**Advanced Blackjack Strategy  
Software Requirements Specification**

5-12-2021 - v1.0

C.Finch (cmf)

E. Mielke (enm)

E. Hsu (eh)

Y. Chen (yc)

**Table of Contents**

**1. SRS Revision History [2](#_heading=h.30j0zll)**

**2. The Concept of Operations (ConOps) [2](#_heading=h.1fob9te)**

**2.1. Current System or Situation [2](#_heading=h.3znysh7)**

**2.2. Justification for a New System 2**

**2.3. Operational Features of the Proposed System 2**

**2.4. User Classes 3**

**2.5. Modes of Operation 3**

**2.6. Operational Scenarios (Also Known as “Use Cases”) 3**

**3. Specific Requirements 4**

**3.1. External Interfaces (Inputs and Outputs) 4**

**3.2. Functions 4**

**3.3. Usability Requirements 5**

**3.4. Performance Requirements 5**

**3.5. Software System Attributes 5**

**4. References 5**

**5. Acknowledgements 6**

# 

# 

# 

# 

# 1. SRS Revision History

**Date Author Description**

**————— ——— ————————————————————————————**

5-10-2021 enm Created Initial Document

5-11-2021 enm Completed 2.1

5-11-2021 enm Completed 2.2

5-11-2021 enm Completed 2.3

5-11-2021 enm Completed 2.4

5-11-2021 enm Completed 2.5

5-11-2021 enm Completed 2.6

5-12-2021 enm Completed 3.5

5-12-2021 enm Completed 3.4

5-12-2021 enm Completed 3.3

5-12-2021 enm Completed 3.2

5-12-2021 enm Completed The First SRS Draft

5-12-2021 eh Edited 3.1

5-12-2021 enm Elaborated on 2.1

# 2. The Concept of Operations (ConOps)

## 2.1. Current System or Situation

The game Blackjack is a popular game at casinos where you try to beat the card dealer by having the hand with a higher total than the dealer's hand without going over 21. Aces are worth 1 or 11 points based on context, the number cards are equal to their number and face cards are worth 10 points. While there does not truly exist a fair casino game, this game is commonly known as one of, if not the most fair game in said casinos with a house edge of approximately 1.5%. Craps being a close second at approximately 1.4-5% in house favor depending on the type of bet.

## 2.2. Justification for a New System

Due to Blackjack being inherently a game, an individual should be able to improve at playing it, either for playing with friends or playing at a casino. The goal of this new system is to be an interactive way to teach someone how to count cards in order to become better at the game.

## 2.3. Operational Features of the Proposed System

This system aims to teach players how to properly “count cards” in Blackjack to minimize the house edge as much as possible by having an interactive teaching system that will give the user the best advice as to how to count cards given known cards on the table.

This system will be able to give advice towards when to hit or stay and also when to bet optimally, while being a guided interactive system with all the advantages of interactive experience over reading about such strategies in a book or online.

## 2.4. User Classes

There exists two user classes, the main user class being the actual “Blackjack player” where the system is build around teaching them how to play the game, and the “Maintenance class” which is more of a debug/settings class than anything else, where you could input settings to make sure the system is set up properly for the user.

## 2.5. Modes of Operation

There exists two modes of operation for this system, a user mode and a maintenance mode, the user mode is all the modules the user will see and interact with. This includes the PyGame component and launching the program. Then there is the Maintenance use case where much of the actual code exists, where a developer could tweak settings.

## 2.6. Use Cases

**Use Case: An individual tries to start and use the program.**

***Brief description:*** This use case describes how a user starts the program and starts a game

***Actors:*** User and Simulations module

***Preconditions:***

1. The individual has a device that currently has the program installed.

2. The individual has a device that can interact with the contents of the PyGame interface

***Steps to Complete the Task:***

1. The individual opens the program by double clicking the main file (probably will be named main.py at release)

***Postconditions:***

The user now has access to a system that can teach them how to count cards in blackjack.

**Using this prior use case you have now multiple options:**

**1.**

**Use Case: An individual wants to practice**.

***Brief description:*** This use case describes how a user could use the program to practice.

***Actors:*** User and Simulations module

***Preconditions:***

1. Finish the *“*An individual tries to start and use the program.” use case.

***Steps to Complete the Task:***

1. From the main menu select “Strategy Practice”

***Postconditions:***

This allows the user to play the different blackjack hands and hone their skills.

**2.**

**Use Case: An individual wants to test their abilities.**

***Brief description:*** This use case describes how a user could use the program to play a mock game of blackjack.

***Actors:*** User and Simulations module

***Preconditions:***

1. Finish the *“*An individual tries to start and use the program.” use case.

***Steps to Complete the Task:***

1. From the main menu select “Blackjack Game”

***Postconditions:***

This allows the user to test their skills under a full betting system true game of

blackjack with minor hints.

# 3. Specific Requirements

## 3.1. External Interfaces (Inputs and Outputs)

Simulation Module:

Input

1. Playing a Game
2. Select Training game or Real game.
3. In the game the user has to select from a variety of hand options (elaborated upon in the functions portion)
4. After selecting a round ending move the game goes to the end update.

Output

1. End Updates
2. When a game has a change in cards, the game updates the table and shows the player the new cards + new odds.
3. When the round ends the game shows the player if they won or not.

## 3.2. Functions

Simulation Module - Input:

Playing a hand:

1. The user will receive their cards and the Blackjack “Table” will be set up
2. Upon pressing the corresponding commands the following will happen:
   1. HIT: The user receives an additional random card and the table updates accordingly
   2. STAND: The user maintains their hand and the table updates accordingly.
   3. DOUBLE DOWN: Double down rules in casino blackjack happen as normal and the table updates accordingly. (<https://www.blackjackapprenticeship.com/how-to-play-blackjack/>)
   4. SPLIT: Split rules in casino Blackjack happen as normal and the table updates accordingly.

(<https://www.blackjackapprenticeship.com/how-to-play-blackjack/>)

* 1. SURRENDER Surrender rules in casino Blackjack happen as normal and the table updates accordingly.

(<https://www.blackjackapprenticeship.com/how-to-play-blackjack/>)

1. The user then receives an output describing their hand, if they won and how much they won

## 3.3. Usability Requirements

This will be organized to have few, very noticeable, descriptive buttons that will limit ambiguity for the user. The user should be able to simply learn how the program works by looking at it once without issue. This is essential as many people just want a simple drop in and drop out program, a simple program without need of a “README” file or a manual is ideal for this case.

## 3.4. Performance Requirements

Static:

-The calculations for the potential blackjack hand probability should be done in less than a second.

-Recommended Betting will similarly be done in about less than a second.

Dynamic:

-The hand at any given time will wait until the user decides their next move.

## 3.5. Software System Attributes

The Advanced Blackjack Strategy program is designed with reliability and performance in mind, in order to stay as stable as possible we will build this to be lightweight and have as few moving parts as possible, which will additionally play a hand in performance. Performance is important to this program as a slow program will be un-intuitive for users and may even be frustrating.

# 4. References

IEEE Std 1362-1998 (R2007). (2007). IEEE Guide for Information Technology–System Definition–Concept of Operations (ConOps) Document. <https://ieeexplore.ieee.org/document/761853>

IEEE Std 830-1998. (2007). IEEE Recommended Practice for Software Requirements Specifications. <https://ieeexplore.ieee.org/document/720574>

ISO/IEC/IEEE Intl Std 29148:2011. (2011). Systems and software engineering — Life cycle processes — Requirements engineering. <https://ieeexplore.ieee.org/document/6146379>

ISO/IEC/IEEE Intl Std 29148:2018. (2018). Systems and software engineering — Life cycle processes — Requirements engineering. <https://ieeexplore.ieee.org/document/8559686>

Faulk, Stuart. (2013). *Understanding Software Requirements*. <https://projects.cecs.pdx.edu/attachments/download/904/Faulk_SoftwareRequirements_v4.pdf>

Oracle. (2007). White Paper on “Getting Started With Use Case Modeling”. Available at: <https://www.oracle.com/technetwork/testcontent/gettingstartedwithusecasemodeling-133857.pdf>

van Vliet, Hans. (2008). *Software Engineering: Principles and Practice*, 3nd edition, John Wiley & Sons.

# 5. Acknowledgements

This is based off of the SRS template made by Anthony Hornof for use in his CIS 422 class.