# Design Overview for Card Game Program (Blackjack and Baccarat)

Name: Prachponleu Uch

Student ID: 103518056

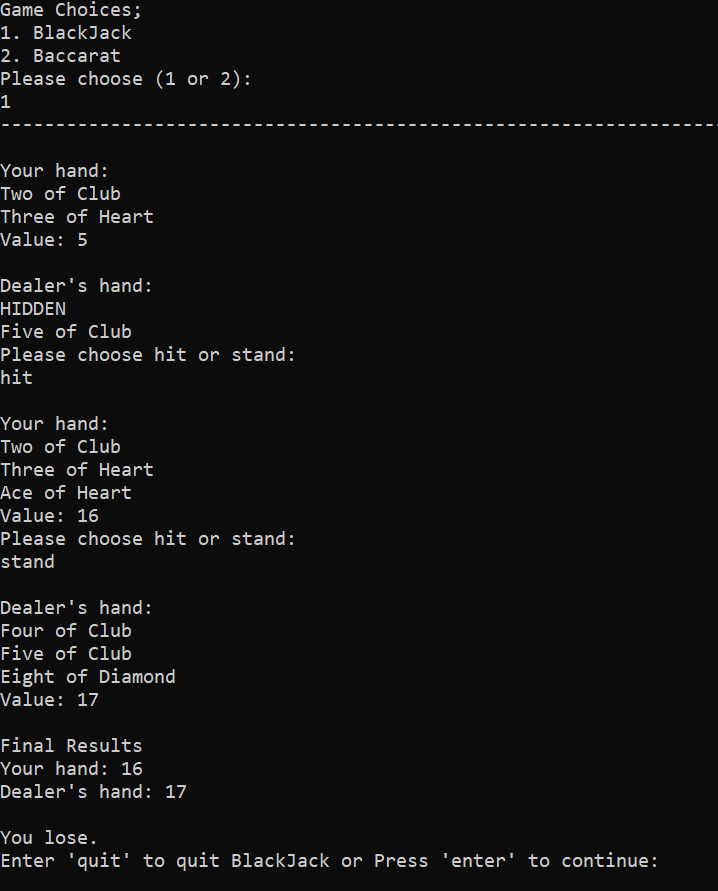
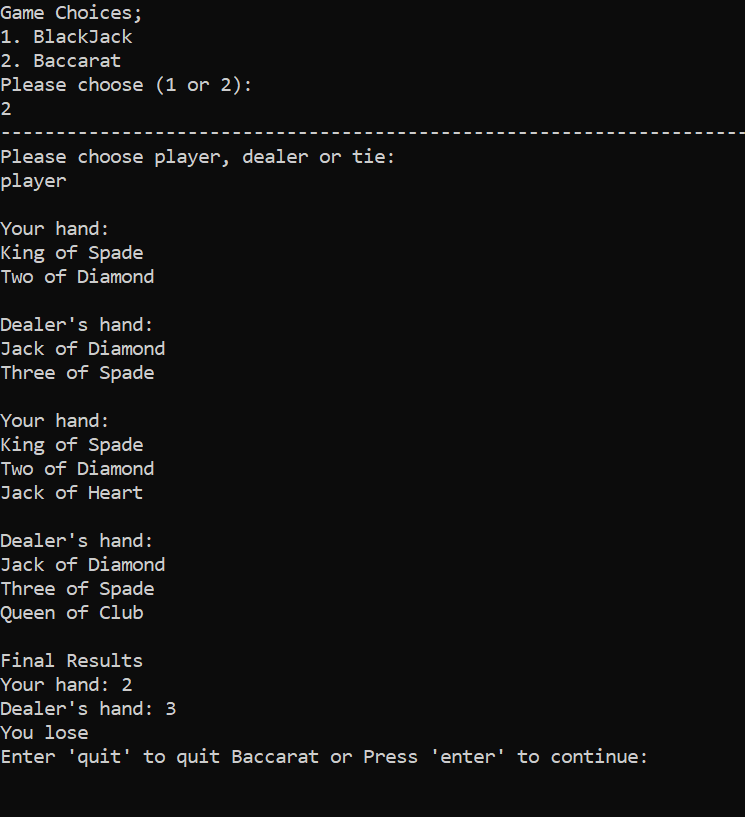
# Summary of Program

This program is a C# implementation of a blackjack and baccarat game, employing object-oriented programming (OOP) principles such as encapsulation, abstraction, polymorphism, and inheritance. Additionally, the program incorporates three design patterns: State, Factory, and Facade.

The State Design Pattern is utilized to enhance the program's scalability, debuggability, and code cleanliness. By avoiding the excessive use of if statements, the code becomes more readable and easier to maintain.

The Factory Design Pattern is employed to improve code readability by reducing the reliance on conditional statements like if and switch case. It also optimizes system resources by reusing existing objects instead of creating new ones for each occurrence.

The Facade Design Pattern enhances the user experience by performing calculations behind the scenes, minimizing the need for extensive user input within the program. This simplifies the user interaction and provides a smoother user experience.

# Required Roles

Describe each of the classes, interfaces, and any enumerations you will create. Use a different table to describe each role you will have, using the following table templates.

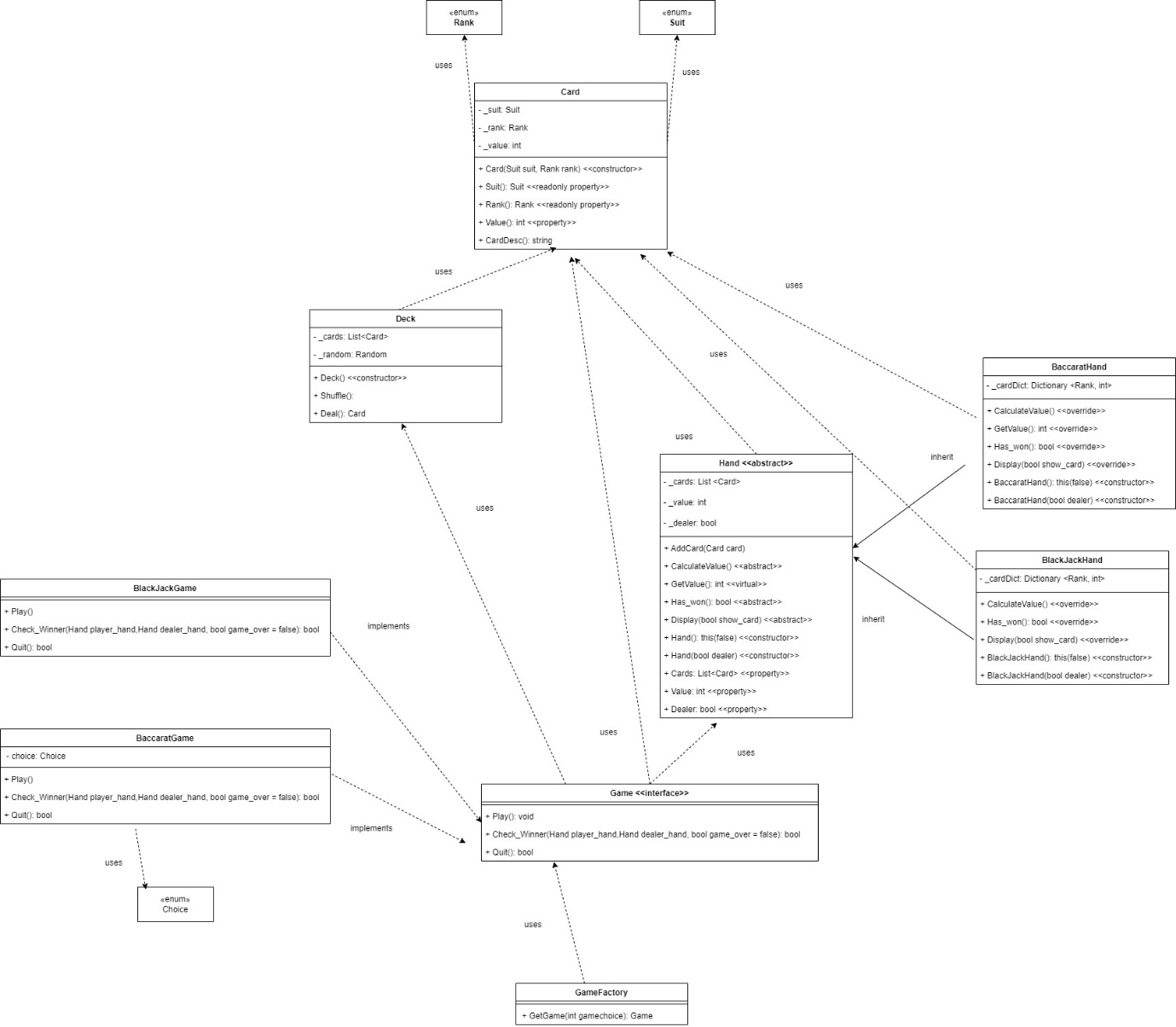
Table : <<role name>> details – duplicate

|  |  |  |
| --- | --- | --- |
| Responsibility | Type Details | Notes |
| Card Class is used to create card object, calculate its value, and get card description |  |  |
| Deck Class is used to create a deck of card, shuffle the deck, and deal the cards from the deck |  |  |
| Hand Class is used as an abstract parent class/blueprint for other states of Hand Class. |  |  |
| Baccarat Hand is child class of Hand used to create a hand object for Baccarat game that have functionalities of adding cards, calculating hand value, return value of the hand, check if the object has won, and displaying the hand or not according to the method parameter. |  |  |
| BlackJack Hand is child class of Hand used to create a hand object for BlackJack game that have functionalities of adding cards, calculating hand value, returning value of the hand, checking if the object has won, and displaying the hand or not according to the method parameter. |  |  |
| Game interface that acts as a blueprint for other games, in specific, card games. |  |  |
| Using Game interface blueprint to create Blackjack game class that can stimulate the game logic of blackjack through Play() method, check\_winner is for checking if any players/dealer has won, and Quit method for exiting the game |  |  |
| Using Game interface blueprint to create Baccarat game class that can stimulate the game logic of blackjack through Play() method, check\_winner is for checking if any player has won, and Quit method for exiting the game |  |  |
| GameFactory class is used to return a game state (BlackJack or Baccarat). |  |  |
|  |  |  |
|  |  |  |

Table : <<enumeration name>> details

|  |  |
| --- | --- |
| Value | Notes |
| Rank | Enumerator for card rank |
| Suit | Enumerator for card suit |
| Choice | Enumerator for user choice |

# Class Diagram



# Sequence Diagram

A notebook with writing on it

Description automatically generated with low confidence