Final Project

Art Dealer Game

Subject: Software Engineering

Submitted By: Group 5

Lewis University

October 16, 2024

Table of Contents

[Introduction 3](#_Toc179991556)

[Overview: 3](#_Toc179991557)

[Purpose and Objective: 3](#_Toc179991558)

[Game Mechanics: 3](#_Toc179991559)

[Patterns: 3](#_Toc179991560)

[Process Model: 4](#_Toc179991561)

[Use cases: 4](#_Toc179991562)

[Use Case 01: Start Game 4](#_Toc179991563)

[Use Case 02: Pick Cards 5](#_Toc179991564)

[Use Case 03: Dealer Feedback 5](#_Toc179991565)

[Use Case 04: Guess Pattern 5](#_Toc179991566)

[Use Case 05: Win Game 6](#_Toc179991567)

[Use Case 06: Lose Game 6](#_Toc179991568)

[Use Case 07: Reset Game 6](#_Toc179991569)

[Use Case 08: View Instructions 6](#_Toc179991570)

[Use Case 09: Exit Game 7](#_Toc179991571)

[UML Model: 7](#_Toc179991572)

[Use case Diagram: 7](#_Toc179991573)

[Deployment Diagram: 8](#_Toc179991574)

[Class Diagram: 9](#_Toc179991575)

[State Diagram: 10](#_Toc179991576)

[Activity Diagram: 11](#_Toc179991577)

[Customer Journey Map: 12](#_Toc179991578)

[Persona: 13](#_Toc179991579)

[Testing Strategy: 13](#_Toc179991580)

[ Unit Testing: 13](#_Toc179991581)

[ Integration Testing: 13](#_Toc179991582)

[ Functional Testing: 13](#_Toc179991583)

[ GUI (Graphical User Interface) Testing: 14](#_Toc179991584)

[ System Testing: 14](#_Toc179991585)

[ Regression Testing: 14](#_Toc179991586)

[ Usability Testing: 14](#_Toc179991587)

[ Performance Testing: 14](#_Toc179991588)

[ User Acceptance Testing (UAT): 14](#_Toc179991589)

# Introduction

## Overview:

In the pattern-matching game sometimes known as the Art Dealer Game, players pretend to be gallery owners. One looks for a hidden pattern an art dealer would be interested in purchasing by substituting the paintings for playing cards. Players try to estimate the card pattern the dealer will buy using just their own four-card hand and words. Project made using Python and Pygame developed a graphical user interface. comfort and simplicity of usage.

Starting the game as an art dealer who logically chooses a concealed pattern, the software represents to find this pattern, the player must closely watch the dealer's "buys," or matching, while selecting their own cards. After selecting four cards, players earn a visual bonus for accurately spotting the dealer's pattern.

## Purpose and Objective:

The Art Dealer Game is one educational logic game asking players to find trends and draw conclusions. The objectives are:

* So that sportsmen may improve their capacity to fit new trends.
* Gamification would help to make learning more interesting and enjoyable.
* Simple and neat design lets you pick cards and lay bets.

## Game Mechanics:

The game simulates an art gallery scenario where a player (gallery owner) attempts to figure out the pattern of cards the art dealer is willing to buy. The key game mechanics include:

* Card Selection: The player selects 4 cards from a displayed deck of cards.
* Pattern Matching: The dealer has a hidden pattern, and based on how many cards match the pattern, feedback is given to the player.
* Pattern Guessing: The player has to guess the hidden pattern from a set of possible patterns after selecting the cards.
* Rounds: The player has up to 3 rounds to guess the pattern, with each round providing new feedback based on the selected cards.

## Patterns:

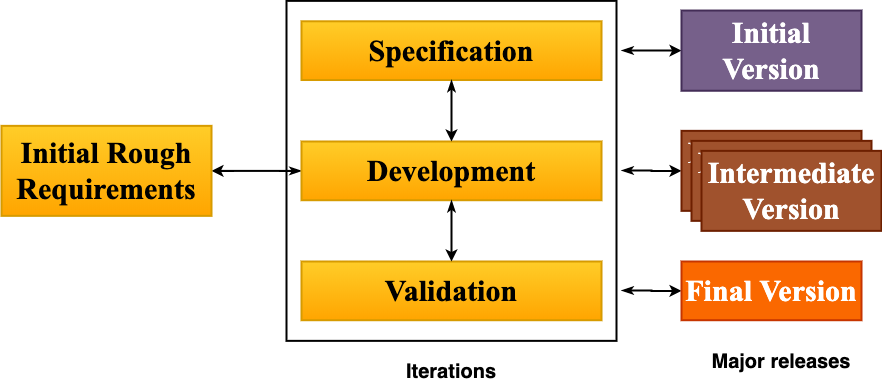
The hidden patterns are generated randomly at the start of each game. Possible patterns in the game include:

1. All face cards: All selected cards must be Jacks, Queens, or Kings.
2. Sum equals 10: The sum of the ranks of the selected cards must be exactly 10.
3. All black cards: All selected cards must be from the black suits (spades or clubs).
4. Two pairs: The selection must consist of two pairs of cards with the same rank.
5. Sequential cards: Cards must be in sequential order (e.g., 4, 5, 6, 7).

# Process Model:

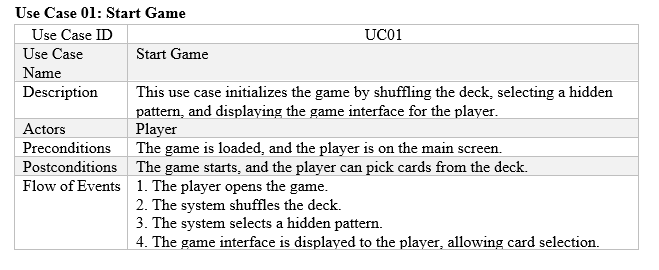
The Evolutionary Process Model is an adaptive approach to software development that allows the system to evolve over time as new requirements emerge or as users interact with the system. This model is often used when the requirements are not fully known at the beginning of the project, or when there is a need for rapid development of working versions.

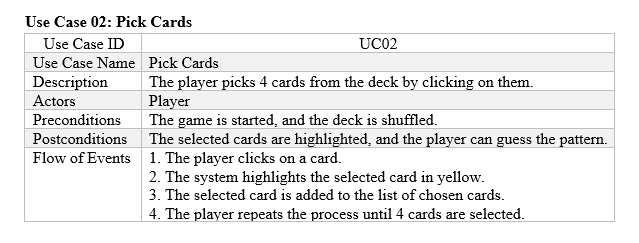
The Evolutionary Model involves developing an initial version of the system, getting feedback, and refining the system through multiple cycles (iterations) of development. It is highly suited for projects like games where continuous feedback and changes are often required to improve user experience and gameplay.

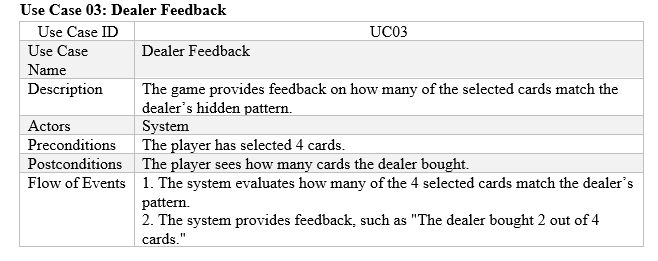


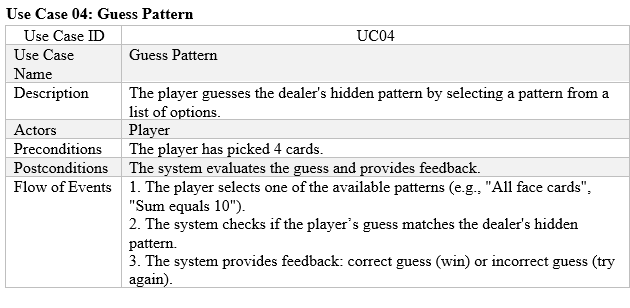
*Fig.1 Evolutionary Process Model*

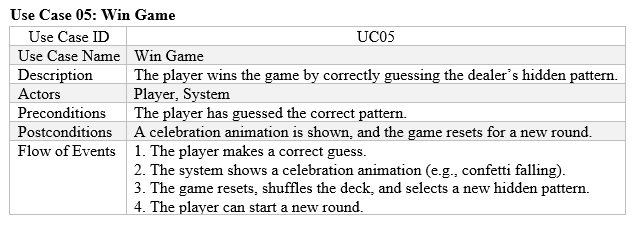
# Use cases:

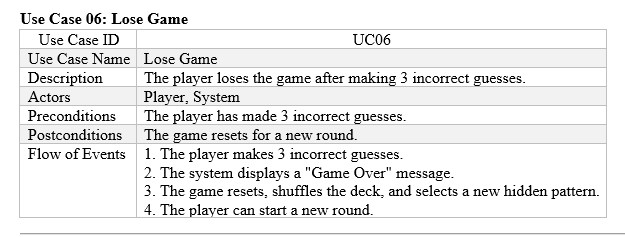


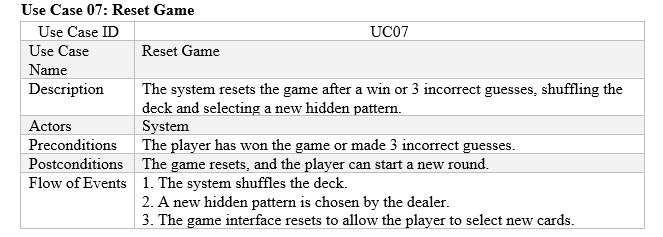


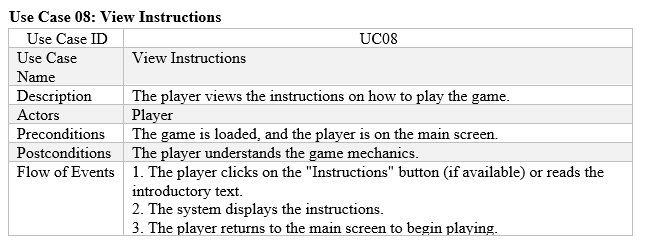


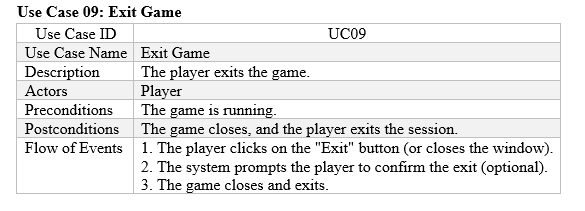






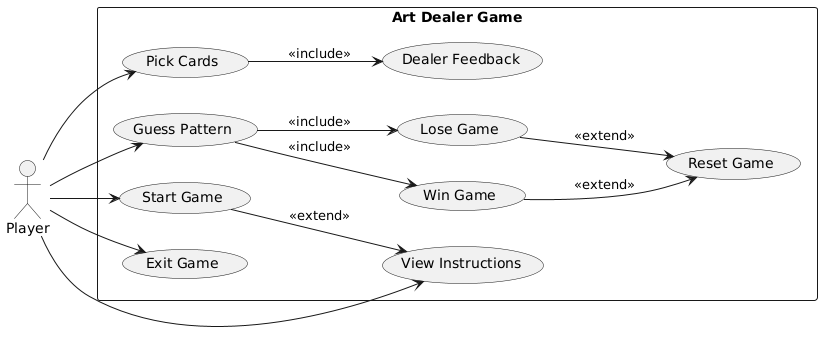






# UML Model:

## Use case Diagram:

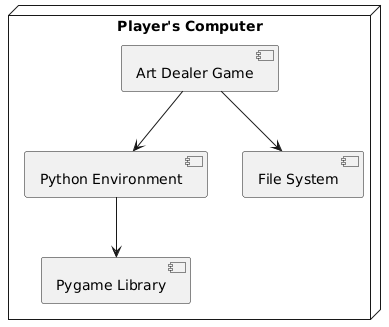


*Fig.2 Use case diagram*

## Deployment Diagram:

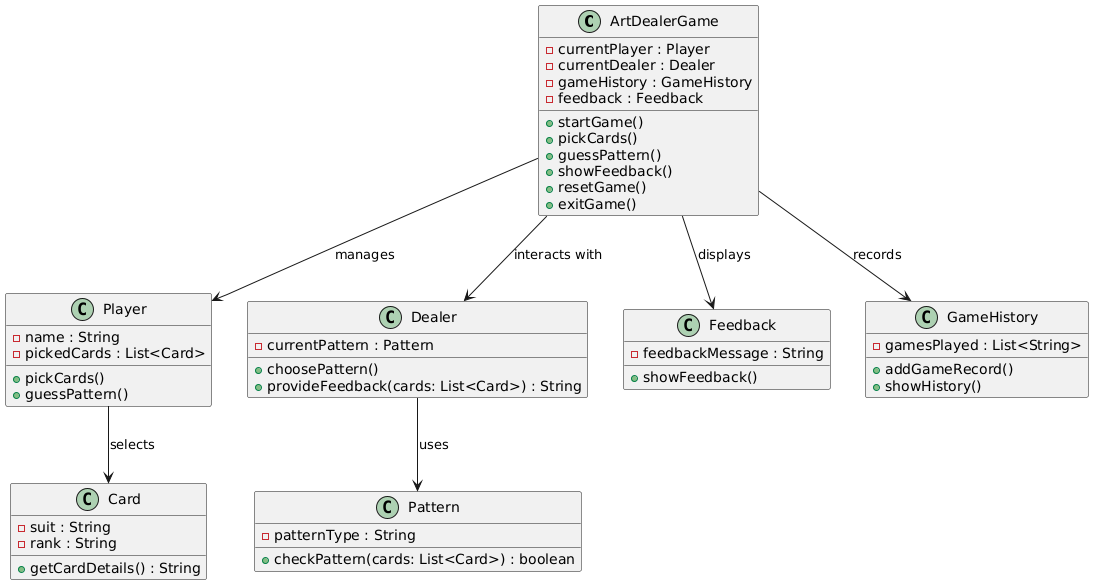
The Art Dealer Game is a desktop application built using Python and the Pygame library. Below is a description of the deployment architecture, assuming a scenario where the game is deployed on multiple user devices, such as personal computers.

* Player's Computer:
  + The player's personal computer where the Art Dealer Game is installed and run.
  + The system includes the game application along with Python and necessary libraries (like Pygame) to execute the game.
* Application Components:
  + Art Dealer Game: The Python-based executable of the game that includes game logic, card management, pattern guessing, feedback, and the graphical user interface.
  + Python Environment: The runtime environment required to execute the Python code. Includes Python interpreter and external libraries such as Pygame.
  + Pygame Library: The external library used to handle graphical user interface components and game logic.



*Fig.3 Deployment Diagram*

## Class Diagram:



*Fig.4 Class Diagram*

* **ArtDealerGame Class**

This class is the main controller of the game. It manages the entire game flow, including starting the game, allowing the player to select cards, making guesses, and showing feedback. It also interacts with other components like the Player, Dealer, and GameHistory classes. The ArtDealerGame class is responsible for resetting the game and managing the overall logic.

* **Player Class**

The actor presents herself as a Player class instance. Every round the player's name and the cards they chose are noted. Playing this card game successfully means carefully selecting your cards to expose the dealer's hidden tactic. This class logs every choice the player takes. The actor of the action takes control in the class Player. Every round the player's name and the cards they chose are noted. Playing this card game well means deliberately selecting your cards to expose the dealer's hidden plan. Every single action the player does here is logged.

* **Dealer Class**

The Dealer class captures the dealer's choice on the player's secret pattern guessing assignment. The player uses the pattern to ascertain the performance level of a deck. Through choosing and maintaining the secret pattern, the dealer regulates the main difficulty of the game.

* **Card Class**

Information about every playing card—including rank (two, three, jack, ace), suit (hearts or spades), and other specifics—is tracked by the Card class. Both dealers and gamers use this class to make decisions grounded on card pattern trends.

* **Pattern Class**

The Pattern class defines what happens should the selected cards fit the dealer's concealed pattern. Here among the statements covered are "all face cards" and "sum equals 10." The fraction of matching cards in this class can be ascertained by the dealer from the player's pattern choices.

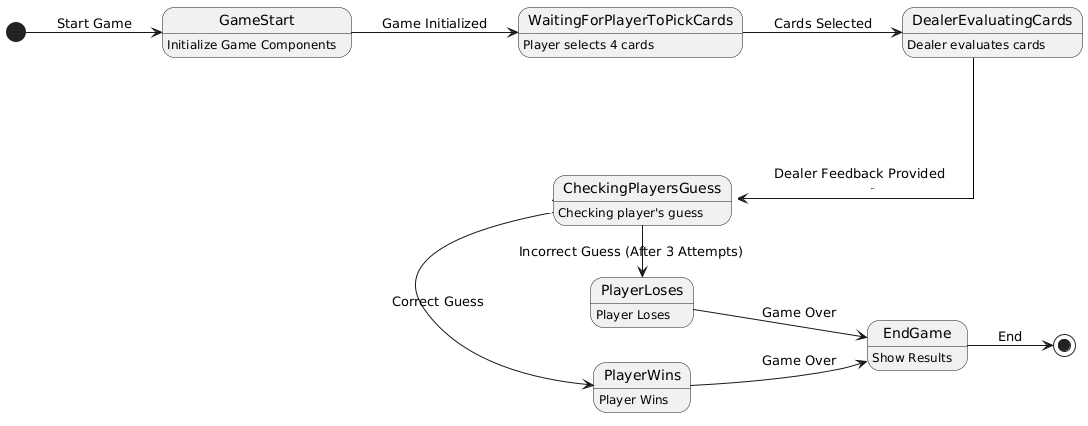
* **Feedback Class**

The Feedback class displays either remarks or results following a player's card choice or pattern prediction. Based on the cards that fit the dealer's pattern, here are your forecasts—either accurate or not. It is in responsibility of the messaging system of the game.

* **GameHistory Class**

The GameHistory class keeps track of all the games played, storing details like previous attempts, the cards picked, and whether the player won or lost. It allows players to review their past performance and see how many rounds it took them to guess the pattern correctly.

## State Diagram:

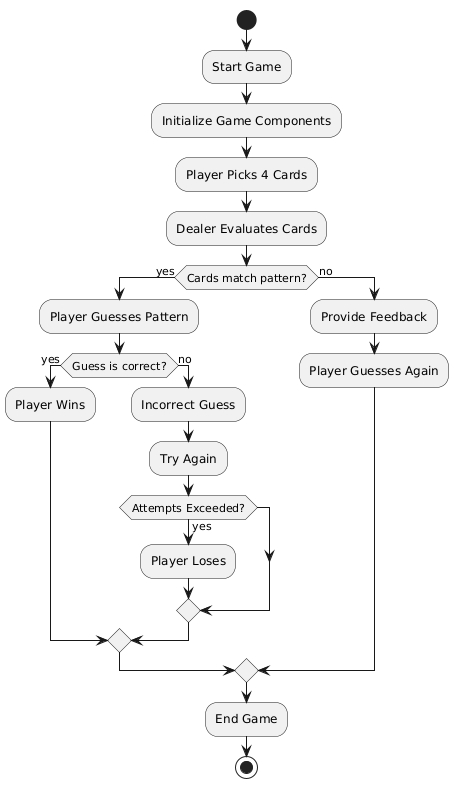


*Fig.5 State Diagram*

The state diagram for the Art Dealer Game represents the flow of the game through various stages. The game begins in the Start Game state, where necessary components such as the player, dealer, and cards are initialized. Once initialized, the game moves to the WaitingForPlayerToPickCards state, where the player selects four cards. The dealer then evaluates these cards in the DealerEvaluatingCards state. The feedback is provided to the player, and the game transitions to the CheckingPlayersGuess state, where the player's guess is evaluated based on the dealer's hidden pattern.

From the CheckingPlayersGuess state, two outcomes are possible: a correct guess leads to the PlayerWins state, while an incorrect guess (after three attempts) leads to the PlayerLoses state. In both cases, the game transitions to the EndGame state, where the results are displayed to the player. Once the game ends, the diagram completes the cycle, either ending the game or resetting for a new round. The state diagram visually captures the core stages of the game's flow, providing a clear understanding of the transitions between game events.

## Activity Diagram:



*Fig.6 Activity Diagram*

The **activity diagram** for the **Art Dealer Game** illustrates the sequence of actions that occur throughout the game. The game starts with the initialization of essential components such as the player, dealer, and deck of cards. Once initialized, the player is prompted to select four cards, which the dealer then evaluates against a hidden pattern. If the dealer's evaluation finds a match, the game proceeds to the guessing phase where the player attempts to guess the pattern. If the player's guess is correct, the player wins; otherwise, the player continues to make guesses until either the correct pattern is identified or the maximum number of attempts is reached.

In the event the player exceeds the allowed number of attempts without guessing correctly, the game transitions to the losing state. Regardless of whether the player wins or loses, the game reaches the **End Game** activity, signaling the conclusion of that round. The diagram highlights the main game loop, emphasizing the decision points where feedback is provided, guesses are evaluated, and actions are repeated as necessary, giving a clear overview of the flow and conditional steps that drive the game.

# Customer Journey Map:

1. Goal Setting (Start)
   * The player sets the goal of figuring out what types of cards the art dealer (software) wants to buy.
   * This involves identifying patterns like "all red cards" or "cards that sum up to 9."
2. Placing Cards
   * The player begins by laying out 4 cards from their hand.
   * The art dealer evaluates these cards and gives feedback on how many of the cards fit the pattern they’re looking for.
3. Evaluating Feedback
   * The player receives feedback about how many of the cards were purchased by the art dealer.
   * The player considers this feedback and tries to identify the pattern based on the cards selected.
4. Guessing the Pattern
   * After several rounds, the player feels confident and guesses the pattern the art dealer is looking for.
   * They use a dropdown or menu to choose the pattern (e.g., "All hearts").
5. Celebration/Results
   * If the player correctly guesses the pattern, the game celebrates with a winning animation (such as balloons).
   * If they guess incorrectly, the player continues guessing until they get it right or exhaust the number of guesses allowed.
6. End Goal (Success)
   * Once the player identifies the pattern, they reach the goal and can save their score or play another round.
   * A victory message or animation displays their success.

# Persona:

|  |  |
| --- | --- |
| Attribute | Details |
| Name | Harleen Kaur |
| Age | 12 years old |
| Grade | 6th Grade |
| Background | Emily loves puzzles and card games. She's skilled in recognizing patterns and enjoys math-related challenges. She has basic experience with online games and enjoys playing them for fun after school. |
| Technology Comfort | Comfortable using computers, tablets, and online games. She quickly learns new systems but prefers simple interfaces. |
| Interests | Puzzle games, card games, and strategy-based challenges. |
| Goals and Motivations | Enjoys figuring out patterns, wants to improve her problem-solving skills, and enjoys visual celebrations when winning. |
| Frustrations | Complicated interfaces, repetitive gameplay, and a lack of feedback that makes the game feel stagnant. |
| Needs | Clear, immediate feedback on card selections, simple controls, and motivational visual rewards like animations when she wins. |

# Testing Strategy:

The testing strategy for the Art Dealer Game is designed to ensure that the game functions correctly, provides the intended experience to the user, and meets the project requirements. The testing strategy will cover different types of testing at various stages of development to catch issues early, validate functionality, and ensure a smooth user experience.

* Unit Testing: This type of testing focuses on verifying individual components or units of the game, such as the functionality of selecting cards, dealer feedback, or the pattern-matching logic. By isolating and testing each function independently, unit tests ensure that all the smallest pieces of the program work as intended. For example, testing a function that checks whether a card matches a specific pattern will help ensure that the game mechanics function correctly before integrating other components.
* Integration Testing: Once individual units have been tested, integration testing ensures that different components of the game (e.g., the player’s card selection, the dealer’s feedback, and the guessing system) work together seamlessly. In this phase, we would test scenarios where the player selects cards, the dealer evaluates them, and feedback is provided in the game’s user interface. Integration testing checks the interactions between these modules and ensures they work well together.
* Functional Testing: Functional testing ensures that the game behaves according to the specified requirements. In this case, the tester would verify that all user actions, like selecting cards, receiving dealer feedback, guessing patterns, and winning or losing, work as expected. The goal is to ensure that the game adheres to its intended functionality from a user’s perspective.
* GUI (Graphical User Interface) Testing: Since the Art Dealer Game is a graphical game with buttons, feedback messages, and animations, GUI testing will focus on ensuring that all visual components, such as buttons, input fields, and feedback messages, work correctly and are visually consistent. This includes testing interactions like the card selection mechanism, ensuring that buttons are clickable, and that feedback and animations (like celebration when a player wins) are displayed appropriately.
* System Testing: An overall picture offered by system testing guarantees the game satisfies all project criteria. From dealer comments and pattern guessing to game resets and player interactions, this comprehensive test covers all aspect of the game. This will ensure that the game satisfies all performance and functionality criteria set for the project and runs without problems.
* Regression Testing: An important first step is regression testing to make sure program enhancements or fixes won't compromise current functioning. Regression testing is used to ensure that recently introduced features—such a pattern to guess—are suitably rolled back to the original patterns and that the current patterns remain as predicted.
* Usability Testing: Usability testing is crucial to identify flaws and ensure that everyone may easily understand the game. For players, consistently trusting their card selections, comment interpretations, and guesses is absolutely vital. Starting with a lovely, simple game where the player can progress through several stages.
* Performance Testing: Performance testing mostly aims to make sure the game operates as expected in all possible conditions. Main aspects of this assessment are game responsiveness to card selection or input, loading times, and prolonged playing. The tester will find trends in the player's erroneous card choices or estimations while they are playing.
* User Acceptance Testing (UAT): User acceptability testing (UAT) is absolutely vital to ensure the game lives up to its promises, particularly because these folks will be the final ones to play it. This assessment will guarantee that the game faithfully reflects the experience of its intended users, thereby attaining its objectives. At this point, actual users alone can provide comments on the game's enjoyment, usefulness, and instructional value.