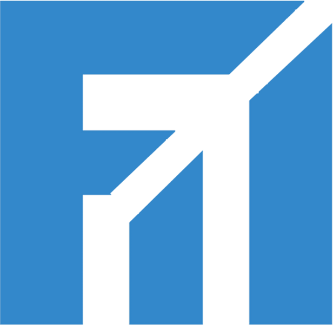
**UNIVERSITATEA “ALEXANDRU IOAN CUZA” DIN IAȘI**

**FACULTATEA DE INFORMATICĂ**



**LUCRARE DE LICENȚĂ**

**PokeTrader**

**Aplicație web pentru facilitarea schimburilor de cărți Pokémon între colecționari**

**propusă de:**

**Vadana Andrei-Ștefan**

**Sesiunea: iulie, 2025**

Coordonator științific

**Lect. dr. Moruz Alex**

**UNIVERSITATEA “ALEXANDRU IOAN CUZA” DIN IAȘI**

**FACULTATEA DE INFORMATICĂ**

**PokeTrader**

**Aplicație web pentru facilitarea schimburilor de cărți Pokémon între colecționari**

**propusă de:**

**Vadana Andrei-Ștefan**

**Sesiunea: iulie, 2025**

Coordonator științific

**Lect. dr. Moruz Alex**

**Cuprins**

**1. Introducere**

1.1 Motivație

1.2 Contribuții proprii

1.3 Structura lucrării

**2. Analiza problemei**

2.1Comunitatea colecționarilor Pokémon în România

2.2 Modalități actuale de schimb

2.3 Aplicații similare și abordări existente

2.4 Concluzii

**3. Analiza cerințelor**

3.1 Cerințe funcționale

3.2 Cerințe nefuncționale

**4. Noțiuni utilizate**

4.1Arhitecturaclient–server

4.2 REST API

4.3 JSON Web Token

4.4 Cross-Origin Resource Sharing

**5. Tehnologii folosite**

5.1 Django

5.2 React

5.3 SQLite

5.4 Pokémon TCG API

5.5 PokéAPI

**6. Implementarea aplicației**

6.1 Arhitectura

6.2 Funcționalitățile

6.2.1 Înregistrare și autentificare

6.2.2 Adăugare de cărți în colecție

6.2.3 Vizualizarea colecției și propunerea schimburilor

6.2.4 Vizualizarea și acceptarea de schimburi

6.2.5 Chatul dintre utilizatori

6.2.6 Minigame-ul “Guess the Pokémon”

6.2.7 Vizulizarea schimburilor anterioare și sistemul de rating

6.3 Implementarea funcționalităților principale

6.3.1 Guess the Pokémon

6.3.2 Sistemul de Trade

6.3.3 Sistemul de chat dintre utilizatori

6.3.4 Sistemul de rating

6.3.5 Login, Register și ProtectedRoute

6.4 Fluxul utilizatorului în aplicație

**7. Concluzii și direcții de dezvoltare viitoare**

7.1 Direcții de dezvoltare viitoare

7.2 Concluzii finale

**8. Bibliografie**

**1. Introducere**

**1.1 Motivație**

Fiind parte din comunitatea Pokémon din România, am realizat că, în prezent, singura modalitate de a face schimb de cărți este prin intermediul grupurilor de Facebook. Acest aspect mi s-a părut descurajator, mai ales pentru persoanele care abia descoperă acest hobby și nu sunt încă familiarizate cu dinamica comunității. Consider că acest lucru contrazice însăși esența unui joc de tip Trading Card Game, precum Pokémon, al cărui scop principal este tocmai realizarea de schimburi între colecționari.

Această observație m-a motivat să dezvolt o aplicație dedicată schimburilor de cărți Pokémon, care să ofere o alternativă digitală modernă, sigură și ușor de utilizat. Mi-am dorit ca aplicația să aibă o interfață prietenoasă, intuitivă, accesibilă atât colecționarilor experimentați, cât și celor care abia încep să se familiarizeze cu universul acestui joc.

Aplicația oferă posibilitatea de a-ți vizualiza colecția, de a marca ce cărți dorești să dai sau să primești, de a căuta cărți după set sau nume și de a propune schimburi altor utilizatori în mod direct, eliminând incertitudinea și lipsa de organizare din metodele tradiționale.

Pe lângă faptul că rezolvă o problemă reală din cadrul comunității Pokémon, mi-am dorit ca această aplicație să mă ajute să mă familiarizez cu tehnologiile web utilizate, precum Django, React și altele.

Îmi doresc ca aplicația să contribuie la întărirea legăturilor dintre colecționari, să încurajeze interacțiunile dintre aceștia și să ofere un spațiu virtual organizat, care să reflecte esența autentică a unui joc de tip Trading Card Game.

**1.2 Contribuții proprii**

În realizarea acestei lucrări de licență, am conceput și implementat o aplicație web complet funcțională care facilitează schimbul de cărți Pokémon între colecționari, contribuind astfel la dezvoltarea comunității locale. În prezent, nu există aplicații active dedicate exclusiv acestui scop în România, ceea ce accentuează utilitatea și unicitatea aplicației dezvoltate.

Aplicația este compusă din două componente principale:

* Frontend-ul, realizat în React, oferă o interfață prietenoasă, adaptată atât pentru desktop, cât și pentru dispozitive mobile. Am dezvoltat componente precum navbar-ul responsive, afișarea cardurilor pe seturi, colecția personală cu posibilitatea de a adăuga sau elimina cărți, precum și o secțiune dedicată schimburilor.
* Backend-ul, construit cu Django și Django REST Framework, gestionează logica aplicației și comunicarea cu baza de date. Am realizat modele personalizate pentru cărți, seturi, schimburi, mesaje și altele. De asemenea, am implementat un sistem de autentificare bazat pe JWT, asigurând o securitate sporită pentru conturile utilizatorilor.

Pe partea de funcționalitate, am contribuit cu:

* Un sistem complet de gestionare a colecției proprii: adăugarea, eliminarea și vizualizarea cărților.
* Crearea și vizualizarea schimburilor, unde utilizatorii pot selecta ce oferă și ce doresc să primească.
* Introducerea unui sistem de coins care pot fi obținuți prin jucarea zilnică a unui joc de tipul “Guess the Pokemon”, pentru a ajuta utilizatorii care nu au încă cărți în colecție.
* Mecanisme de acceptare și confirmare a schimburilor, inclusiv blocarea temporară a cărților implicate.
* Sistem de chat între utilizatori pentru negocierea și stabilirea modului în care va decurge schimbul.
* Un sistem de rating pentru evaluarea celuilalt utilizator care a luat parte la schimb.
* Pagină de profil cu afișarea rating-ului și a schimburilor finalizate anterior.

**1.3 Structura lucrării**

Această lucrare este structurată pe opt capitole, fiecare având rolul de a documenta procesul de analiză și dezvoltare al aplicației PokeTrader.

Capitolul 1: **Introducere**, introduce tema lucrării, motivația personală pentru alegerea subiectului, contribuțiile proprii aduse proiectului, precum și structura generală a documentației.

Capitolul 2: **Analiza problemei**, analizează problema reală identificată în cadrul comunității de colecționari Pokémon, prezentând modalitățile actuale de schimb, lipsurile acestora și o scurtă analiză a aplicațiilor similare existente.

Capitolul 3: **Analiza cerințelor**, detaliază cerințele aplicației, atât din punct de vedere funcțional, cât și nefuncțional, stabilind astfel direcția dezvoltării.

Capitolul 4: **Noțiuni utilizate**, este dedicat noțiunilor teoretice necesare înțelegerii aplicației: arhitectura client–server, REST API, JWT și CORS.

Capitolul 5: **Tehnologii folosite,** prezintă tehnologiile utilizate, justificând alegerea acestora și modul în care au fost folosite în cadrul aplicației.

Capitolul 6: **Implementarea aplicației,** descrie în detaliu procesul de implementare, incluzând arhitectura aplicației, funcționalitățile disponibile, modul de funcționare al acestora, precum și fluxul tipic al unui utilizator în aplicație.

Capitolul 7: **Concluzii și direcții de dezvoltare viitoare**, conține propuneri de dezvoltare viitoare care pot îmbunătăți experiența utilizatorilor, precum și concluziile finale privind utilitatea aplicației și procesul de învățare parcurs.

Capitolul 8: **Bibliografie,** cuprinde bibliografia utilizată în realizarea acestei lucrări.

**2. Analiza problemei**

**2.1 Comunitatea colecționarilor Pokémon în România**

Franciza Pokémon a devenit populară în România în anii 2000, în special prin intermediul desenelor animate și a jocurilor video. Cu timpul, jocurile de tip Trading Card Game, precum Yu-Gi-Oh!, Duel Masters, Magic: The Gathering și Pokémon, au câștigat o comunitate pasionată, deși mai restrânsă decât în alte țări. M-am alăturat acestei comunități în 2021, datorită pasiunii mele pentru obiecte de colecție, în special cărțile Pokémon, și din dorința de a cunoaște și interacționa cu alte persoane care îmi împărtășesc entuziasmul.

Comunitatea din România este activă în principal pe grupuri de Facebook sau Discord, însă nu există o platformă dedicată exclusiv colecționarilor care doresc să facă schimburi de Pokémon Cards. Acest lucru îngreunează organizarea acestora, verificarea autenticității cardurilor și interacțiunile între colecționari.

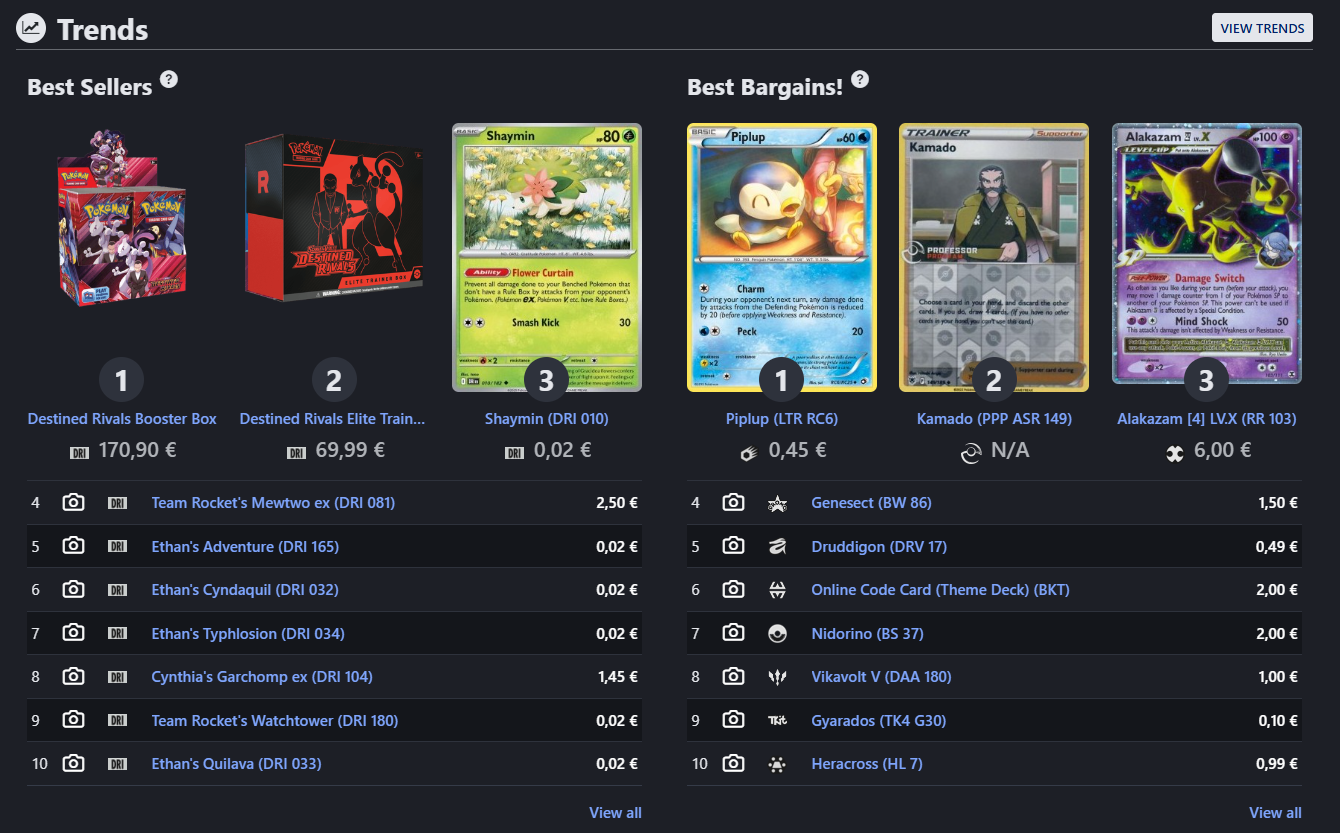
**2.2 Modalități actuale de schimb**

În prezent, membrii comunității Pokémon care doresc să realizeze schimburi cu cărțile de joc din colecțiile proprii sunt nevoiți să apeleze la grupuri de Facebook. Problema este că aceste grupuri sunt axate în principal pe vânzarea și cumpărarea cardurilor, iar partea de schimb este adesea neglijată. Acest aspect poate descuraja utilizatorii care își doresc să experimenteze adevărata esență a unui joc de tip Trading Card Game, bazat pe interacțiune și schimburi între colecționari.

Prin intermediul acestor grupuri, utilizatorii nu au un sistem centralizat prin care să verifice istoricul sau reputația celor cu care interacționează, ceea ce poate duce la ezitări sau neînțelegeri în procesul de schimb. Pe lângă asta, nu li se oferă un sistem de căutare eficient pe baza setului, rarității sau valorii cardurilor, ceea ce face dificilă identificarea rapidă a unui potențial partener de schimb.

**2.3 Aplicații similare și abordări existente**

Deși în prezent există aplicații web consacrate pentru colecționarii de cărți Pokémon, precum Cardmarket sau TCGPlayer, acestea sunt orientate în mod clar către comerțul cu cărți, având ca scop principal vânzarea și cumpărarea acestora. Utilizatorii pot crea anunțuri, verifica prețurile de piață și achiziționa cărți direct de la alți colecționari sau de la comercianți.



**Screenshot de pe site-ul Cardmarket**

Cu toate acestea, funcționalitatea de tip "trading" — adică schimbul direct de cărți între utilizatori fără implicarea unei tranzacții financiare — este inexistentă sau neglijată pe aceste platforme. Acest aspect face ca experiența oferită să fie incompletă din perspectiva unui joc de tip Trading Card Game, al cărui scop principal este tocmai facilitarea schimburilor.

**2.4 Concluzii**

În urma analizei realizate, se poate observa o lipsă clară a unei platforme dedicate schimburilor de cărți Pokémon. Deși există site-uri populare pentru comerțul cu astfel de cărți, acestea nu răspund nevoilor colecționarilor care doresc să realizeze tranzacții bazate exclusiv pe schimb, într-un mod sigur, transparent și ușor de gestionat.

Aplicația PokeTrader vine ca un răspuns direct la această problemă, oferind o alternativă digitală modernă care facilitează schimburile de cărți într-un mod organizat, accesibil și adaptat specific pentru comunitatea Pokémon. Prin introducerea unor funcționalități precum profiluri de utilizator, rating și sistem de chat pentru negociere, aplicația urmărește să încurajeze interacțiunea între colecționari și să redefinească experiența de schimb, aducând-o mai aproape de spiritul autentic al unui Trading Card Game.

**3. Analiza cerințelor**

Pentru a răspunde nevoilor identificate în cadrul comunității colecționarilor Pokémon, aplicația PokeTrader trebuie să îndeplinească o serie de cerințe funcționale și nefuncționale. Scopul este crearea unei platforme ușor de utilizat, care să faciliteze schimbul de cărți într-un mod organizat, sigur și eficient.

**3.1 Cerințe funcționale**

* **Înregistrare și autentificare utilizator**

Utilizatorii trebuie să-și poată crea un cont, să se autentifice și să gestioneze în siguranță accesul la propriul profil.

* **Vizualizarea și gestionarea colecției personale**

Utilizatorii își pot adăuga, elimina și organiza cărțile Pokémon deținute într-o secțiune personală.

* **Crearea și vizualizarea propunerilor de schimb**

Se pot iniția oferte de schimburi între utilizatori, selectând cărțile oferite și cele dorite.

* **Chat între utilizatori**

Odată ce doi utilizatori sunt implicați într-un schimb, aceștia pot discuta detaliile semnificative prin intermediul sistemului de chat.

* **Acceptarea și confirmarea schimburilor**

După ce ambii utilizatori au confirmat că schimbul a avut loc, colecțiile acestora vor fi actualizate.

* **Sistem de rating**

După finalizarea unui schimb, utilizatorii pot acorda un rating celuilalt, contribuind la reputația din platformă.

* **Minigame zilnic pentru obținerea de coins**

Utilizatorii pot juca „Guess the Pokémon” pentru a câștiga coins, care pot fi folosiți în schimburi, utili în special pentru utilizatorii care încă nu dețin cărți în colecția lor.

**3.2 Cerințe nefuncționale**

* **Interfață intuitivă și responsive**

Aplicația trebuie să funcționeze fluent pe desktop, tabletă și mobil.

* **Performanță ridicată**

Căutările și operațiile frecvente trebuie să se execute rapid.

* **Securitate sporită**

Datele utilizatorilor sunt protejate prin autentificare JWT și bune practici de securitate.

* **Scalabilitate**

Platforma trebuie să permită extinderea în viitor pentru noi funcționalități sau mai mulți utilizatori.

**4. Noțiuni utilizate**

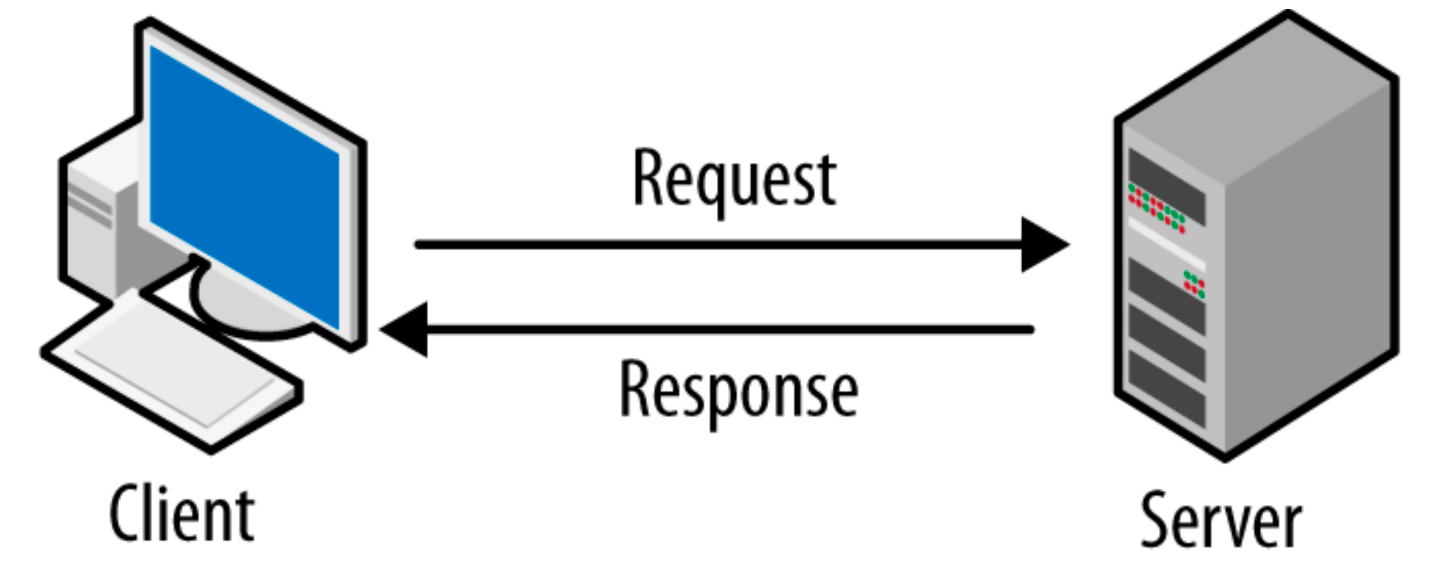
Dezvoltarea aplicației PokeTrader a presupus înțelegerea și aplicarea mai multor concepte teoretice legate de dezvoltarea aplicațiilor web, sisteme REST, etc.

**4.1 Arhitectura client-server**

Arhitectura client-server reprezintă un model clasic de organizare a aplicațiilor, în care sarcinile sunt împărțite între două componente principale: clientul și serverul. Clientul este partea care interacționează direct cu utilizatorul, fiind responsabil cu trimiterea cererilor către server și afișarea răspunsurilor primite.

Serverul gestionează logica aplicației și resursele necesare funcționării, cum ar fi datele stocate într-o bază de date. El primește cererile trimise de client, le procesează și returnează un răspuns corespunzător. Comunicarea dintre client și server se realizează, în general, prin protocolul HTTP sau HTTPS, iar datele transmise sunt, de obicei, în format JSON sau XML.

Un avantaj important al acestui model este separarea clară a responsabilităților: clientul se ocupă de interfața cu utilizatorul, iar serverul de procesarea datelor și logica aplicației. Această separare permite o dezvoltare mai organizată, scalabilitate crescută și întreținere mai facilă. De asemenea, arhitectura client-server permite ca același server să poată deservi simultan mai mulți clienți, indiferent de platforma de pe care accesează aplicația.



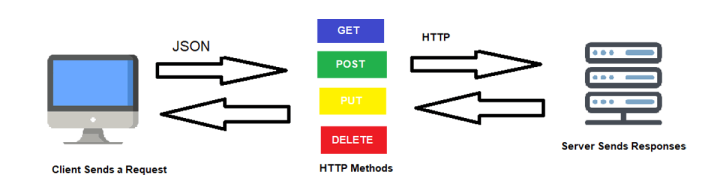
**Ilustrarea arhitecturii client-server**

**4.2 REST API**

Un API (Application Programming Interface) reprezintă o interfață care permite comunicația dintre două sisteme software. În contextul aplicațiilor web, un API facilitează schimbul de date între client și server. Unul dintre cele mai utilizate tipuri de API este API-ul REST (Representational State Transfer).

REST este un stil arhitectural care definește un set de reguli pentru crearea de servicii web. Aceste servicii RESTful sunt bazate pe protocoalele HTTP standard și folosesc concepte precum resurse, URL-uri unice pentru fiecare resursă și metode HTTP pentru interacțiune. Fiecare resursă este identificată printr-un endpoint – adică un URL specific.

Operațiile asupra resurselor se realizează prin metodele HTTP: **GET** pentru obținerea datelor, **POST** pentru crearea unei resurse noi, **PUT** sau **PATCH** pentru actualizare și **DELETE** pentru ștergere. Astfel, un API REST permite implementarea completă a operațiilor CRUD (Create, Read, Update, Delete) într-un mod standardizat și predictibil.



**Reprezentare a schimbului de date într-un API de tip REST**

**4.3 JSON Web Token**

JWT (JSON Web Token) este o metodă modernă și eficientă prin care clientul și serverul pot face schimb de informații într-un mod sigur. Informațiile sunt ambalate într-un format compact, care poate fi ușor transmis prin rețea. Un token JWT este compus din trei secțiuni: header-ul, care specifică tipul de token și algoritmul de semnare, payload-ul, care conține datele efective și o semnătură digitală, folosită pentru a garanta că tokenul nu a fost modificat.

În general, JWT este folosit în procesul de autentificare. După ce un utilizator se autentifică, serverul generează un token JWT care conține informațiile necesare și îl trimite clientului. Acest token este apoi inclus de către client în fiecare cerere ulterioară, astfel încât serverul să poată valida identitatea fără să păstreze informații despre sesiune.

**4.4 Cross-Origin Resource Sharing**

CORS (Cross-Origin Resource Sharing) este un mecanism de securitate implementat de browsere pentru a controla accesul la resursele unui server dintr-un alt domeniu decât cel al aplicației frontend.

În mod normal, o aplicație web poate face cereri HTTP doar către domeniul de pe care a fost încărcată. Dacă frontend-ul și backend-ul sunt pe domenii diferite, browserul blochează aceste cereri din motive de securitate. CORS permite totuși această comunicare, dar doar dacă serverul specifică explicit că acceptă cereri din acel alt domeniu.

**5. Tehnologii folosite**

**5.1 Django**

Django este un framework web open-source, dezvoltat în limbajul Python, care permite crearea rapidă și eficientă a aplicațiilor web. Acesta se bazează pe arhitectura MVT (Model-View-Template), un model similar cu MVC, unde componenta „Model” gestionează logica legată de date și interacțiunea cu baza de date, „View” controlează logica aplicației și răspunsurile la cereri, iar „Template” este responsabil pentru partea de interfață și afișare în browser.

Am ales să folosesc Django în dezvoltarea aplicației deoarece eram deja familiar cu Python, iar framework-ul oferă un set de unelte bine structurat, ușor de extins și foarte bine documentat. De asemenea, vine cu funcționalități integrate utile precum sistemul de autentificare, panoul de admin și suportul pentru REST API, ceea ce m-a ajutat să mă concentrez mai mult pe funcționalitățile pe care voiam să le implementez.

**5.2 React**

React este o bibliotecă JavaScript folosită pentru construirea de user interfaces. Este utilizată în principal pentru a crea aplicații web cu o interfață dinamică, într-un mod eficient și modular. React se bazează pe conceptul de componentă, adică porțiuni independente și reutilizabile de cod care descriu cum ar trebui să arate și să se comporte anumite părți ale aplicației.

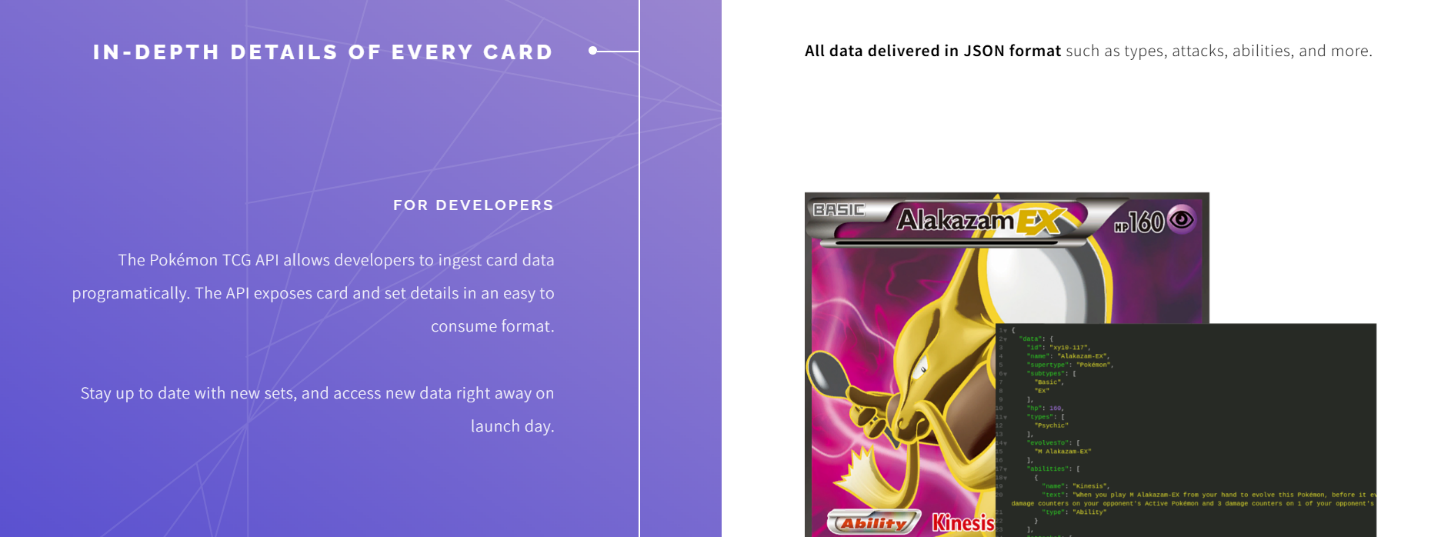
Am ales să folosesc React deoarece este una dintre cele mai populare biblioteci JavaScript în prezent și este foarte căutată în industrie. Mi s-a părut important să învăț cum funcționează și să mă obișnuiesc cu stilul componentelor și cu modul în care se gestionează starea aplicației.

**5.3 SQLite**

Pentru stocarea datelor aplicației, am folosit baza de date oferită de Django, și anume SQLite. Aceasta este o bază de date relațională, ușor de configurat, care nu necesită un server separat și salvează toate datele într-un singur fișier pe disc. Am considerat că SQLite este potrivită pentru nivelul actual al aplicației, mai ales în faza de dezvoltare și testare, întrucât oferă o soluție simplă, rapidă și integrată direct cu Django. În plus, pentru o aplicație de dimensiuni mici spre medii, SQLite este suficient de performantă și nu necesită setări suplimentare de administrare sau conectare. La nevoie, aplicația poate fi migrată ușor către alte baze de date precum PostgreSQL sau MySQL, fără modificări majore în cod.

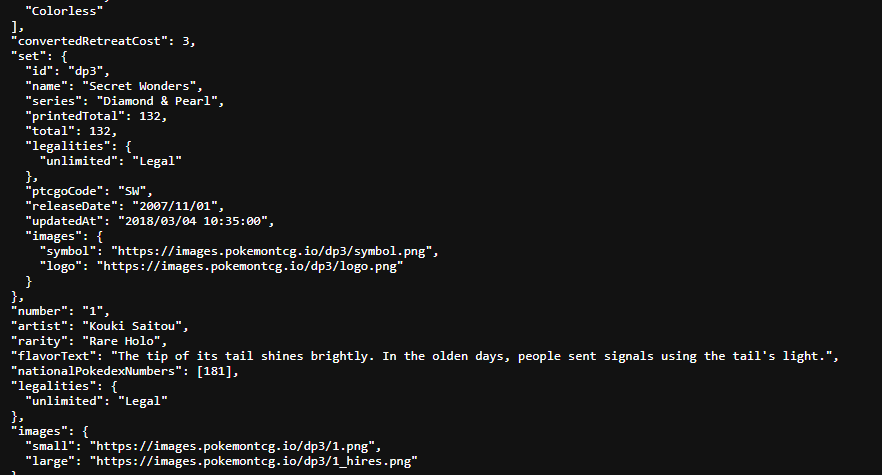
**5.4 Pokémon TCG API**

Pokémon TCG API este un serviciu public care oferă acces la o bază de date actualizată cu toate seturile și cărțile din Pokémon Trading Card Game. API-ul returnează date în format JSON și include atribute precum numele cărții, numărul din set, imaginea sau market price-ul.



**Screenshot de pe site-ul celor de la Pokémon TCG API**

Inițial, de fiecare dată când aveam nevoie de o informație din API, făceam un request direct, însă din cauza numărului limitat de requesturi și a timpului mare de încărcare la fiecare refresh al paginii, am decis să creez un tabel în baza de date, împreună cu un script care face un singur request, preia informațiile necesare și populează acest tabel. API-ul a fost esențial în dezvoltarea aplicației PokeTrader, deoarece toate imaginile, numele cărților, prețurile de piață și alte detalii sunt obținute prin intermediul său.

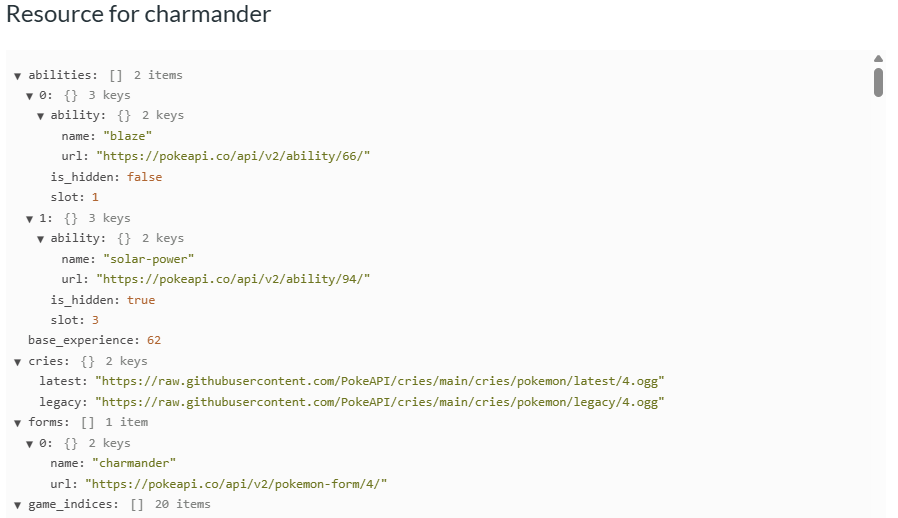


**Exemplu de response de la Pokémon TCG API**

**5.5 PokéAPI**

PokéAPI este un serviciu web gratuit care oferă un API RESTful ce permite accesul la date detaliate despre universul Pokémon, în special din seria jocurilor video. Prin intermediul acestui API, dezvoltatorii pot obține informații despre Pokémoni, mișcările acestora, abilități, tipuri și multe altele.

Am folosit acest API prin crearea unui script care populează un tabel din baza de date cu informațiile obținute în urma requesturilor. Datele furnizate de acest API m-au ajutat la realizarea minigame-ului Guess the Pokémon, deoarece am avut nevoie de mai multe trăsături pentru fiecare Pokémon, astfel încât utilizatorul să poată pune întrebări precum: care este tipul său, ce base stats are, care este înălțimea sau greutatea lui etc. Minigame-ul este util deoarece poate fi jucat zilnic și oferă utilizatorului coins, care pot fi folosiți ulterior în cadrul schimburilor.



**Exemplu de response de la PokéAPI**

**6. Implementarea aplicației**

**6.1 Arhitectura**

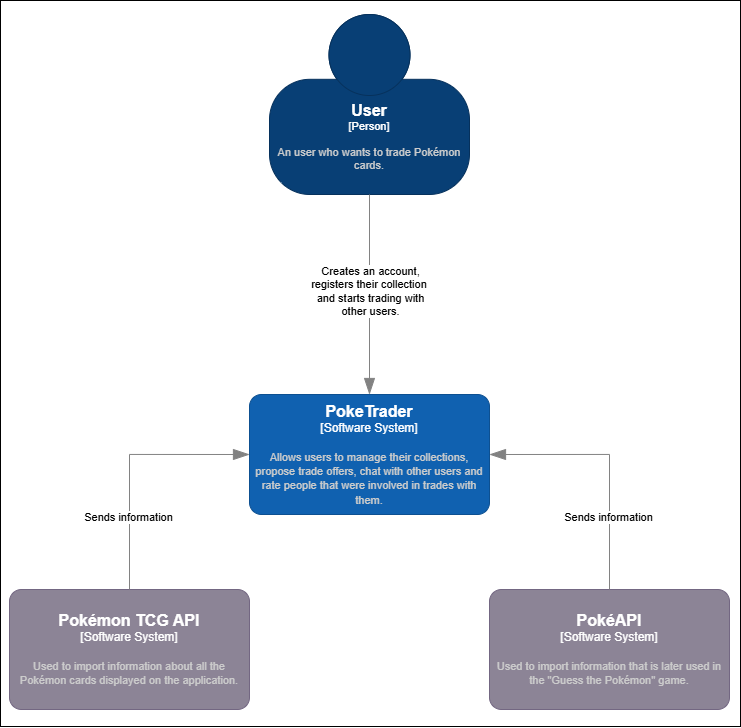
Aplicația PokeTrader este structurată după modelul client–server, unde frontend-ul este dezvoltat în React, iar backend-ul în Django. Această separare permite o dezvoltare independentă a celor două părți și o comunicare prin intermediul unui API REST.

Backend-ul este construit folosind arhitectura MVT specifică Django, dar partea de Template este înlocuită cu React, astfel că Django se comportă ca un API, nu ca un generator de pagini HTML.

Frontend-ul consumă datele oferite de backend prin apeluri HTTP, folosind librăria axios. Pentru securitate și autentificare, aplicația folosește JWT, care permite clientului să primească un token la autentificare și să îl trimită ulterior în fiecare request către server.

Datele sunt gestionate într-o bază de date SQLite. Aplicația preia informații despre cărți și pokemoni din două API-uri externe, pe care le salvează local într-o bază de date pentru a reduce numărul de request-uri și a îmbunătăți performanța.

Structura aplicației este modulară, cu separare clară între componentele backend și frontend. Această organizare face mai ușoară mentenanța și extinderea ulterioară a proiectului.



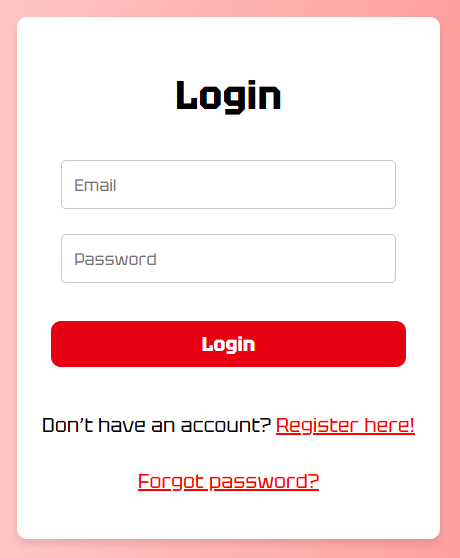
**Reprezentare de tip diagramă C4 (nivelul 1) a aplicației**

**6.2 Funcționalitățile**

În cadrul acestui subcapitol voi prezenta funcționalitățile principale ale aplicației PokeTrader, evidențiind modul în care utilizatorii pot interacționa cu platforma.

**6.2.1 Înregistrare și autentificare**

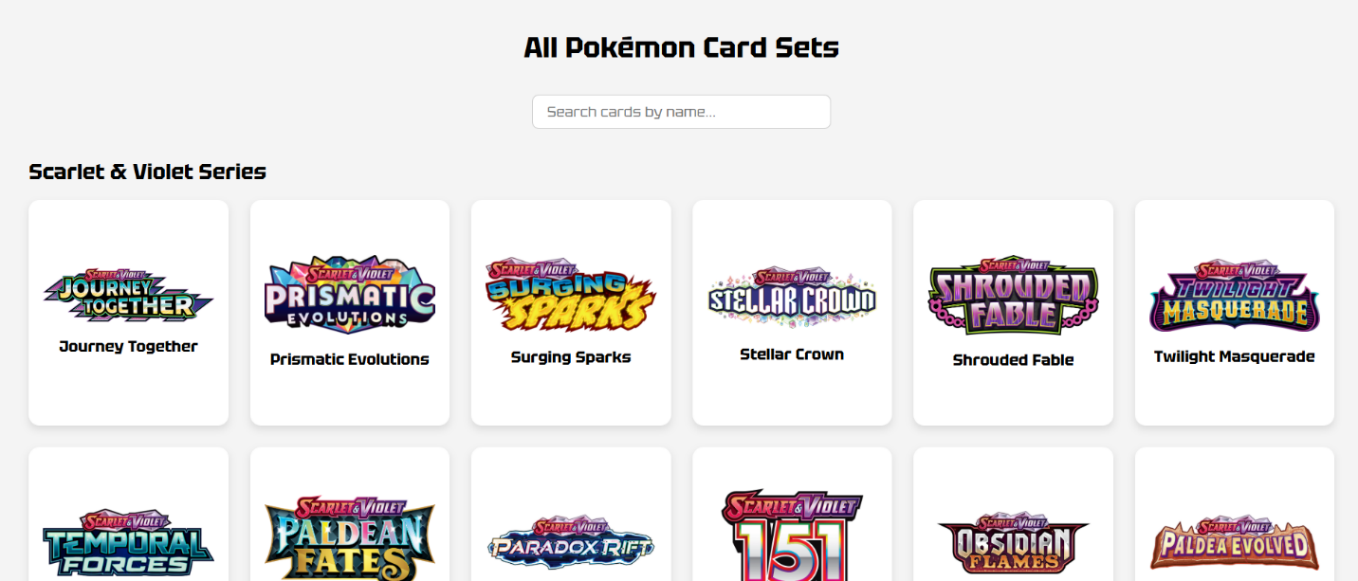
Pentru a folosi funcționalitățile principale ale aplicației, utilizatorii trebuie să își creeze mai întâi un cont prin completarea unui formular în care introduc un username, o adresă de email și o parolă. După crearea contului, utilizatorul trebuie să se autentifice folosind emailul și parola, având posibilitatea de a-și reseta parola în cazul în care a uitat-o. În urma autentificării cu succes, serverul generează un token JWT, care este stocat local în browser și este folosit pentru autorizarea accesului la endpoint-urile protejate din aplicație.



**Screenshot din aplicație cu formularul de Login**

**6.2.2 Adăugare de cărți în colecție**

Un utilizator autentificat își poate înregistra colecția fizică de cărți direct în aplicație. Prin accesarea paginii Sets, acesta are acces la toate seturile Pokémon disponibile, inclusiv cele mai recente.



**Screenshot de pe pagina Sets**

După selectarea unui set, utilizatorul poate alege ce carte dorește să adauge în colecție, precum și numărul de exemplare, folosind butoanele "+" și "−" afișate sub fiecare carte. Pe această pagină sunt afișate și informații utile despre fiecare carte, precum numărul acesteia în set sau market price-ul.

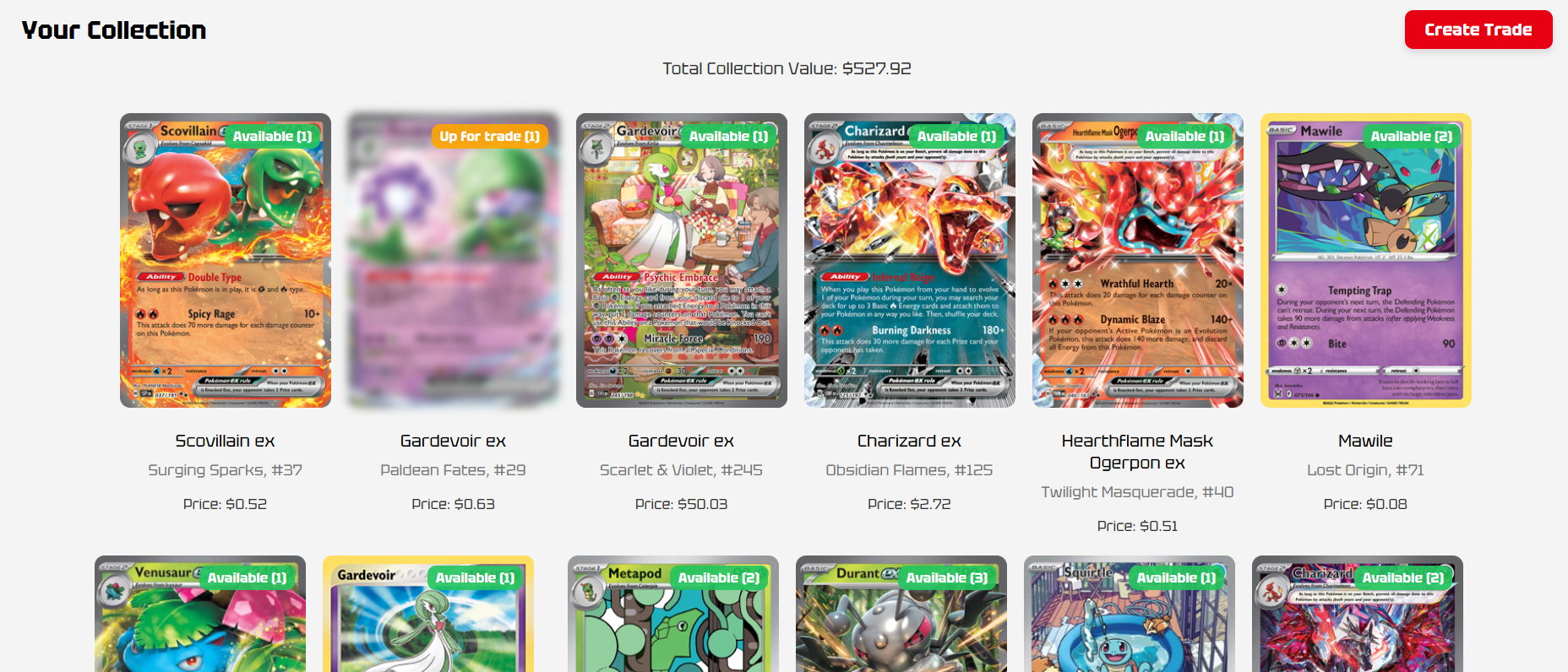
În cazul în care utilizatorul nu cunoaște setul din care face parte o anumită carte, o poate căuta direct după nume folosind bara de căutare din partea de sus a paginii.



**Screenshot cu cărțile din setul “Journey Together”**

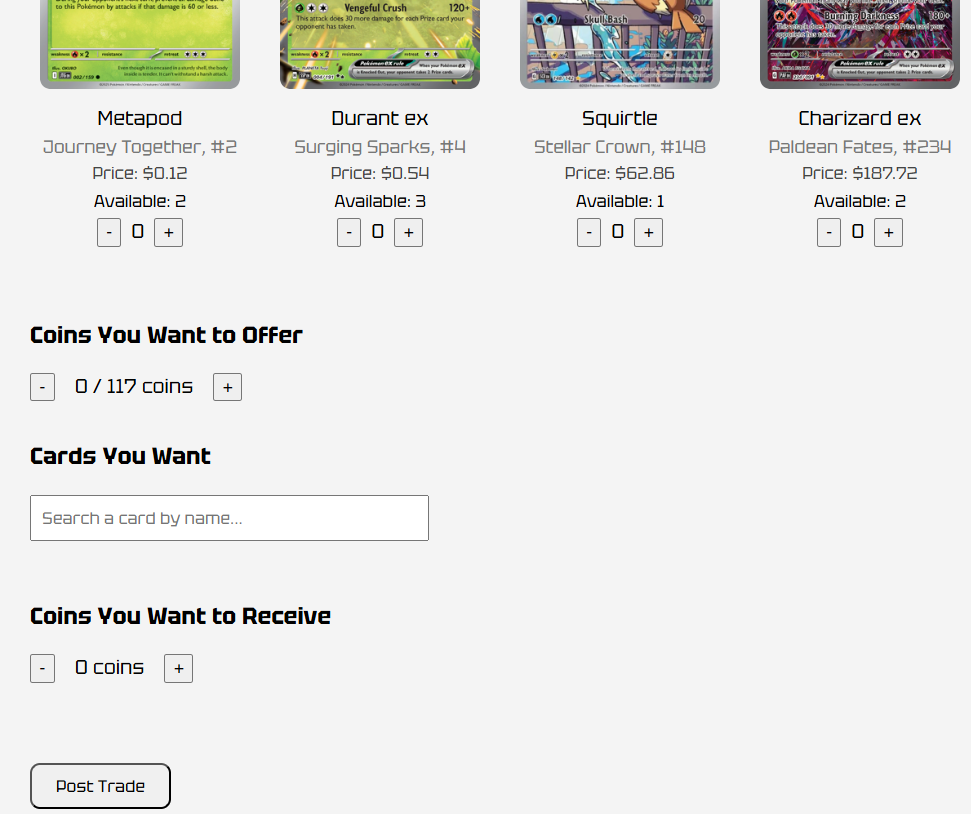
**6.2.3 Vizualizarea colecției și propunerea schimburilor**

Utilizatorii își pot vizualiza cărțile din colecție prin intermediul paginii Collection. Acolo pot vedea detalii precum câte exemplare ale unei cărți sunt încă disponibile (odată ce o instanță a unei cărți este propusă pentru un trade sau un trade care o include este acceptat, acea carte devine indisponibilă până când schimbul este finalizat, pentru a preveni folosirea ei în mai multe trade-uri simultan), care este valoarea totală a colecției, precum și alte informații similare cu cele afișate în pagina setului care include acea carte.



**Screenshot de pe pagina Collection**

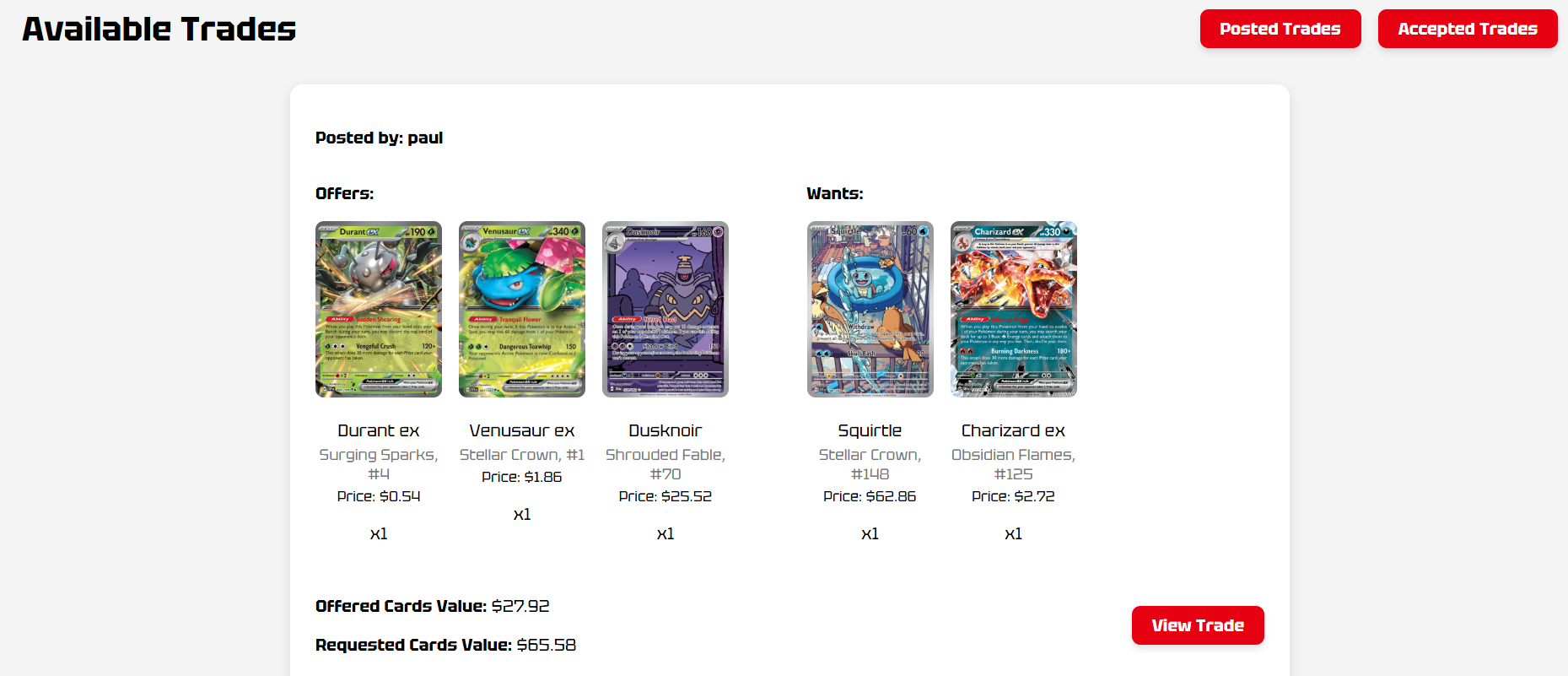
Tot prin intermediul paginii Collection, utilizatorul poate crea un trade. Pentru aceasta, trebuie să selecteze ce cărți din colecția sa (și eventual coins) dorește să ofere, precum și ce cărți (sau coins) își dorește în schimb. După finalizarea procesului, cărțile selectate pentru oferire vor apărea ca „Up for trade” în pagina Collection.

****

**Screenshot de pe pagina TradeForm**

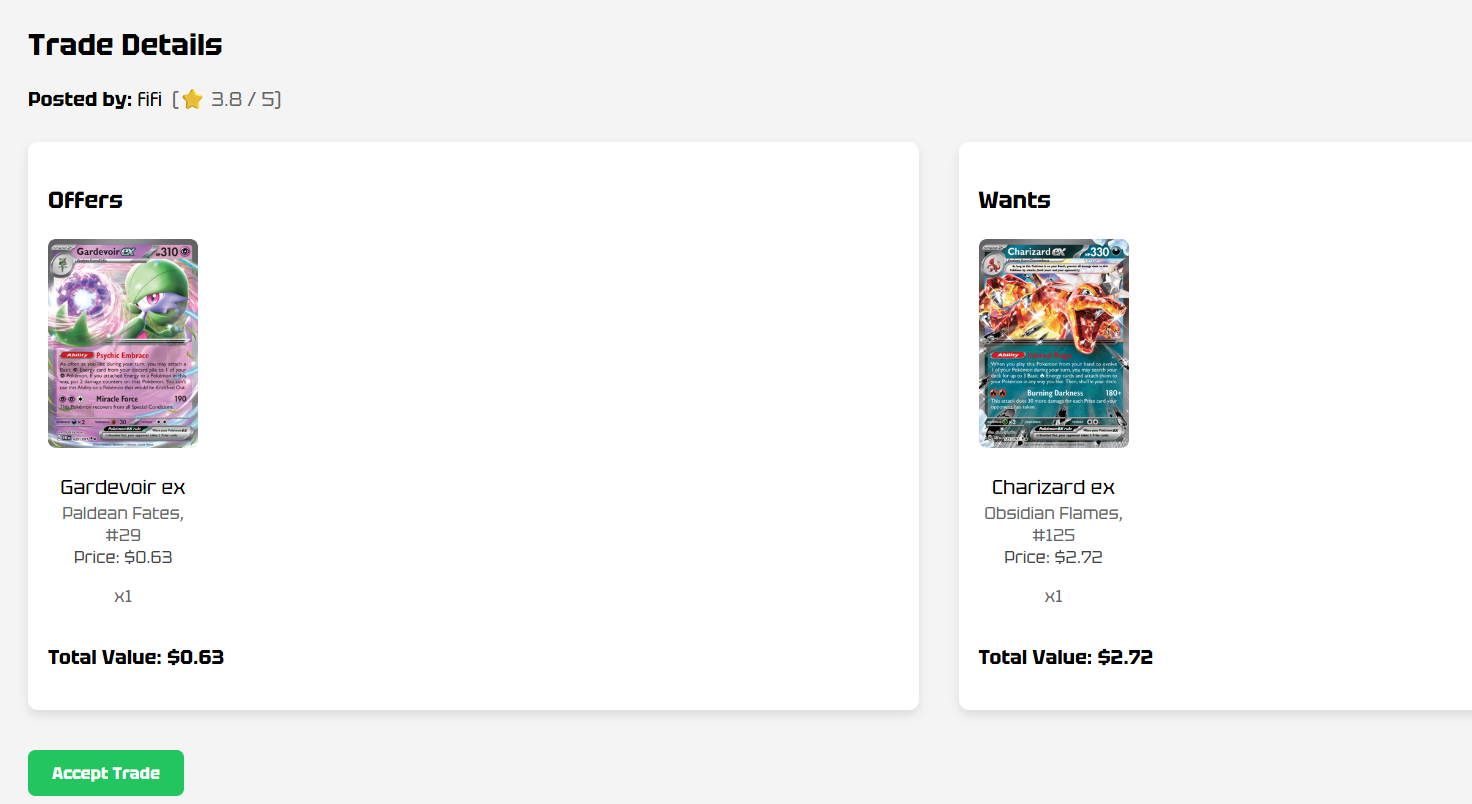
**6.2.4 Vizualizarea și acceptarea de schimburi**

Pentru a vizualiza schimburile propuse de alți utilizatori, poate fi folosită pagina Trades, unde sunt afișate cele mai recente trade-uri, împreună cu detalii despre acestea, precum setul din care fac parte cărțile implicate, market price-ul fiecărei cărți, valoarea totală a cărților oferite și a celor cerute și cine a postat trade-ul.



**Screenshot de pe pagina Trades**

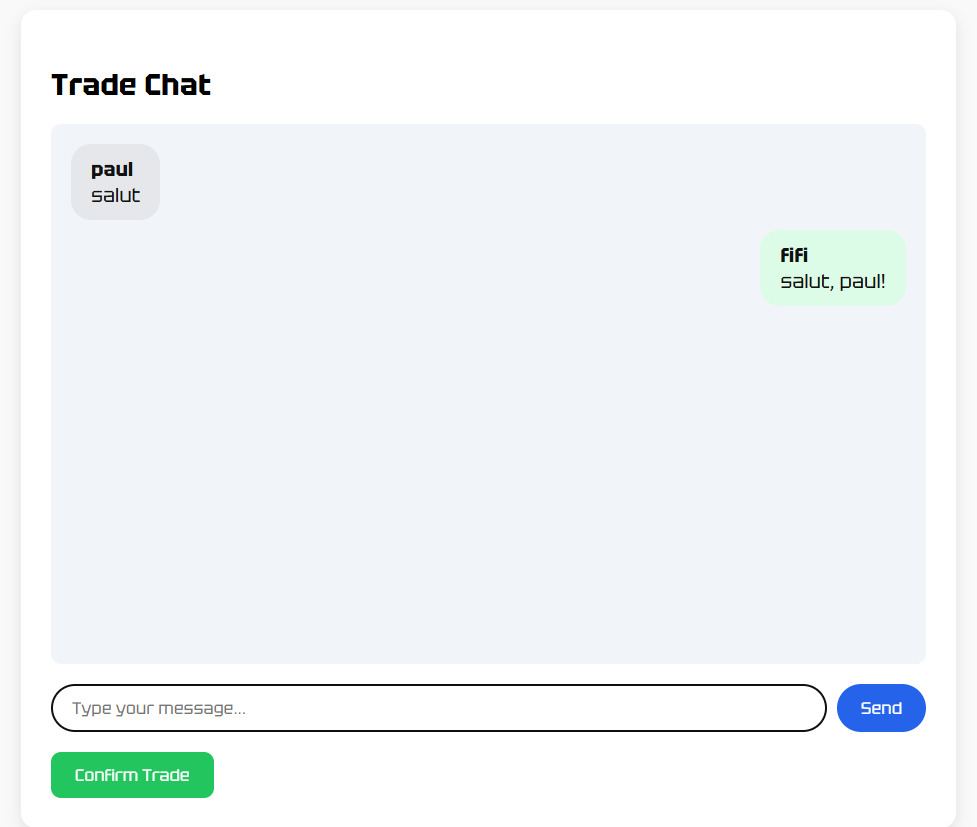
Prin apăsarea butonului „View trade”, utilizatorul poate vedea ratingul celui care a postat acel trade și, dacă deține cărțile necesare, îl poate accepta. De pe pagina Trades pot fi accesate și paginile Posted Trades, unde sunt afișate trade-urile propuse de utilizatorul logat, și Accepted Trades, unde se găsesc trade-urile care au fost acceptate de acesta și așteaptă confirmare.



**Screenshot cu detaliile unui trade**

**6.2.5 Chatul dintre utilizatori**

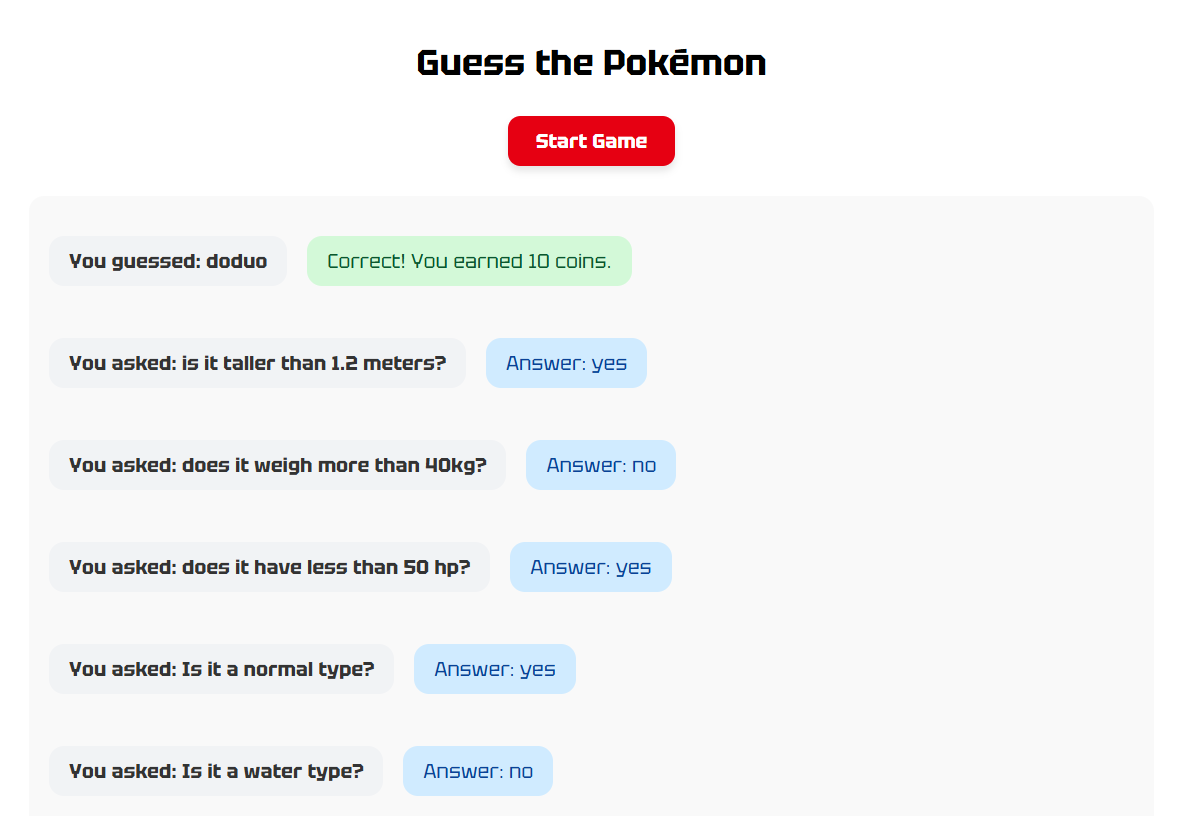
După ce un trade a fost acceptat, cei doi utilizatori implicați pot comunica între ei în pagina de chat, pentru a stabili detaliile legate de desfășurarea schimbului. După ce amândoi confirmă că trade-ul a avut loc, colecțiile lor virtuale sunt actualizate. Astfel, sistemul asigură că fiecare carte oferită sau primită este gestionată corect și nu mai este disponibilă pentru alte trade-uri.



**Screenshot cu chatbox-ul dintre doi utilizatori**

**6.2.6 Minigame-ul “Guess the Pokémon”**

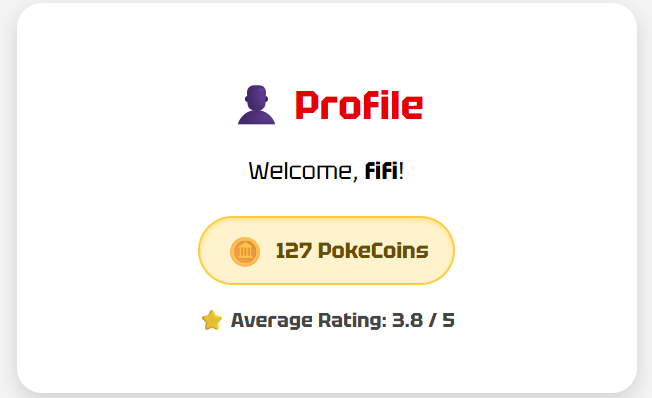
„Guess the Pokémon” este un minigame implementat pentru a putea fi jucat de utilizatorii care nu dețin încă o colecție fizică de cărți, dar nu se adresează exclusiv acestora. Utilizatorul trebuie să ghicească un Pokémon din prima generație, punând o serie de întrebări succesive despre caracteristicile acestuia: tipul, înălțimea, greutatea sau alți parametri precum HP, Attack, Defense etc. Jocul poate fi jucat o dată la 24 de ore, iar dacă utilizatorul ghicește corect Pokémonul, primește 10 coins, care pot fi ulterior folosiți în cadrul schimburilor.

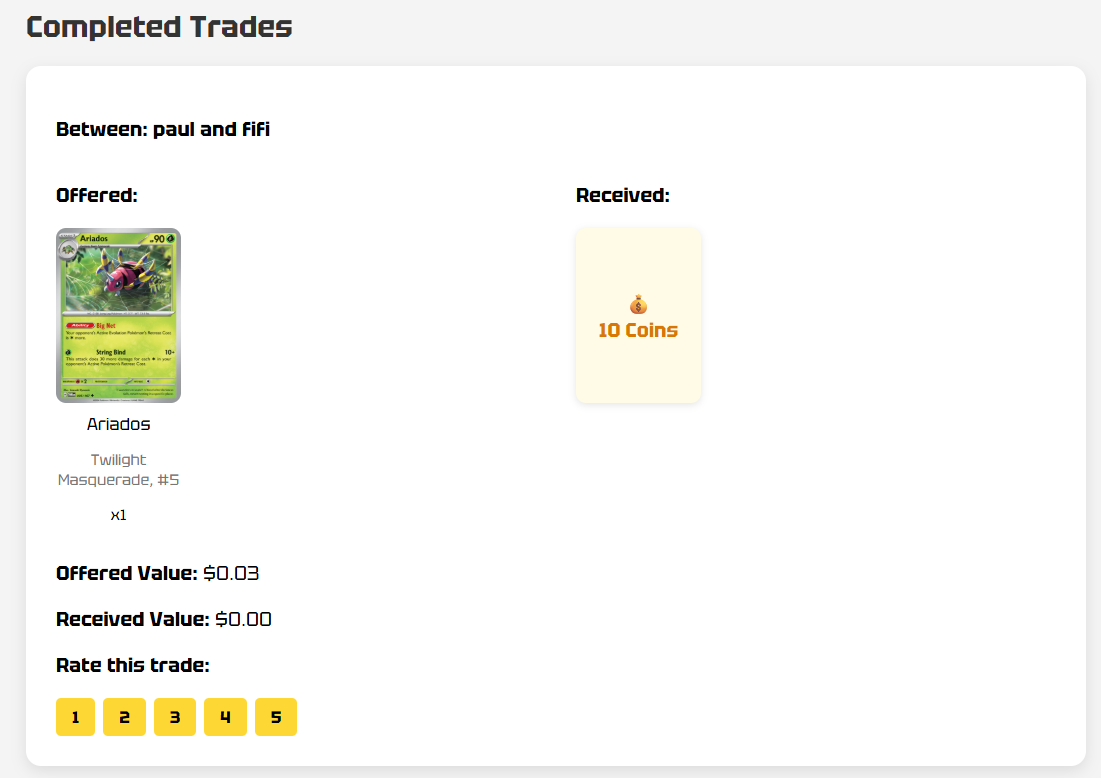
****

**Screenshot din minigame-ul „Guess the Pokémon”**

**6.2.7 Vizulizarea schimburilor anterioare și sistemul de rating**

Pe pagina Profile, utilizatorii pot vedea câți coins au și ratingul acumulat de-a lungul timpului din trade-urile în care au fost implicați. De asemenea, pot vizualiza trade-urile anterioare pe care le-au completat și pot lăsa, la rândul lor, un rating pentru a contribui la creșterea calității schimburilor viitoare dintre utilizatori.





**Screenshot-uri de pe pagina Profile**

**6.3 Implementarea funcționalităților principale**

În acest subcapitol voi explica modul în care au fost implementate câteva dintre funcționalitățile principale ale aplicației PokeTrader.

**6.3.1 Guess the Pokémon**

Când utilizatorul apasă pe butonul „Start Game”, este apelat endpoint-ul StartGuessGameView. Acest endpoint verifică dacă utilizatorul a jucat deja jocul în ziua curentă sau dacă există deja un joc activ. Dacă nu, este selectat un Pokémon aleatoriu din baza de date și este creată o nouă instanță de joc. Ulterior, în modelul UserProfile se salvează data ultimei participări la joc a utilizatorului curent.

Utilizatorul poate pune întrebări despre caracteristicile Pokémonului, iar acestea sunt trimise către endpoint-ul AskQuestionView. Acesta preia jocul activ și Pokémonul țintă, analizează întrebarea folosind expresii regulate și cuvinte cheie, și răspunde cu „yes” sau „no”, în funcție de informațiile disponibile despre Pokémon. Întrebarea și răspunsul sunt apoi salvate în modelul GuessMessage, pentru a fi vizibile și după un refresh. În cazul în care întrebarea nu este recunoscută, utilizatorul primește mesajul: „I don't understand the question.”

Utilizatorul poate oricând să încerce să ghicească numele Pokémonului. Endpoint-ul GuessPokemonView verifică dacă ghicirea este corectă. Dacă da, jocul este șters din baza de date, iar utilizatorul primește 10 coins. Dacă nu, presupunerea este adăugată în conversație, iar jocul poate continua.

Toate întrebările, răspunsurile și presupunerile sunt stocate în modelul GuessMessage, care este legat de jocul activ. Acestea sunt afișate în interfață sub formă de chat, oferind utilizatorului context pentru a deduce corect Pokémonul.

**6.3.2 Sistemul de Trade**

Funcționalitatea principală a aplicației PokeTrader constă în sistemul de schimb de cărți între utilizatori. Procesul începe prin completarea unui formular accesibil din pagina Collection, unde utilizatorul autentificat selectează cărțile sau monedele pe care dorește să le ofere, precum și ceea ce își dorește să primească în schimb. Utilizatorul poate selecta doar cărțile marcate ca „Available” în colecție, adică cele care nu sunt deja implicate în alte schimburi.

După completarea formularului, datele sunt trimise către backend, unde endpointul TradeListCreateView le primește și verifică dacă utilizatorul are resursele necesare pentru a posta trade-ul. Dacă totul este în regulă, în baza de date se salvează un obiect Trade cu informațiile corespunzătoare.

Când un alt utilizator dorește să accepte un trade, acesta apasă pe butonul „Accept Trade”, care trimite un request către endpointul AcceptTradeView. Aici se verifică dacă utilizatorul deține toate cărțile și monedele necesare pentru a finaliza schimbul. Dacă toate condițiile sunt îndeplinite, trade-ul este marcat ca inactiv, este salvat utilizatorul care l-a acceptat și se trece în etapa de confirmare.

Pentru a finaliza schimbul, ambii utilizatori trebuie să-l confirme. Acest lucru se realizează în pagina TradeChat, printr-un request către endpointul ConfirmTradeView. După ce ambele părți confirmă schimbul, cantitățile de cărți și monede sunt actualizate: acestea se scad din colecția utilizatorului care le oferă și se adaugă în colecția celuilalt. Informațiile despre schimb sunt salvate într-un model CompletedTrade, pentru a putea fi afișate ulterior pe profilurile utilizatorilor, iar trade-ul inițial este șters din baza de date.

Am implementat acest sistem pentru a preveni utilizarea acelorași cărți în mai multe schimburi simultan, folosind blocarea temporară a cărților implicate până la confirmarea finală a trade-ului.

**6.3.3 Sistemul de chat dintre utilizatori**

După ce un schimb este acceptat de către un alt utilizator, aplicația oferă un spațiu dedicat comunicării dintre cei doi utilizatori, pentru a stabili detalii legate de desfășurarea trade-ului. Acest lucru se realizează pe pagina TradeChat, care implementează un sistem simplu de mesagerie.

Când un utilizator accesează chatul asociat unui trade, aplicația face două requesturi către backend: unul pentru a prelua mesajele anterioare asociate acelui trade, iar altul pentru a verifica dacă utilizatorul curent a confirmat deja trade-ul. Fiecare mesaj este salvat în modelul Message, care reține textul, utilizatorul care a trimis mesajul, trade-ul asociat chatului și momentul în care mesajul a fost trimis.

Pe lângă trimiterea de mesaje, această pagină permite și confirmarea trade-ului. Butonul "Confirm Trade" trimite un request care marchează utilizatorul respectiv ca fiind de acord cu schimbul. După ce ambii utilizatori confirmă, cantitățile de cărți sunt actualizate în baza de date, trade-ul este finalizat și este salvat în modelul CompletedTrade.

**6.3.4 Sistemul de rating**

După finalizarea unui schimb și confirmarea acestuia de către ambii utilizatori, aplicația permite evaluarea celuilalt user printr-un sistem de rating simplu, de la 1 la 5 stele. Această funcționalitate este implementată în pagina Profile, unde utilizatorii pot vizualiza lista schimburilor anterioare, care au fost completate și da o notă de la 1 la 5 celuilalt utilizator care a fost implicat in trade. Evaluarea se realizează printr-un simplu click pe una dintre stele, acțiune care trimite un request către backend, salvând scorul în baza de date, asociindu-l atât cu utilizatorul care a lasat rating-ul, cât și cu trade-ul evaluat.

Rating-urile primite sunt apoi folosite pentru a calcula media fiecărui utilizator, medie care este afișată pe pagina Profile. Aceasta are scopul de a reflecta comportamentul utilizatorului în cadrul schimburilor și de a încuraja utilizatorii să fie serioși în legătură cu schimburile pe care le propun.

**6.3.5 Login, Register și ProtectedRoute**

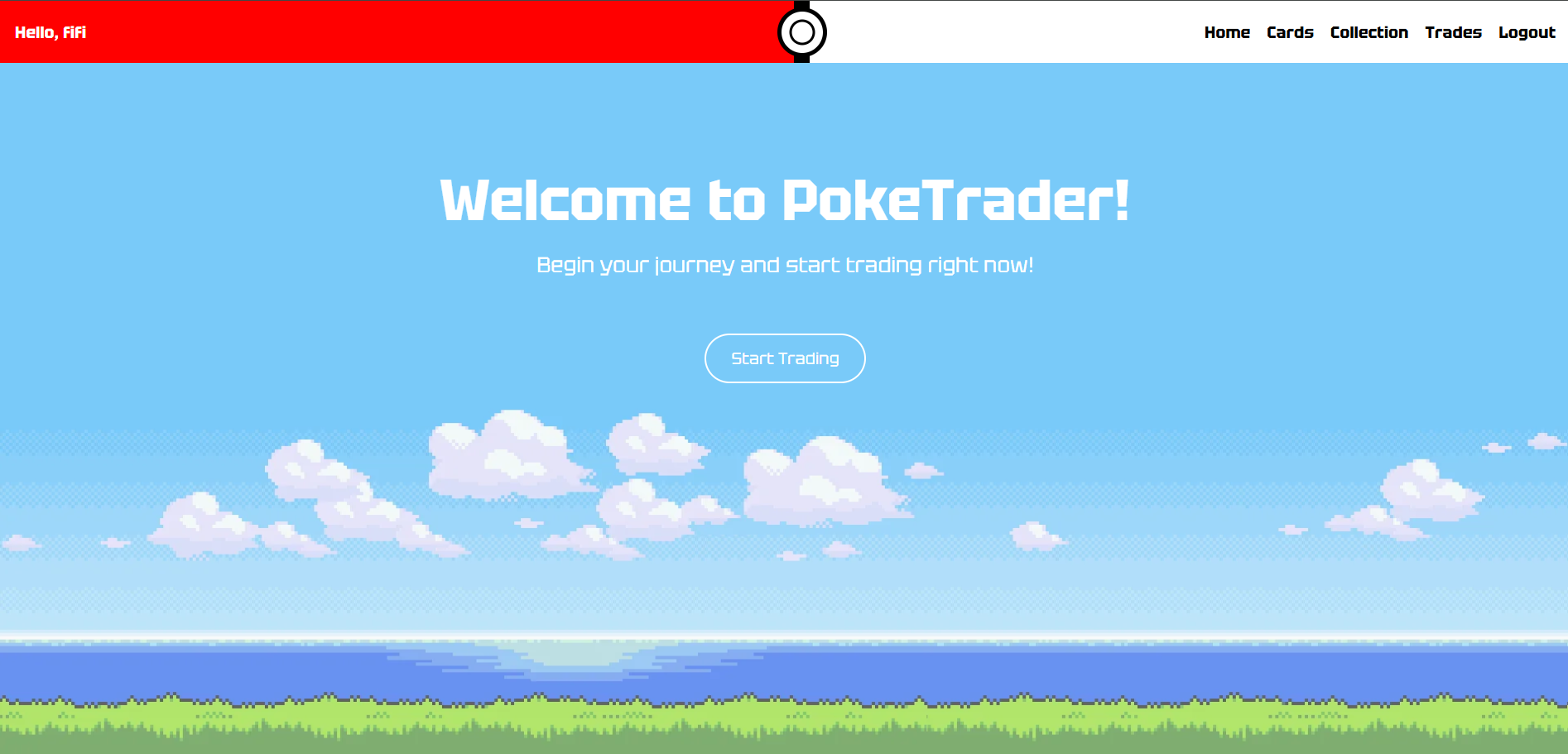
Când un utilizator accesează pagina de înregistrare, este afișat conținutul componentei RegisterForm, care conține un formular pentru introducerea unui username, a unei adrese de email și a unei parole. După completarea formularului, datele sunt trimise către backend printr-un request de tip POST, iar în backend este creat un nou cont de utilizator. Dacă înregistrarea are succes, utilizatorul este redirecționat automat către pagina de Login.

Pe pagina de Login este afișată componenta LoginForm. Aici, utilizatorul introduce adresa de email și parola, iar aplicația trimite un request către backend, unde aceste date sunt verificate. Dacă autentificarea reușește, serverul trimite un token JWT, care este salvat în localStorage. Acest token este utilizat în requesturile ulterioare pentru a demonstra că utilizatorul este autentificat.

Pentru a preveni accesul neautorizat la anumite pagini, aplicația utilizează componenta ProtectedRoute, care verifică dacă tokenul JWT este prezent în localStorage. Dacă tokenul există, utilizatorul poate accesa pagina dorită; în caz contrar, este redirecționat către pagina de login. Această abordare asigură că doar utilizatorii autentificați pot accesa funcționalitățile principale ale aplicației, cum ar fi vizualizarea colecției, propunerea de schimburi și altele.

**6.4 Fluxul utilizatorului în aplicație**

Fluxul tipic al unui utilizator în aplicația PokeTrader începe cu procesul de înregistrare. Utilizatorul își creează un cont prin completarea unui formular cu username, email și parolă, după care se autentifică folosind adresa de email și parola. Odată logat, acesta este redirecționat către pagina Home.



**Screenshot de pe pagina Home a aplicației**

De acolo, poate accesa pagina Sets, unde are la dispoziție toate seturile disponibile. După selectarea unui set, utilizatorul poate adăuga cărțile dorite în colecția sa. Cărțile adăugate devin vizibile în pagina Collection, unde sunt afișate informații precum cantitatea deținută, market price-ul și statusul (available sau up for trade).

Pentru a iniția un schimb, utilizatorul accesează formularul de trade, selectează cărțile pe care dorește să le ofere și ce își dorește în schimb, apoi postează trade-ul. Alți utilizatori pot vizualiza oferta pe pagina Trades și o pot accepta, cu condiția să dețină cărțile sau monedele necesare. După acceptarea trade-ului, cei doi utilizatori pot discuta în pagina TradeChat, unde stabilesc detalii legate de desfășurarea schimbului. Ambele părți trebuie să confirme manual finalizarea, moment în care cantitățile de cărți sunt actualizate în baza de date, iar schimbul este marcat ca finalizat.

Pe lângă funcționalitatea de trading, utilizatorul poate accesa minigame-ul Guess the Pokémon, unde poate câștiga coins zilnic. Monedele acumulate pot fi folosite ulterior în cadrul schimburilor.

În final, utilizatorul poate vizualiza pe pagina Profile toate trade-urile finalizate, numărul de coins disponibili și ratingul primit din partea celorlalți utilizatori.

**7. Concluzii și direcții de dezvoltare viitoare**

**7.1 Direcții de dezvoltare viitoare**

Deși aplicația PokeTrader este complet funcțională în forma actuală, există mai multe moduri prin care aceasta poate fi extinsă și îmbunătățită în viitor. O primă idee ar fi dezvoltarea unei aplicații mobile, pentru a permite utilizatorilor să acceseze platforma mai ușor, direct de pe telefon. Acest lucru ar aduce un plus de confort și ar crește gradul de utilizare a aplicației.

În ceea ce privește minigame-ul Guess the Pokémon, acesta ar putea fi îmbunătățit prin integrarea unui model de inteligență artificială mai avansat, care să permită utilizatorilor să formuleze întrebări mai naturale, fără a fi constrânși la anumite expresii cheie. Astfel, interacțiunea ar deveni mai fluentă și mai interesantă.

O altă funcționalitate utilă ar fi introducerea de notificări în timp real, de exemplu atunci când un trade este acceptat sau un mesaj nou a fost trimis într-un chat, iar pentru a îmbunătăți comunicarea în chat, ar fi benefică trecerea de la sistemul actual de mesagerie bazat pe requesturi la o arhitectură cu WebSockets, care să permită transmiterea mesajelor în timp real, fără refresh.

În ceea ce privește infrastructura, aplicația ar putea fi migrată în viitor către o bază de date mai performantă, precum PostgreSQL, pentru a susține un număr mai mare de utilizatori și date fără a afecta performanța.

De asemenea, funcționalitatea de explorare a schimburilor poate fi îmbunătățită prin adăugarea unor filtre avansate, care să permită sortarea și căutarea trade-urilor în funcție de niște criterii care ar ajuta utilizatorul să găsească mai rapid ce îl interesează.

**7.2 Concluzii finale**

Lucrarea de față a avut ca scop dezvoltarea unei aplicații web care să faciliteze schimburile de cărți Pokémon între colecționari. Pornind de la o nevoie reală identificată în comunitate (lipsa unei platforme dedicate schimburilor), aplicația propusă oferă o alternativă modernă, sigură și prietenoasă față de metodele tradiționale, cum ar fi grupurile de Facebook.

Prin funcționalitățile implementate aplicația reușește să acopere principalele nevoi ale unui colecționar care dorește să interacționeze eficient cu alți utilizatori. De asemenea, dezvoltarea aplicației a reprezentat o oportunitate de învățare, oferindu-mi ocazia să mă familiarizez cu tehnologii moderne precum Django, React, REST API, JWT sau integrarea de API-uri externe.

**8. Bibliografie**