Blackjack

**Prezentare generală**

1. **Front-end**

**First Page**

**O imagine care conține captură de ecran, text, Software multimedia, Software de grafică

Descriere generată automat**

**Game Page**

Pe această pagină, jucătorul poate plasa o miză. (Suma trebuie să fie între 10 și soldul disponibil al jucătorului.)**O imagine care conține captură de ecran, Dreptunghi, verde, proiectare

Descriere generată automat**

După ce introduce o sumă în câmpul de introducere și apasă pe „Bet”, jocul începe. Jucătorul are acum opțiunea de a cere o carte suplimentară (hit) sau de a rămâne cu cărțile pe care le are (stand) apăsând butoanele din partea dreaptă.

El are deja o sumă bună de puncte (20), așa că alege să rămână (stand).

**O imagine care conține captură de ecran, Dreptunghi

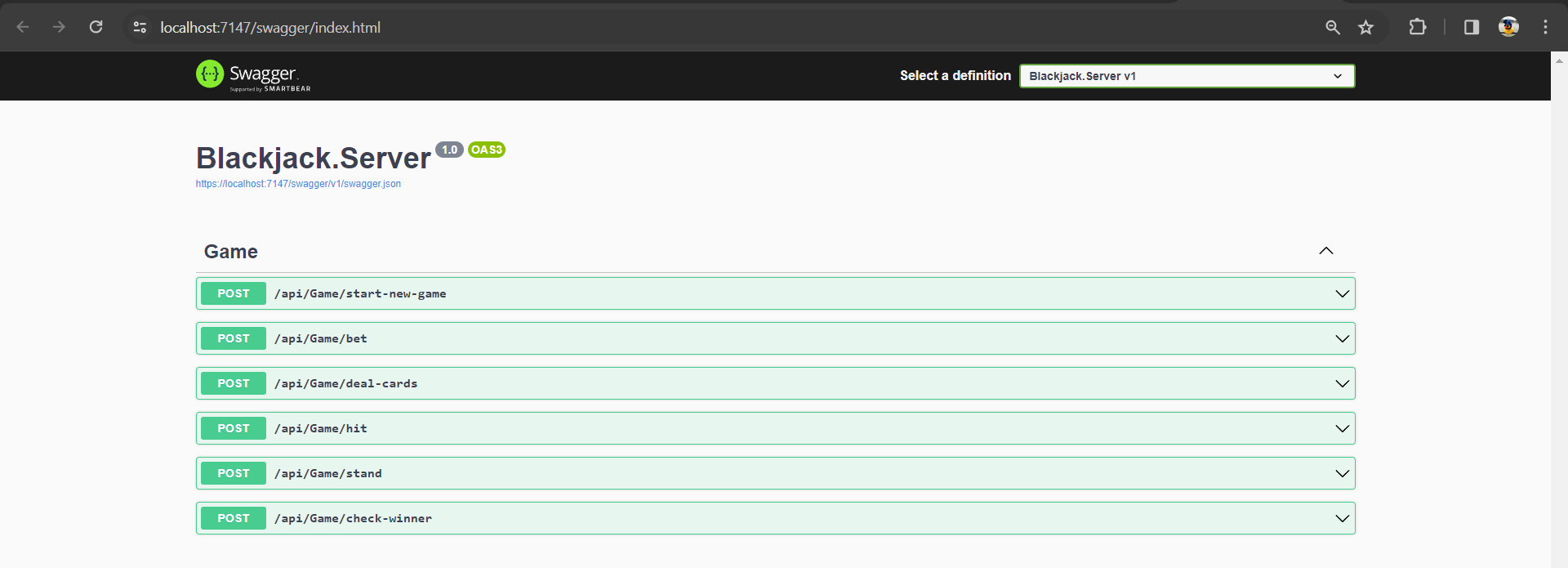
Descriere generată automat**

Jucatorul caștiga. 20 > 19

**O imagine care conține captură de ecran

Descriere generată automat**

1. **Back-end**



Backend-ul, realizat cu ASP.NET Core și .NET 8.0, constituie nucleul logic al aplicației și furnizează mai multe endpoint-uri esențiale pentru crearea și desfășurarea unui joc de Blackjack:

O imagine care conține text, captură de ecran, software, Software multimedia

Descriere generată automat

**start-new-game** - Acest endpoint inițiază un joc nou de blackjack. Este responsabil pentru începerea unei noi runde, resetarea stării jocului și pregătirea pentru o nouă rundă de pariere.

**Raspuns**: Stadiul jocului. (mana jucatorului, a dealer-ului si datele jucatorului)

**bet** - Aici se plasează pariurile pentru runda curentă de blackjack. Jucătorul poate specifica suma pe care dorește să o parieze pentru a participa la mâna respectivă.

**Raspuns**: Mesaj text care precizeaza dacă pariul a fost plasat cu succes.

**deal-cards** - Acest endpoint împarte cărțile către jucător și dealer în cadrul unei runde. Jucătorul primește inițial două cărți, iar dealerul primește o carte vizibilă și una ascunsă.

**Raspuns**: Stadiul jocului. (mana jucatorului, a dealer-ului si datele jucatorului)

**hit** - Atunci când jucătorul dorește să primească o carte suplimentară din pachet, apelează acest endpoint. Aceasta îi oferă o nouă carte jucătorului în încercarea de a se apropia cât mai mult de valoarea 21 fără a o depăși.

**Raspuns**: Stadiul jocului. (mana jucatorului, a dealer-ului si datele jucatorului)

**stand** - Jucătorul folosește acest endpoint pentru a rămâne cu cărțile pe care le are în mână, indicând că nu mai dorește cărți suplimentare. Acesta este pasul în care jucătorul decide să rămână cu mâna sa actuală și să lase dealerul să-și continue jocul.

**Raspuns**: Stadiul jocului. (mana jucatorului, a dealer-ului si datele jucatorului)

O imagine care conține text, software, Font, număr

Descriere generată automat

**check-winner** - După ce jucatorul și dealerul și-au ales acțiunile finale (hit sau stand), acest endpoint determină câștigătorul rundei. Verifică mâinile jucătorului și ale dealerului pentru a decide cine câștigă mâna conform regulilor blackjack-ului și distribuie câștigurile sau pierderile corespunzătoare.

**Raspuns**: Mesaj text. (“Player wins”, “Dealer wins” sau “Tie”)

**Structura fișierelor**

**O imagine care conține text, captură de ecran, software

Descriere generată automat** **O imagine care conține text, captură de ecran, Font

Descriere generată automat**

**Blackjack.Server** reprezinta logica aplicației și are o structură bine definită:

**Controllers** - reprezintă modulul care definește și gestionează endpoint-urile utilizate în interacțiunea cu aplicația.

**Models** - cuprinde clasele, enum-urile și interfețele folosite în logica jocului, precum și stările specifice ale jocului (faze precum Betting, Player turn, Dealer turn, Game over). Aici sunt și fabricile de cărți și pachete de cărți, detalii despre acestea puteți găsi în secțiunea de Design Patterns.

**Services** - acționează ca puntea de legătură între logica jocului și endpoint-urile definite în Controllers, facilitând interacțiunea dintre aceste componente.

Directorul **Blackjack.Tests** conține seturile de fișiere de test pentru modelele și alte componente din serverul de Blackjack. Aici sunt incluse testele unitare și de integrare pentru a evalua corectitudinea și funcționalitatea modelelor din aplicație.

Aceste teste sunt esențiale pentru asigurarea calității codului și pentru identificarea eventualelor probleme sau erori în implementare. Ele sunt proiectate pentru a verifica comportamentul și funcționarea corectă a claselor, enum-urilor și interfețelor din cadrul logicii jocului de Blackjack.

O imagine care conține text, captură de ecran, Font

Descriere generată automat

În plus față de fișierele de test, directorul poate conține și un folder separat numit TestResults, care este folosit pentru a stoca rezultatele testelor rulate în timpul procesului de testare. Acest folder poate conține rapoarte, fișiere de tip log sau alte informații relevante generate în urma rulării și evaluării testelor pentru a facilita analiza și remedierea eventualelor probleme identificate.

**Teste:**

**CardTests** – Verifica compotamentul si funcționalitatea corectă a clasei ‘Card’ din perspectiva valorii cărților.

**DealerHandTests** – Testele asociate clasei ‘DealerHand’ evalueaza modul în care dealer-ul gestionează și își formează mâna în timpul jocului.

**DealingPackTests** – Testele pentru clasa ‘Deck’ evaluează logica de creare, gestionare și utilizare a unui întreg set de pachete de cărți în cadrul jocului de Blackjack.

**DeckTests** – Acest set de teste verifică funcționalitatea corectă a clasei ‘Deck’, care include manipularea pachetului de cărți și amestecarea lor.

**PlayerHandTests** – Verifică comportamentul clasei ‘PlayerHand’, focalizându-se pe modul în care mâinile jucătorilor sunt gestionate, inclusiv adăugarea, evaluarea valorii mâinii și alte acțiuni relevante pentru mâna jucătorului.

**PlayerTests** - Testele asociate clasei ‘Player’ evalueaza setarea corectă a atributelor acestuia.

O imagine care conține text, captură de ecran, software, Software multimedia

Descriere generată automat

**Design Pattern**

1. **Singleton cu Double Locking** pentru clasa BlackjackGame: Ați aplicat pattern-ul Singleton pentru clasa BlackjackGame, asigurând că există o singură instanță a acestei clase și oferind acces global la această instanță. Implementarea Double Locking asigură eficiența și siguranța în accesul la instanță în timpul executării aplicației.

**O imagine care conține text, captură de ecran, Font

Descriere generată automat**

1. **Factory Design Pattern** pentru crearea cărților și a pachetelor de cărți din joc: Utilizarea pattern-ului Factory a permis crearea și gestionarea eficientă a obiectelor Card și Deck (pachet de cărți). Factory pattern-ul oferă o metodă centralizată pentru crearea acestor obiecte, abstractizând procesul de creare și permitând extensibilitatea în adăugarea altor tipuri de cărți sau pachete de cărți în viitor.

**O imagine care conține text, captură de ecran, Font

Descriere generată automat**

1. **State Design Pattern** pentru tranziția între stările jocului: Implementarea State Design Pattern pentru stările jocului (BettingPhase, PlayerTurn, DealerTurn, GameOver) oferă o structură în care obiectele pot schimba comportamentul lor în funcție de starea în care se află. Acest pattern permite o gestionare flexibilă a tranzitiei între diversele stări ale jocului, permițând jocului să reacționeze și să se comporte diferit în funcție de starea actuală.

O imagine care conține text, captură de ecran, Font, proiectare

Descriere generată automat

**Diagrame**

1. **Diagrama de utilizare (Use Case Diagram)**

Această diagramă arată interacțiunile între utilizatori și sistemul jocului de blackjack. Include actorii implicați, cum ar fi jucătorii și dealerul, precum și cazurile de utilizare cum ar fi "Place bet", "Hit with a card", "Stay with current card" etc. Diagrama de utilizare evidențiază funcționalitățile oferite de sistem și interacțiunile acestuia cu utilizatorii săi.

O imagine care conține text, diagramă, linie, desen

Descriere generată automat

1. **Diagrama de clase (Class Diagram)**

Această diagramă arată structura statică a jocului de blackjack, evidențiind clasele utilizate, relațiile dintre acestea, atributele și metodele acestora. Clasele precum Player, Card, Deck, Game și altele sunt prezentate, împreună cu relațiile lor, oferind o imagine de ansamblu asupra structurii și relațiilor din cadrul aplicației.

O imagine care conține text, diagramă, captură de ecran, Plan

Descriere generată automat

1. **Diagrama de stare (State Chart Diagram)**

Diagrama de stare prezintă stările diferite ale jocului de blackjack și tranzitia între acestea. Pentru jocul de blackjack, stările cum ar fi "Împarte carti", "Randul dealer-ului", "Joc castigat" ar putea fi ilustrate, evidențiind tranzitia între aceste stări și condițiile care le guvernează.

O imagine care conține text, diagramă, captură de ecran, Paralel

Descriere generată automat

1. **Diagrama de activitate (Activity Diagram)**

Această diagramă oferă o vedere de ansamblu asupra fluxului de activități din cadrul jocului de blackjack. Ilustrează procesele specifice, cum ar fi fluxul de acțiuni în timpul unei runde de joc, cum începe și se desfășoară jocul, precum și deciziile luate de jucători și dealer în cadrul jocului.

O imagine care conține diagramă, text, linie, Paralel

Descriere generată automat

1. **Diagrama de secvență (Sequence Diagram)**

Această diagramă oferă o reprezentare a interacțiunilor între obiecte în cadrul jocului de blackjack într-un anumit scenariu sau flux de acțiuni. Ilustrează ordinea și interacțiunile dintre obiecte precum jucătorul, dealerul și alte entități, evidențiind mesajele trimise și primite între acestea.

O imagine care conține text, diagramă, Paralel, linie

Descriere generată automat

1. **Diagrama de colaborare (Collaboration Diagram)**

Acest tip de diagramă oferă o reprezentare grafică a interacțiunilor între obiecte sau clase în cadrul jocului de blackjack. Arată relațiile și schimbul de mesaje între diferitele obiecte sau entități într-un mod vizual și clar, evidențiind colaborarea și comunicarea dintre ele în cadrul aplicației.

O imagine care conține text, captură de ecran, diagramă, Paralel

Descriere generată automat