### **SYST 17796 DELIVERABLE 1: DESIGN DOCUMENT**

## **OVERVIEW**

### **1. Project Background and Description**

The project aims to develop a simple card game where two players compete by drawing cards from a deck. The player who draws the higher card wins the round and earns a point. The first player to reach three points wins the game. This project helps in understanding basic game logic implementation and object-oriented programming.

**How to Play:**

* Each player takes turns drawing a card from a shuffled deck.
* The player with the higher card wins the round and scores a point.
* The game continues until a player reaches three points and is declared the winner.

**Reference to Rules:**

* Standard card values are used (2-10, Jack, Queen, King, Ace with Ace being the highest).
* Suits do not matter in this game.

**Current Starting Base Code:**

* **Language:** Java
* **Patterns:** The code follows the Model-View-Controller (MVC) pattern, separating game logic from the user interface.
* **Conventions:** Standard Java naming conventions (camelCase for variables/methods, PascalCase for classes), and code is documented using Javadoc.

### **2. Project Scope**

**Team Members:**

* Moksh Sharma: Project Manager
* Kenil Patel: Lead Developer
* Pramish Subedi: Developer
* Prajwol Lama: Quality Assurance

**Technical Scope:**

* **Interface:** The game will have a console-based interface allowing players to draw cards and see scores.
* **Completion Criteria:**
  + Functional console interface allowing card draws.
  + Score tracking and display.
  + Game logic correctly determines the winner.

### **3. High-Level Requirements**

The new system must include the following:

* Ability for each player to register with the game.
* Ability for the game to communicate a win or loss.
* Ability for players to know their status (score) at all times.

### **4. Implementation Plan**

**Git Repository URL:** [GitHub Repository URL]

**Expected Use:**

* Each developer will check in code at the end of each day/week.
* Text files, code, and UML diagrams are stored in separate directories:
  + /src for source code
  + /resources for project explanation docs
  + /uml for UML diagrams

**Coding Standards and Tools:**

* **Coding Standards:** Follow Oracle’s Java coding guidelines.
* **Tools:**
  + Visual Studio Code for development
  + Visual Paradigm (VP) for UML diagrams
  + JUnit for testing

### **5. Design Considerations**

**Current Code Structure and OO Principles:**

**Encapsulation:**

* The Card class has private fields for rank and suit, with public getters and setters.
* The Deck class encapsulates the card collection and provides methods for shuffling and drawing cards.

**Delegation:**

* The Game class delegates the task of drawing a card to the Deck class.
* The Player class delegates score tracking to a Scoreboard class.

**Flexibility/Maintainability:**

* The use of interfaces for the player and deck allows for easy substitution with different implementations (e.g., AI player).
* The MVC pattern separates concerns, making it easier to update the logic without affecting the overall structure.

**Examples from Base Code:**

1. **Encapsulation:**
   * private int rank;
   * public int getRank() { return rank; }
2. **Delegation:**
   * public Card drawCard() { return deck.draw(); }
3. **Flexibility/Maintainability:**
   * public interface Player { void playTurn(); }