Проект №379

Тема: Тути Квантум (Tutti Quantum)

Направление: Мултимедийни приложения

Автор:

Деян Кръстев Гигов, 1052106643, гр. София ж.к. Люлин 7 бл. 769А, 0878432626, deyankg@students.npmg.org, НПМГ „Акад. Любомир Чакалов“ 9 клас

Ръководители:

Екатерина Димитрова Мицева, 0889389662, ekaterina.mitseva@npmg.org, Старши учител по Информатика и ИТ

Добринка Венциславова Златинска, 0878432181, dobrinka.zlatinska@npmg.org, Старши учител по Информатика и ИТ

1. Тема

Tutti Quantum е настолна игра за изграждане на диаграми на Файнман, вдъхновена от квантовата физика на елементарните частици. Играчите се състезават, като свързват карти с различни видове частици и образуват физически валидни взаимодействия (върхове), с цел натрупване на максимален брой точки.

2. Цели

Предназначение: Tutti Quantum е образователна и развлекателна уеб игра, чиято основна цел е да запознае играчите с концепциите на квантовата физика – по-конкретно с диаграмите на Файнман – по достъпен и занимателен начин. Приложението е предназначено за ученици и студенти, интересуващи се от физика, но също и за всички, които обичат стратегически игри.

Анализ на потребностите: Квантовата физика се преподава предимно чрез тежки учебници и формули, което затруднява визуалното и интерактивното осмисляне на взаимодействията между частиците. Липсват достъпни, безплатни и онлайн инструменти, които да превръщат абстрактните понятия в игрови изживявания.

Съществуващи решения: Настолната игра *Tutti Quantum*, разработена от EPFL+ECAL LAB & Helvetiq, представя оригиналната механика на играта, но е достъпна само като физически продукт. Цифровите алтернативи са ограничени и не предоставят някои по-разширени функции. Проектът запълва тази празнина, като предлага безплатна уеб-базирана реализация.

3. Основни етапи в реализирането на проекта

1. **Проучване и анализ (седмица 1–2):** Запознаване с правилата на оригиналната настолна игра; проучване на диаграмите на Файнман и физическите правила за валидни върхове; дефиниране на изисквания към приложението.
2. **Проектиране на архитектурата (седмица 3):** Избор на технологичен стек (React / TypeScript / Vite / Supabase); дефиниране на типовете данни, логиката на играта и структурата на базата данни.
3. **Реализация на основната логика (седмица 4–6):** Имплементация на хексагоналната решетка, правилата за поставяне на карти, изчисляването на резултата и валидирането на върховете.
4. **Разработка на потребителски интерфейс (седмица 7–9):** Изграждане на всички екрани – начален, локална игра, игра срещу ИИ, онлайн игра, класация, как се играе; добавяне на анимации и частично визуализиране.
5. **Интеграция на изкуствен интелект (седмица 10):** Разработка на алгоритъма за ИИ с три нива на сложност (лесно, средно, трудно).
6. **Онлайн мултиплейър и автентикация (седмица 11–12):** Настройване на Supabase за автентикация на потребители, съхранение на профили и управление на онлайн стаи за игра в реално време.

7. **Тестване и отстраняване на грешки (седмица 13–14):** Написване на unit тестове, ръчно тестване на всички сценарии, оптимизиране на производителността.

4. Ниво на сложност на проекта

Проектът е с **висока степен на сложност** поради следните основни предизвикателства:

- **Хексагонална решетка:** Работата с аксиални координати (q, r) за позициониране и намиране на съседи в шестоъгълна мрежа изисква специализирана математика и внимателна имплементация.
- **Валидиране на върхове:** Определянето кои три линии на частици се срещат в дадена точка и дали комбинацията е физически валидна изисква сложна геометрична логика.
- **ИИ алгоритъм:** Намирането на оптимален ход сред всички комбинации от карти, позиции и ротации за три нива на сложност представлява предизвикателство от областта на комбинаторната оптимизация.
- **Онлайн мултиплейър в реално време:** Синхронизирането на игровото състояние между множество клиенти чрез Supabase Realtime изисква управление на конкурентни промени и консистентност на данните.
- **Потребителски интерфейс:** Интерактивното визуализиране на хексагонална дъска с анимации и поддръжка на различни устройства (responsive design) е технически сложна задача.

5. Логическо и функционално описание на решението

Приложението е изградено като **Single Page Application (SPA)** с клиентска рендиране и сървърна поддръжка чрез BaaS (Backend-as-a-Service). Архитектурата включва следните основни модули:

Модул	Файлове	Функции
Маршрутизация	App.tsx	Дефинира всички маршрути на приложението чрез React Router: начална страница, регистрация, вход, трите режима на игра, класация и помощна страница.
Игрова логика	src/lib/gameLogic.ts	Съдържа всички типове данни (ParticleCard, GridPosition, Vertex, PlacedCard, GameState, PlayerState), конфигурациите на частиците, правилата за валидни върхове, функциите за инициализиране на играта, поставяне и завъртане на карти, изчисляване на резултата и напредване на играта.

ИИ играч	src/lib/aiPlayer.ts	Реализира алгоритъм за намиране на най-добър ход при три нива на сложност: лесно (случаен ход), средно (greedy – избира хода с максимален незабавен резултат), трудно (задълбочено претърсване с оценка на бъдещи позиции).
Автентикация	src/contexts/AuthContext.tsx	Управлява сесията на потребителя чрез Supabase Auth; предоставя контекст за текущия потребител и профила му на всички компоненти.
Онлайн игра	src/pages/OnlineGame.tsx	Управлява създаване и присъединяване към стаи за игра; синхронизира игровото състояние в реално време чрез Supabase Realtime (Postgres Changes & Broadcast).
Локална игра	src/pages/LocalGame.tsx	Реализира режима „подай и играй" за 2–4 играча на едно устройство; управлява настройката на играта, хода на играчите и края на играта.
Игра срещу ИИ	src/pages/AIGame.tsx	Предоставя игра на един играч срещу компютърен противник; интегрира модула за ИИ и визуализира мисленето на компютъра.
UI компоненти	src/components/GameBoard.tsx, ParticleCard.tsx, PlayerHand.tsx, ScoreDisplay.tsx, ParticleBackground.tsx	GameBoard визуализира хексагоналната дъска и обработва кликовете за поставяне; ParticleCard рендира отделна карта с частица; PlayerHand показва ръката на играча; ScoreDisplay показва точките; ParticleBackground добавя анимиран фон с частици.
База данни	supabase/migrations/	Съдържа SQL миграции за таблиците profiles (потребителски профили), game_rooms (стаи за онлайн игра) и leaderboard (класация).

Взаимодействия между модулите: Страниците (pages) използват игровата логика от *gameLogic.ts* за всички изчисления; онлайн играта изпраща/получава актуализации на GameState чрез Supabase Realtime; автентикационният контекст се консумира от всички страници, изискващи вход. UI компонентите са изцяло представителни (presentational) и получават данни чрез props.

6. Реализация

❖ Технологичен стек и обосновка:

- **React 18 + TypeScript:** Компонентно-базираният подход и строгото типизиране намаляват грешките и улесняват поддръжката на код с комплексна игрова логика.
- **Vite:** Ултра-бърза среда за разработка и оптимизирано production bundle.
- **Tailwind CSS + shadcn/ui:** Бързо изграждане на достъпен и responsive интерфейс с готови компоненти (диалози, карти, бутони, раздели).
- **Supabase:** Предоставя PostgreSQL база данни, автентикация, real-time subscriptions и Row Level Security – всичко без нужда от отделен backend сървър.
- **TanStack Query:** Управление на сървърното състояние и кеширане на данни (класация, профили).
- **React Router DOM:** Клиентска маршрутизация за SPA навигация.
- **Vitest + Testing Library:** Единично тестване на игровата логика и компонентите.

❖ Ключови алгоритми:

- **Аксиални хексагонални координати:** Използват се координатите (q, r) за представяне на позициите в решетката; намирането на 6-те съседа се реализира чрез фиксирани вектори на отместване.
- **Детектиране на върхове:** За всяка поставена карта алгоритъмът проверява всяка от шестте ребра; когато линиите от три карти се срещат в обща точка, се формира връх и се проверява неговата валидност спрямо таблица с разрешени комбинации.
- **ИИ greedy алгоритъм (средно ниво):** За всяка налична карта и всяка валидна позиция се изчислява разликата в резултата преди и след хода; избира се максимумът.
- **ИИ с предвиждане (трудно ниво):** Разширява greedy алгоритъма, като оценява и потенциалните бъдещи позиции, намалявайки честотата на невалидни върхове.

❖ Използвана литература и ресурси:

- Настолна игра *Tutti Quantum* – EPFL+ECAL LAB & Helvetiq
- Redblobgames – „Hexagonal Grids" (https://www.redblobgames.com/grids/hexagons/)
- Официална документация на React, Vite, Supabase, Tailwind CSS
- Feynman, R. P. – *QED: The Strange Theory of Light and Matter*
- Physics icon by Icons8

7. Описание на приложението

Инсталиране и стартиране:

1. Клонирайте хранилището:
   git clone https://github.com/dido739/tutti-quantum.git или използвайте архива
2. Инсталирайте необходимите пакети: ***npm install***

3. Копирайте ***.env.example*** като ***.env*** и попълнете данните за Supabase проект (URL и анонимен ключ).
4. Стартирайте dev сървъра: ***npm run dev***
5. Отворете браузър на посочения адрес

Production build: ***npm run build*** създава оптимизирани файлове в папка ***dist/***, готови за хостване на всяка статична хостинг платформа (Netlify, Vercel, Lovable и др.).

Как се използва:

- **Начална страница:** Показва три бутона за режими на игра (срещу ИИ, локален мултиплейър, онлайн), бутон „Как се играе" и бутон към класацията. Незарегистрираните потребители могат да играят локално и срещу ИИ без регистрация.
- **Регистрация / Вход:** Попълнете имейл и парола, за да получите достъп до онлайн мултиплейъра и класацията (Трябва да потвърдите имейла си! Ще получите линк за потвърждение на него).
- **Игра срещу ИИ:** Изберете ниво на сложност (лесно / средно / трудно) и въведете своето потребителско име. Играта стартира на два реда – наличните карти в центъра и вашата дъска. Изберете карта, завъртете я при нужда и кликнете на свободна клетка от дъската, за да я поставите.
- **Локален мултиплейър:** Изберете 2–4 играча и въведете имената им. Играта е от типа „подай и играй" – всеки играч взима своя ход на общото устройство.
- **Онлайн мултиплейър:** Създайте нова стая (получавате код за присъединяване) или въведете код на съществуваща стая. Играта се синхронизира в реално време между всички участници.
- **Как се играе:** Страницата обяснява правилата, видовете частици, валидните комбинации и системата за точкуване с интерактивни раздели.

Поддръжка:

- Промени в схемата на базата данни се правят чрез Supabase миграции в папка ***supabase/migrations/***.
- Тестовете се изпълняват с ***npm test*** (Vitest).
- Линтерът се стартира с ***npm run lint*** (ESLint + TypeScript ESLint).

8. Заключение

Основен резултат: Разработена е функционална уеб-базирана карточна игра, вярно имплементираща правилата на ***Tutti Quantum*** с три независими режима – локален мултиплейър, игра срещу ИИ и онлайн мултиплейър в реално време. Приложението е достъпно безплатно на всяко устройство с браузър, без нужда от инсталация.

Приложения до момента: Проектът е разработен и тестван от авторът и неговите съученици като образователен инструмент. Приложението е хостнато и достъпно онлайн чрез платформата Vercel.

Възможности за развитие и усъвършенстване:

- Добавяне на **кооперативен режим**, в който играчите заедно изграждат общ Файнман граф.
- Разширяване на **ИИ алгоритъма** с машинно обучение (reinforcement learning) за по-предизвикателна игра на трудно ниво.
- Въвеждане на **обучителни нива** с насоки за валидни взаимодействия, насочени към ученици, изучаващи физика.
- Поддръжка на **турнирен режим** + таблица с резултати на живо.
- **Мобилно приложение** (React Native / Capacitor) за iOS и Android.
- Разширяване на **базата данни с физически частици** с нови типове (бозони W/Z, неутрино) за по-сложни вертекс правила.

Данни за уеб приложението:

URL: https://tutti-quantum.vercel.app/
User1: test1@test.bg
Pass1: 123456
User2: test2@test.bg
Pass2: 123456