítésére Next.js keretrendszert használtam, a valós idejű funkciók és adatbázis kezelésére Supabase fejlesztési platformot és a hozzá tartozó API-t. A munka során a következő kihívásokkal kellett szembesülnöm: a játékmenet meghatározásához elég részletes szabályrendszer kitalálása, a szabályrendszer használatával egy komplex játék elkészítése.

Továbbfejlesztési lehetősége lehet a rendszernek, a szabályrendszer kiegészítése két mező matematikai összegével vagy különbségével történt összehasonlításokkal.

Úgy érzem sikerült elérni a rendszer elkészítésével magam elé helyezett célt, mivel ez lehetőséget biztosít az emberi kapcsolatok ápolására a közös játék során.

# Irodalomjegyzék


[1] Supabase. *Organizáció készítése.* https://app.supabase.com/new Elérés dátuma: (2023.05.26.)

[2] *PostgreSQL.* https://www.postgresql.org/ Elérés dátuma: (2023.05.26.)

[3] Supabase. *Valós idejű támogatás engedélyezése.* https://app.supabase.com/project/_/database/replication Elérés dátuma: (2023.05.26.)

[4] Supabase. *API beállítások.* https://app.supabase.com/project/_/settings/api Elérés dátuma: (2023.05.26.)

[5] *TypeScript.* https://www.typescriptlang.org/ Elérés dátuma: (2023.05.26.)

[6] *Next.js.* https://nextjs.org/ Elérés dátuma: (2023.05.26.)

[7] *React.* https://react.dev/ Elérés dátuma: (2023.05.26.)

[8] *Tailwind CSS.* https://tailwindcss.com/ Elérés dátuma: (2023.05.26.)

[9] *Supabase.* https://supabase.com/ Elérés dátuma: (2023.05.26.)

[10] *Firebase.* https://firebase.google.com/ Elérés dátuma: (2023.05.26.)

[11] *Auth UI.* https://supabase.com/docs/guides/auth/auth-helpers/auth-ui Elérés dátuma: (2023.05.26.)

[12] *Next.js Auth Helper.* https://supabase.com/docs/guides/auth/auth-helpers/nextjs-pages Elérés dátuma: (2023.05.26.)

[13] *Supabase projekt alapú adatbázis diagram generálása.* https://supabase-schema.vercel.app/ Elérés dátuma: (2023.05.26.)

[14] *Fisher Yates modern algoritmusa.* https://en.wikipedia.org/wiki/Fisher-Yates_shuffle Elérés dátuma: (2023.05.26.)

# Ábrák jegyzéke

# Táblázatok jegyzéke

# Forráskódjegyzék