# BlackJack

**15.01.2025**

**Îndrumător:**

**dr. ing. Daniel Morariu**

**Student:**

**David Raul Denis**

**C 23/2**

# Istoric Versiuni

| Data | Versiune | Descriere | Autor |
|---|---|---|---|
| <dd/mmm/yy> | <x.x> | <detalii> | <nume> |
| | | | |
| | | | |
| | | | |

[În această secțiune puteți scrie informații despre când ați rezolvat o anumită problemă din proiect. Aveți astfel o evoluție a implementării proiectului realizat. Puteți scrie în coloana de autor și cine v-a ajutat în cazul în care nu ați rezolvat o anumită problemă singuri. Pot să existe în proiect anumite părți mici făcute de alt cineva în cazul în care nu ați înțeles momentan cum se rezolvă problema respectivă și ați cerut ajutor de la un coleg/ prieten.

Atenție în continuare și la aranjarea documentului în pagină, se va nota și acest lucru.

Acest document trebuie să îl faceți astfel încât să vă ajute la susținerea proiectului. Deci orice informație pe care o considerați utilă pentru a explica cum ați gândit proiectul puteți să o includeți aici. Este singurul loc în care în timpul p redării puteți căuta.

Textele scrise cu albastru în document conțin doar informații despre ce trebuie scris la secțiunea respectivă, ulterior trebuie sa fie șterse din document. La final în document nu vor fi texte scrise cu albastru.

Pentru redactarea proiectului recomand fontul Times New Roman 12pt, aliniere justified, spatiere la 1,5pt și cu 6pt spatiu între paragrafe, fără identare pe prima linie din paragraf]

Acesta este un text pentru a exemplifica modul de scriere recomandat în document. Font Times New Roman de 12pt, aliniere justified, spațiere 1.5pt, cu spațiere de 6pt între paragrafe identice și fără identare pe prima linie din paragraf. Se pot folosi stilurile deja pregătite in document (Normal, Heading 1, Heading 2, Title,...(nu și stilul comentariu folosit de mine pentru a explica informațiile care trebuie scrise în acea secțiune).

# Cuprins

# 1 Specificarea cerințelor software

## *1.1 Introducere*

Proiectul presupune dezvoltarea unui joc de Blackjack utilizând Embarcadero C++ Builder. Aplicația oferă utilizatorilor posibilitatea de a juca Blackjack într-un mediu grafic, integrând funcționalități precum distribuirea cărților, calculul scorurilor și respectarea regulilor jocului. Obiectivul principal al proiectului este de a pune în practică cunoștințele de programare orientată pe obiecte.

### 1.1.1 Obiective

- Crearea unui joc simplu de BlackJack
- Crearea unei interfețe pentru jocul de BlackJack
- Implementarea unui sistem de reguli
- Implementarea opțiunilor de joc prin butoanele „Hit" și „Stand"
- Implementarea unui sistem de afișare a cărților în funcție de butonul apăsat
- Determinarea câștigătorului

### 1.1.2 Definiții, Acronime și Abrevieri

**Definiții și termeni:**

- **BlackJack –** Un joc de cărți al cărui scop este de a obține un scor de 21 sau cât mai aproape de această valoare, scorul jucătorului trebuind să fie mai mare decât scorul dealer-ului. De asemenea, dacă scorul cuiva depășește 21, acesta pierde jocul.
- **Player –** Cel care utilizează aplicația. Acesta are de ales între butonul „Hit" și cel de „Stand".
- **Dealer** – Oponentul Player-ului, reprezentat de calculator. Acesta joacă după reguli prestabilite.
- **Deck –** Un pachet de 52 de cărți de joc.
- **Card –** Reprezintă o carte din Deck. Aceasta are două atribute: **Symbol** și **Suit**.

**Acronime și abrevieri:**

- **P –** Se referă la toate elementele care au legătură cu Player-ul.
- **D –** Se referă la toate elementele care au legătură cu Dealer-ul.

- **Nr** – Este o prescurtare a cuvântului „număr”.
- **Panel** – Element specific componentelor de tip “Panel”.

### 1.1.3 Tehnologiile utilizate

- **Embarcadero C++ Builder** – Mediul în care au fost create aplicația și interfața grafică.
- **C++** – Limbajul de programare utilizat.
- **Visual Studio** – Mediul în care au fost create inițial clasele.
- **ChatGPT** – Folosit pentru crearea unui logo în interiorul interfeței grafice.

## *1.2 Cerințe specifice*

[Această secțiune ar trebui să conțină doar obiectivele propuse în secțiunea 1.1.1 și care au fost și realizate efectiv. Aceste obiective o sa le numim în continuare funcționalități ale aplicației realizate. În această secțiune se vor enumera toate funcționalitățile implementate ale aplicației și eventual o scurtă descriere a ceea ce face fiecare funcționalitate în parte dacă nu se înțelege din numele funcționalității.

Urmează ca în secțiunea următoare (secțiunea 2 ) să se prezinte detaliat cel puțin două dintre aceste funcționalități (se va realiza cate un capitol 2 separat pentru fiecare funcționalitate în parte). Se vor prezentat funcționalitățile care sunt considerate ca fiind cele mai relevante pentru aplicația realizată].
Exemplu de funcționalități:

- Desenare tablă joc
- afișarea pieselor pe tabla de joc
- mutarea pieselor pe tablă
- rotirea tablei de joc când este afișată la un jucător sau la alt jucător (în cazul în care este nevoie)
- codificarea realizată pentru mesajele transmise / recepționate pe rețea
- etc.

# 2 Nume Funcționalitate 1 și 2

## *2.1 Descriere*

[Această secțiune cuprinde o scurtă descriere a funcționalității care va fi prezentată. Tot aici se va specifica de ce se consideră că funcționalitatea aleasă pentru a fi prezentată este relevantă în aplicația curentă.]

## *2.2 Fluxul de evenimente*

### 2.2.1 Fluxul de bază (dați un nume fluxului de bază)

[În această secțiune se va detalia modul de realizare a funcționalității prezentate în această secțiune. Se vor descrie în pași detaliați modul de implementare a funcționalității respective. Această descriere trebuie să conțină tot ce trebuie să facă utilizatorul pentru a pornii această funcționalitate, ce face sistemul ca răspuns la acea funcționalitate și ce ar trebui să obțină utilizatorul la ieșire.

Dacă se consideră necesar în această secțiune se pot prezenta și porțiuni de cod care se consideră de autor a fi relevant și care realizează o parte din funcționalitate. Fiecare cod trebuie însoțit de comentarii.

Dacă funcționalitatea descrisă a fost apelată din mai multe locuri se va detalia toate locurile din care a fost apelată și modul în care a fost apelată.

Dacă aceeași funcționalitate a fost implementată în mai multe locuri în aplicație iar în fiecare loc a fost implementată altfel atunci se va descrie în secțiunea Fluxuri alternative fiecare mod de implementare în parte. Dacă funcționalitatea a fost implementată doar într-un singur loc se va renunța la secțiunea 2.2.2. ].

[De exemplu funcționalitatea de parsare a unui sir de caracter la un moment dat s-a făcut extrăgând și analizând fiecare caracter în parte iar ulterior ați găsit o variantă mai simplă pentru parsarea șirului de caractere. Nu trebuie ștearsă prima implementare ci oferită și a doua implementare și în cod eventual apelate ambele variante (chiar dacă fac același lucru), atunci se vor explica separat ambele implementări.]

### 2.2.2 Fluxuri alternative

#### 2.2.2.1 < Primul flux alternativ >

[Se va detalia implementarea făcută în cazul în care funcționalitatea curentă a fost rezolvată și altfel decât rezolvarea de la fluxul de bază, pe motiv că ulterior s-au găsit și alte moduri mai optime de a implementa un anumit lucru. Fiecare mod de implementare va ocupa o secțiune separată. ]

#### 2.2.2.2 < Al doilea flux alternativ >

[Un alt mod folosit pentru a implementa funcționalitatea curentă.]

### 2.2.3 Pre-condiții

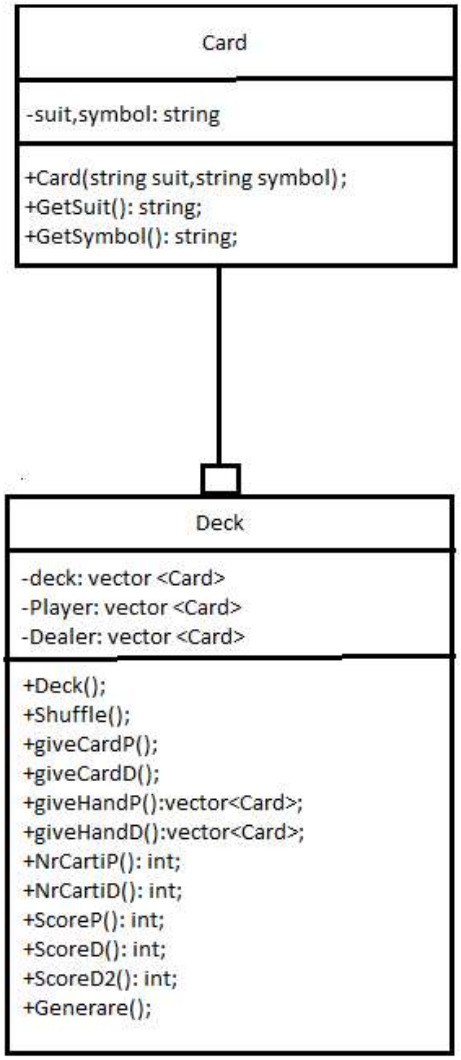
[Această secțiune conține o listă cu toate acțiunile care trebuie realizate de către utilizator pentru a porni în mod corect funcționalitatea curentă. Conține acțiunile și setările care trebuie realizate pentru o bună funcționare a aplicației (funcționalității descrise). Se vor detalia toate problemele care nu au fost momentan rezolvate și pot aduce aplicația într-o stare nefuncțională. Nu trebuie rezolvate toate problemele, este ok ca la predare sa mai fie cazuri netratate in cod, doar să știți de ele. Poate unele sunt momentan dificil de rezolvat. De exemplu pe ce butoane trebuie apăsat și în ce ordine, în cazul in care la început toate butoanele sunt active.]

### 2.2.4 Post-condiții

[Această secțiune conține o listă cu ceea ce obține utilizatorul după rularea funcționalității prezentate. La ce ar trebui să se aștepte utilizatorul după rularea funcționalității? Ce vede el pe ecran? Ce ar trebui ca să facă ca să vadă că funcționalitatea s-a terminat și că s-a terminat corect?]

# 3 Implementare

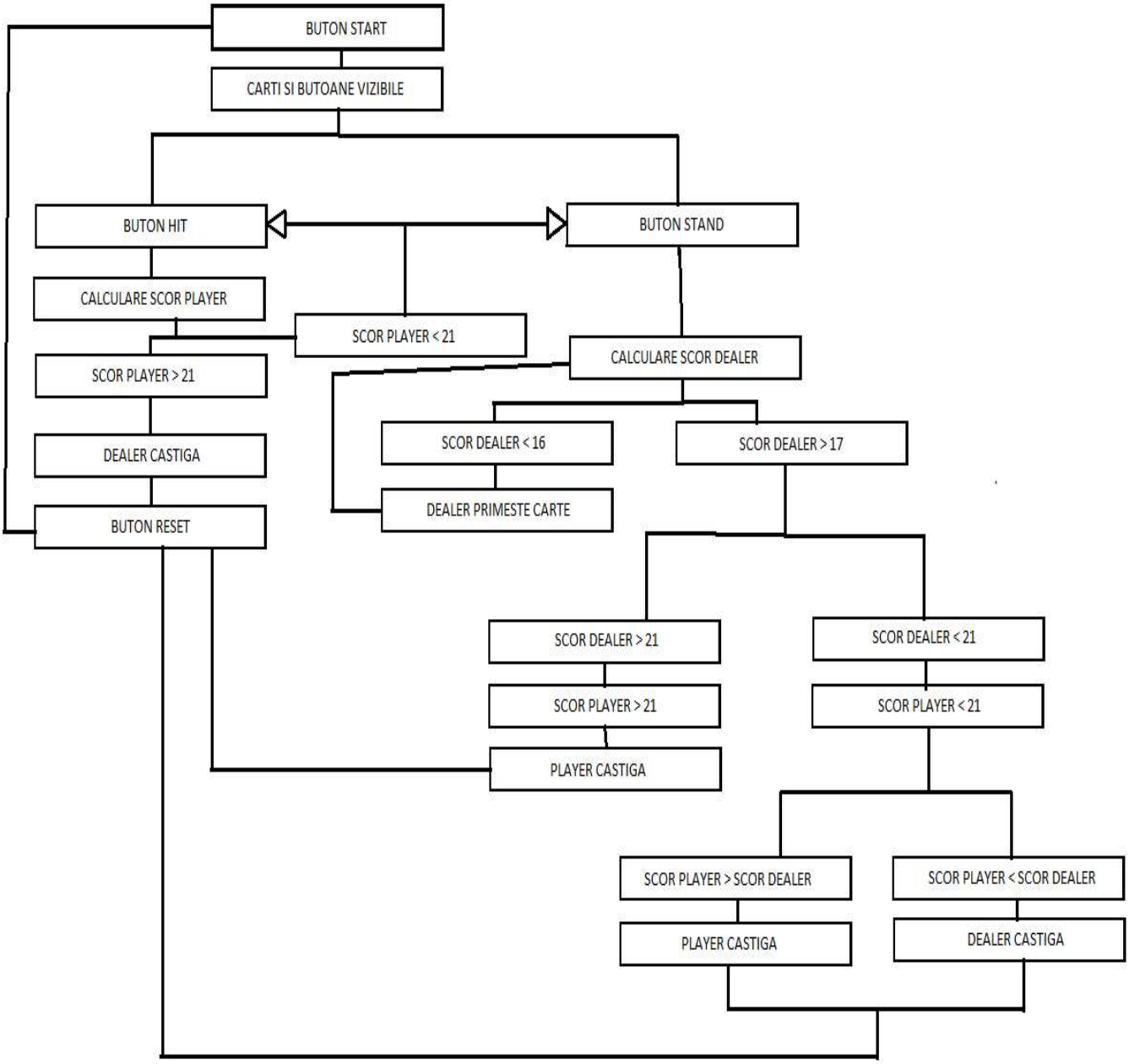
## *3.1 Diagrama de clase*




## *3.2 Descriere detaliată*

[Această secțiune va conține o diagramă bloc de funcționare a aplicației. Diagrama va conține toate blocurile funcționale considerate importante pentru aplicația curentă. Nu trebuie făcută diagrama detaliată a întregii aplicații ci, acum după implementare, cum ați considera că funcționează aplicația realizată]

Exemplu schemă logică:



# 4 Bibliografie


- The Washington Post - https://games.washingtonpost.com/games/blackjack
- Crazygames.com - https://www.crazygames.com/game/blackjack-master
- Replit.com - https://replit.com/@appbrewery/blackjack-start#main.py
- Cplusplus.com - https://cplusplus.com/forum/beginner/162557/
- Opengameart.org - https://opengameart.org/content/playing-cards-vector-png
- Opengameart.org - https://opengameart.org/content/colorful-poker-card-back
- StackOverflow.com - https://stackoverflow.com/questions/46752965/c-blackjack-stuck-trying-to-program-ace
- C++ Better Explained - https://www.youtube.com/watch?v=TUXanmeigqE&t=470s