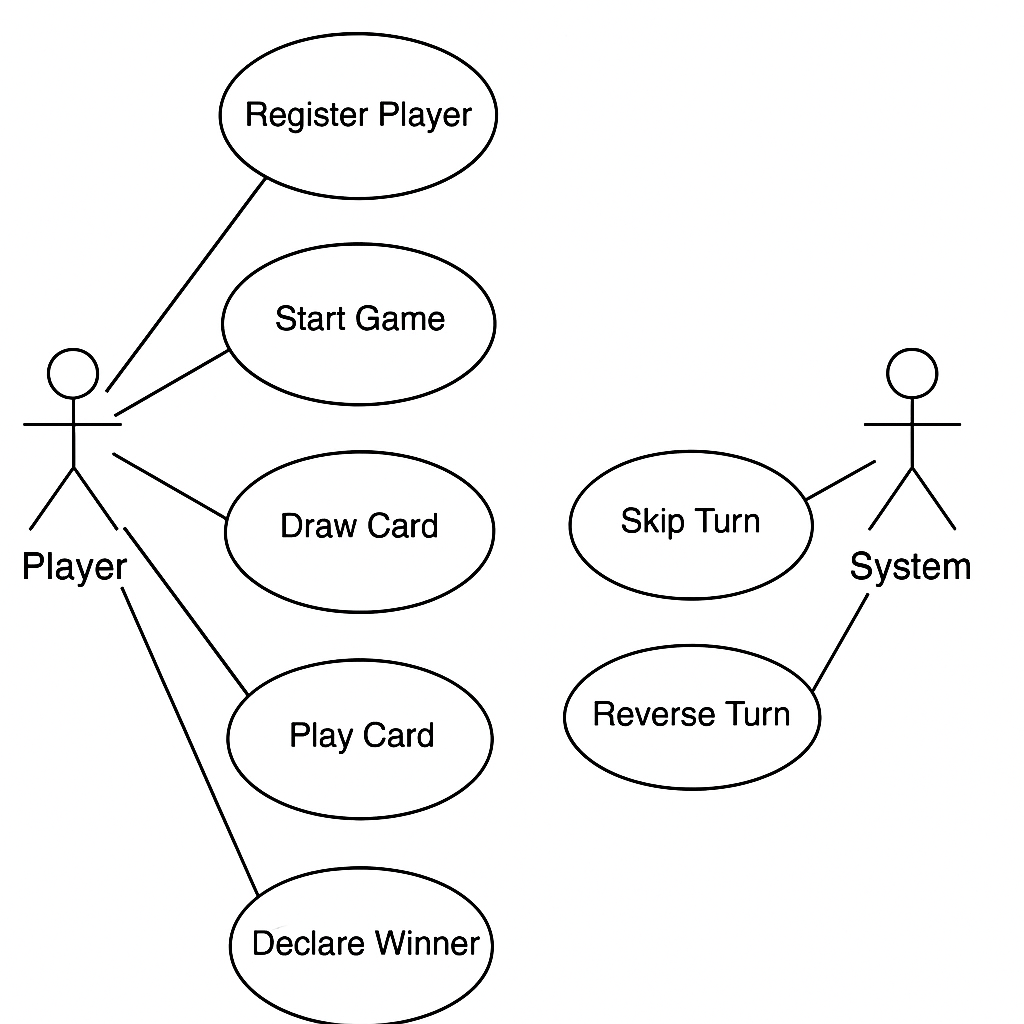
**Deliverable 3** for SYST17796 UNO Java OOP Game – Group 3

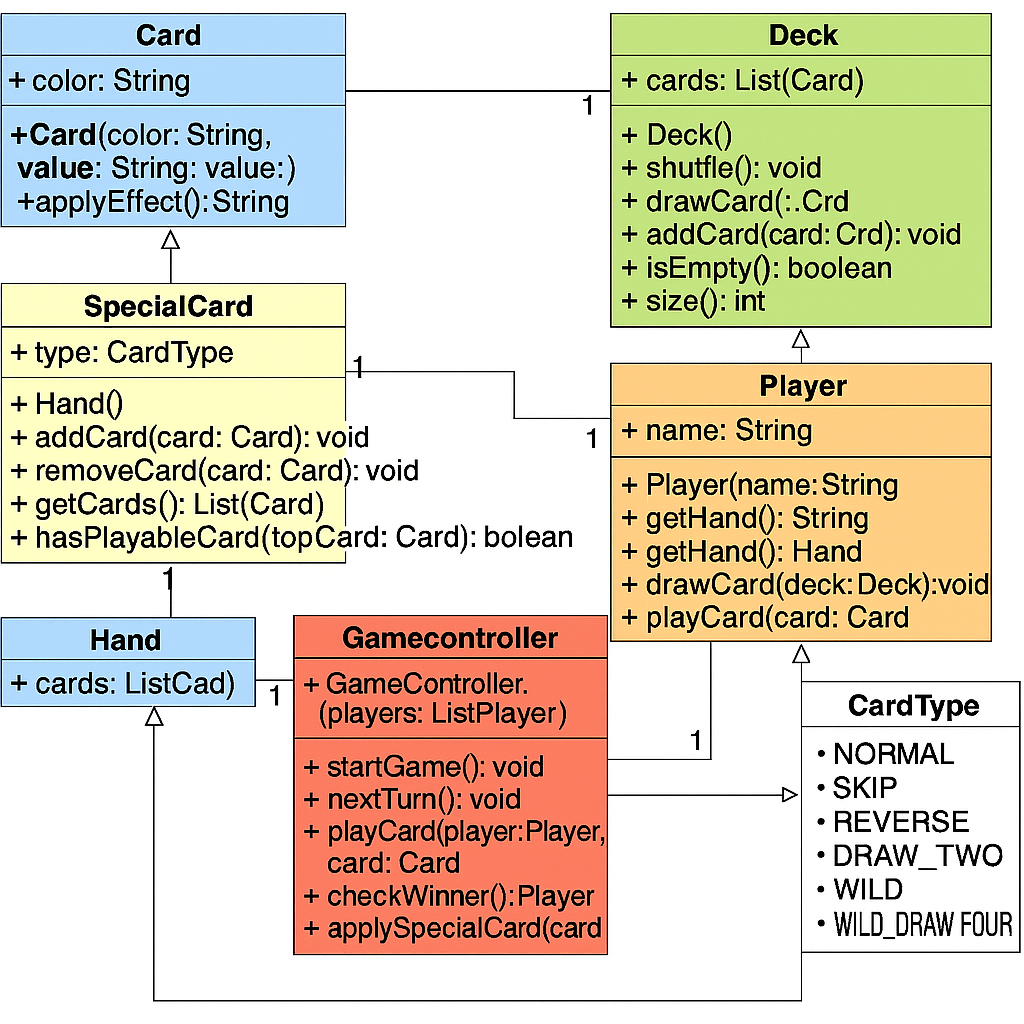
# Table of contents

* Use Case Diagram
* Class Diagram
* Source Code Design Summary
* Game Playability Notes
* GIT HUB - <https://github.com/Aulaprab/SYST17796--GROUP-PROJECT.git>

# Use Case Diagram



Class Diagram



# 

**1. Card**

* **Attributes:**
  + color: String
  + value: String
* **Methods:**
  + Card(color: String, value: String)
  + getColor(): String
  + getValue(): String
  + toString(): String

**2. SpecialCard (extends Card)**

* **Attributes:**
  + type: CardType
* **Methods:**
  + SpecialCard(color: String, value: String, type: CardType)
  + applyEffect(game: GameController, currentPlayer: Player)

**3. Deck**

* **Attributes:**
  + cards: List<Card>
* **Methods:**
  + Deck()
  + shuffle(): void
  + drawCard(): Card
  + addCard(card: Card): void
  + isEmpty(): boolean
  + size(): int

**4. Hand**

* **Attributes:**
  + cards: List<Card>
* **Methods:**
  + Hand()
  + addCard(card: Card): void
  + removeCard(card: Card): void
  + getCards(): List<Card>
  + hasPlayableCard(topCard: Card): Boolean

**5. Player**

* **Attributes:**
  + name: String
  + hand: Hand
* **Methods:**
  + Player(name: String)
  + getName(): String
  + getHand(): Hand
  + drawCard(deck: Deck): void
  + playCard(card: Card, discardPile: Deck): void

**6. GameController**

* **Attributes:**
  + players: List<Player>
  + deck: Deck
  + discardPile: Deck
  + currentPlayerIndex: int
  + direction: int
* **Methods:**
  + GameController(players: List<Player>)
  + startGame(): void
  + nextTurn(): void
  + playCard(player: Player, card: Card): boolean
  + checkWinner(): Player
  + applySpecialCard(card: SpecialCard): void

**7. Enum CardType**

* NORMAL, SKIP, REVERSE, DRAW\_TWO, WILD, WILD\_DRAW\_FOUR

# Source Code Design Summary

Encapsulation: getters/setters for controlled access, all fields private.  
  
Delegation: GameController invokes Player and Deck functions to manage the game flow.  
  
Cohesion: Every lesson focuses on a single duty.  
  
Low Coupling: Rather to using direct field access, classes communicate with one another through clearly defined methods.  
  
In order to apply unique UNO effects, SpecialCard inherits from Card.  
  
Polymorphism: The applyEffect() method's behavior is overridden by special cards.  
  
Flexibility: By extending CardType and modifying the logic in applySpecialCard(), new card types can be introduced.

Game Playability Notes

Notes on Game Playability: Supports two or more players.  
  
Deck: Contains special cards in addition to all of the regular UNO cards.  
  
The following special cards have been implemented: Draw Two, Skip, Reverse, Wild, and Wild Draw Four.  
  
Turn Logic: Control direction (reverse/clockwise).  
  
The winner is the first player with no cards.  
  
Deck management: The discard pile is reshuffled if the draw pile is empty.  
  
Interface: User activities are prompted via a console.