**SYST17796**

**Deliverable 2**

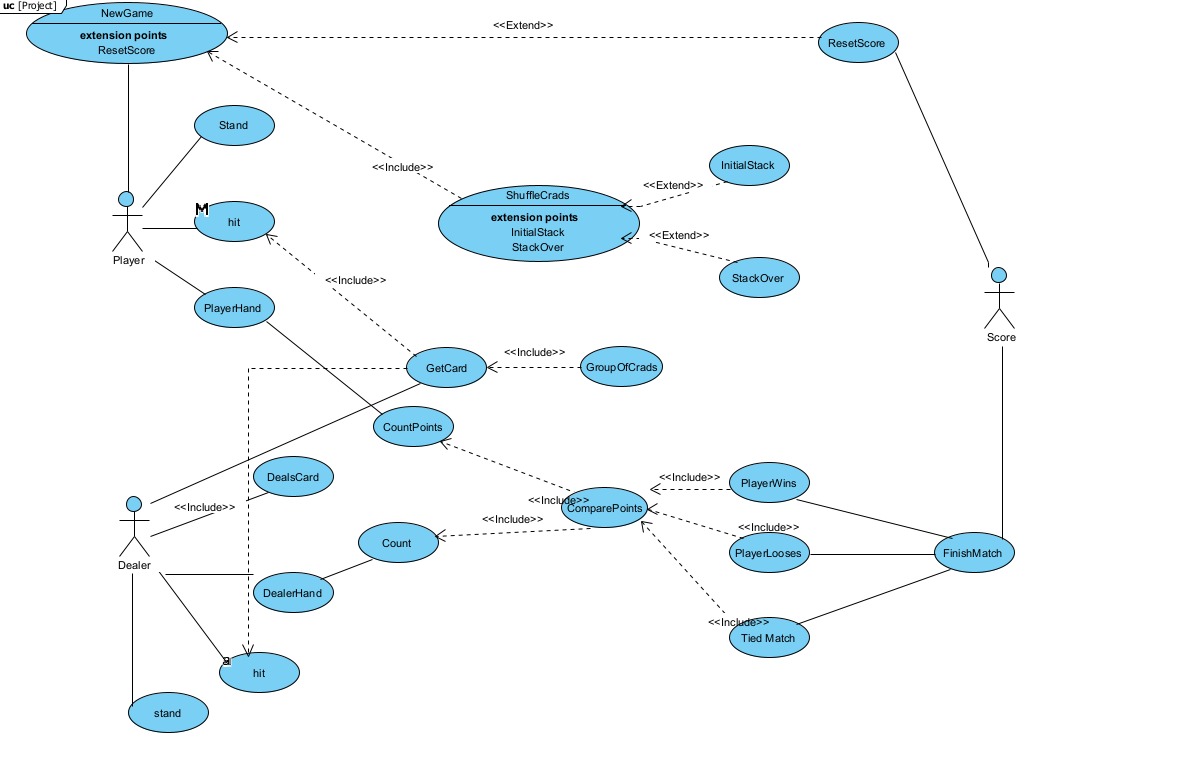
**Team Name: Amigos**

**UML Class Diagram**

A picture containing text, sign, screenshot

Description automatically generated

**Use Case Diagram**



**(1)Project Background and Use Case Description**

The objective of the game is to get a total score of 21 without going over or else the player gets busted. There needs to be a minimum of one player and a dealer for the game to function efficiently.

The primary actor is the player. The dealer here is the computer that deals out and shuffles the card. The secondary actor is the score as it helps the player to finish the game by providing necessary information to the system.

2 cards are handed out at first. If the dealer and/or the player decide to hit during their turn one card is handed to them from the deck.

The count of the Player’s hand and Dealer’s Hand is kept and compared after every move. If the player gets a total of 21, the player wins. If the dealer reaches a total of 21 before the player or player’s score gets more than 21, the player loses. If both get a total of 21 then the match is tied. Either of 3 ways, the match finishes.

If the player wishes to play a new game, the scores are reset and the game starts all again.

**(2) Design Consideration**

The UML diagram depicts various OOP principles followed in this code. To begin with, most of the methods and properties are kept private to follow **encapsulation**. Others will be made private wherever possible along the way to the development of code. Another example of encapsulation is using enums to declare the suit and rank of the card.

Each class is given a specific task to follow **cohesion**. This in terms makes the program **loosely coupled** because changes made in one class won’t affect another.

The dealerHand and PlayerHand both will extend the Hand class thereby satisfying **Inheritance.**

**Aggregation** can be seen between Game and Player as the player is a part of the game and followed by **Composition** is observed Hand and Count because the class Hand cannot exist without Count.