SYST17796 – Fundamentals of Software Design

Deliverable 1: Go Fish Card Game

Group: Dev Patel

Submitted By:

- Jaini

- Dev

- Parth

Date: June 10, 2025

Table of Content

1. Team Contract.......... 2

2. UML Diagram..........3

3. Design Document.......... 4

  3.1 Project Background and Description..........4

  3.2 Project Scope..........4

  3.3 High-Level Requirements.......... 5

  3.4 Implementation Plan.......... 6

  3.5 Design Considerations.......... 7

Team Contract

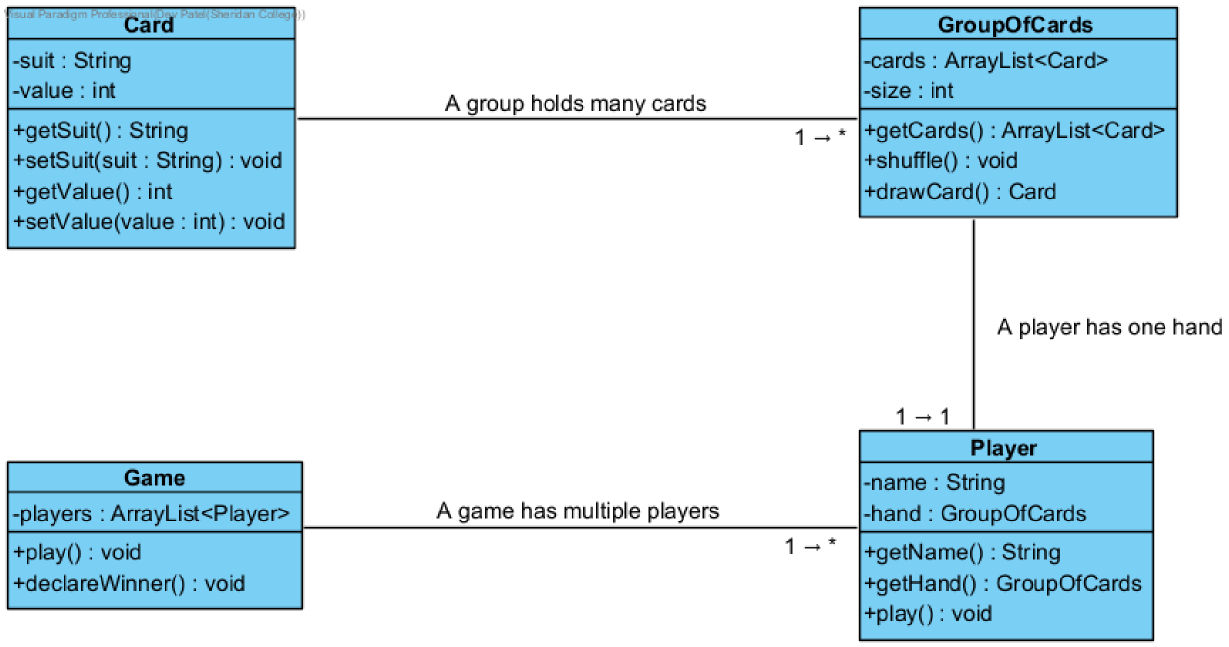
Group Members and Roles:

- **Jaini** – Game Logic Implementation
- **Dev** – UML Diagram + Documentation
- **Parth** – Planning + GitHub Repository Setup

Team Agreement:

- We agree to meet at least once per week to collaborate.
- Each team member is responsible for completing their assigned role on time.
- We will use GitHub for all version control and file sharing.
- We understand that all team members are equally responsible for academic integrity.
- Any issues in communication or workload will be discussed openly with the group.

UML Diagram


Visual Paradigm Professional(Dev Patel(Sheridan College))
Card
-suit : String
-value : int
+getSuit() : String
+setSuit(suit : String) : void
+getValue() : int
+setValue(value : int) : void
A group holds many cards
1 → *
GroupOfCards
-cards : ArrayList<Card>
-size : int
+getCards() : ArrayList<Card>
+shuffle() : void
+drawCard() : Card
A player has one hand
1 → 1
Player
-name : String
-hand : GroupOfCards
+getName() : String
+getHand() : GroupOfCards
+play() : void
A game has multiple players
1 → *
Game
-players : ArrayList<Player>
+play() : void
+declareWinner() : void

Design Document

1. Project Background and Description

We are implementing the classic card game Go Fish using Java. In this game, players take turns asking each other for cards of a particular rank. If the other player has any, they must give all of them. If not, the player must draw ("Go Fish") from the deck. The goal is to form as many “books” (sets of four cards of the same rank) as possible.

The provided base code includes four classes: Card, GroupOfCards, Player, and Game. These are written in Java using object-oriented principles such as encapsulation and delegation. Our goal is to extend this base to implement Go Fish logic fully, following clean, maintainable code practices.

2. Project Scope

Team Members and Roles:

- **Jaini** – Game logic implementation (coding core gameplay)
- **Dev** – UML diagram creation and full documentation
- **Parth** – Project planning and GitHub repository setup

Scope of the Project:
The project scope includes extending the base card game framework provided to implement a fully functional console-based Go Fish game. Players will be able to:

- Take turns asking for cards
- Draw cards from the deck when required
- Form books (sets of four cards of the same rank)
- Track and display scores
- End the game once all books are collected and declare a winner

The interface will remain console-based with text prompts for interaction. The project will be considered complete once all key game mechanics are implemented, tested, and playable from start to finish without errors.

3. High-Level Requirements

- Player Registration: Players must be able to register with the game (enter their names).
- Turn-Based Gameplay: Players will take turns asking another player for cards of a specific rank.
- Card Transfer: If the requested player has any matching cards, they must give all of them to the asking player.
- Go Fish Action: If no cards are available, the asking player must draw a card from the deck.
- Book Formation: When a player collects four cards of the same rank, they form a “book” which adds to their score.
- Score Tracking: The system must keep track of how many books each player has formed.
- Game End Condition: The game ends when all books have been collected.
- Winner Announcement: The game must calculate and display the winner (player with the most books).

4. Implementation Plan

Git Repository:
A GitHub repository has been created to manage and store the code, UML diagrams, and design documentation. All group members have access.

Repository URL:https://github.com/dpatelcsk/GoFishCardGame

Usage Strategy:

- Source code is stored under the /src directory following the package structure ca.sheridancollege.project.
- UML diagrams will be exported to the /uml directory.
- The design document and team contract are stored in the /docs directory.
- Commits are made regularly after major changes or new features are added.

Folder Structure:

```
GoFishCardGame/
├── src/
│   └── ca/sheridancollege/project/
├── uml/
└── docs/
```

Tools and Technologies:

- Java (JDK 17+)
- NetBeans
- Visual Paradigm
- GitHub

Coding Standards:

- Follow standard Java conventions: camelCase for methods/variables, PascalCase for classes
- Use private fields with public getters/setters to enforce encapsulation
- Keep methods short and modular
- Use clear comments and meaningful variable names

5. Design Considerations

Our design follows three key OOP principles: Encapsulation, Delegation, and Maintainability.

Encapsulation:

- Card class uses private fields accessed via getters/setters.
- GroupOfCards manages its internal card list with controlled public methods.

Delegation:

- Game delegates responsibilities like card drawing to GroupOfCards and player interaction to Player.

Maintainability:

- The modular structure allows the game to be extended or refactored without affecting core code.
- Go Fish logic can be added through subclassing or composition with minimal disruption.