HYBRID DESIGN DOCUMENT

SOFTWARE ENGINEERING PRACTICES – Assignment 2 of 2

Author: Tomas Atanasov
Tutor: Chris Green

Contents

Introduction

As a part of the interview for a junior developer position at the emerging mobile game development company Nine-Nine Ltd, I'm required to present this hybrid document that outlines the design and technical structure for the development of a new python project.

Overview of Game

The project target is a straightforward card game, which features a big deck of 56 unique cards, and each player receives one card at a time, then players take turns playing the card from their hand, with the highest played card winning the round, then the game continues until one player has 28 points accumulated. Using the Pygame library in this project, the aim is to combine strategic gameplay with engaging mechanics through algorithm that allows the program to shuffle and deal the deck of cards, keep track of each player's hand, and determine the winner of each round.

Requirement of Document

Game design document, according to the blog of French, J (2024) is not mandatory, and using it is a personal preference, but it represent a framework of ideas and guidelines about the project which steer the technical team into the right direction, and this can be crucial for the development of the project.

Technical design document, in the other hand, is presented by studytonight (2024) as a clear technical plan for the software development team to implement core features and mechanics through coding.

Therefore, following a structured approach, this hybrid document combines the contents of the game design documentation (GDD) and technical design documentation (TDD) into a single complete development blueprint for the game.

Game Design

Game Mechanics

The game will be designed as a Single Player to begin with, with a possibility to expand to Multiplayer in the future. The ‘Win condition’ is going to be decided first by the points counter, with highest points card winning a round, and secondly by the number of rounds won. After opening the application, the initial state of the game is the main menu, where players can choose to either play the game, read the manual, or quit. Menu interaction, drawing cards, and restarting the game are currently the only mechanics of the game, but more mechanics like comparing card values, and win condition, are currently under development.

Game Mechanics	Description
Main Menu interaction	• Start Button: Transitions the game to the "game" state. • How to Play Button: Transitions the game to the "how_to_play" state. • Quit Button: Exits the game.
Deal Cards	A standard deck of 52 cards is shuffled. Cards are dealt to each player, and each player receives one card at a time (adjustable by cards_per_player).
Restart	When the "Restart" button is clicked, it reshuffles the deck, return the cards to the deck, and resets any variables to their initial states. This ensures a fresh start.
Comparing Card Values	Comparing the values of the cards players draw.
Win Condition	Determining which player win the Game based on the number of points they have accumulated
Points counter	Determining which player wins the Round based on the values of the cards they draw.

Level Design

The level design of my game includes the ‘Main Menu level’ as shown in Appendix #, which present the player with the choice to either play the game, read the game rules, or quit the game.

Next level available in the game is the ‘How to play’ level and it provides text instructions about the game, as well as a ‘go back’ button which can bring the player back into the previous game state.

Last but not the least, we have the most complicated ‘Game’ level, the ‘Restart’ button in this level allows players to reset the game without restarting the application, providing a quick way to replay the game, while the "Draw" button is a central mechanic in the game, which allows players to draw cards from the deck and facilitate the primary gameplay loop.

UI/UX Design

The UI of the game begins with the ‘Main Menu’ background image ‘menu1.png’ from Appendix 3, scaled to fit the screen, background theme music for the game ‘casino_music.wav’ displayed in Appendix 1, and buttons with integrated sound effect ‘select.wav’ presented in Appendix 2. The whole finished state can be seen in Appendix 7.

Next state UI available in the game is in the ‘How to Play ‘level, which can be seen in Appendix 8, it includes the background image ‘startgame.png’ from Appendix 5, as well as text instructions explaining the game rules and mechanics, and a ‘Go Back’ button that also have the ‘select.wav’ sound effect from Appendix 2.

Lastly, we have the UI design of the ‘Game’ state shown in Appendix 9, which is composed of the ‘startgame.png’ image from Appendix 4, different buttons like ‘Draw, Restart, Go Back, and Quit’, all of which have different functions as well as entirely different sound effects assigned to them, as presented in Appendix 2.

Technical Architecture

Game flow chart

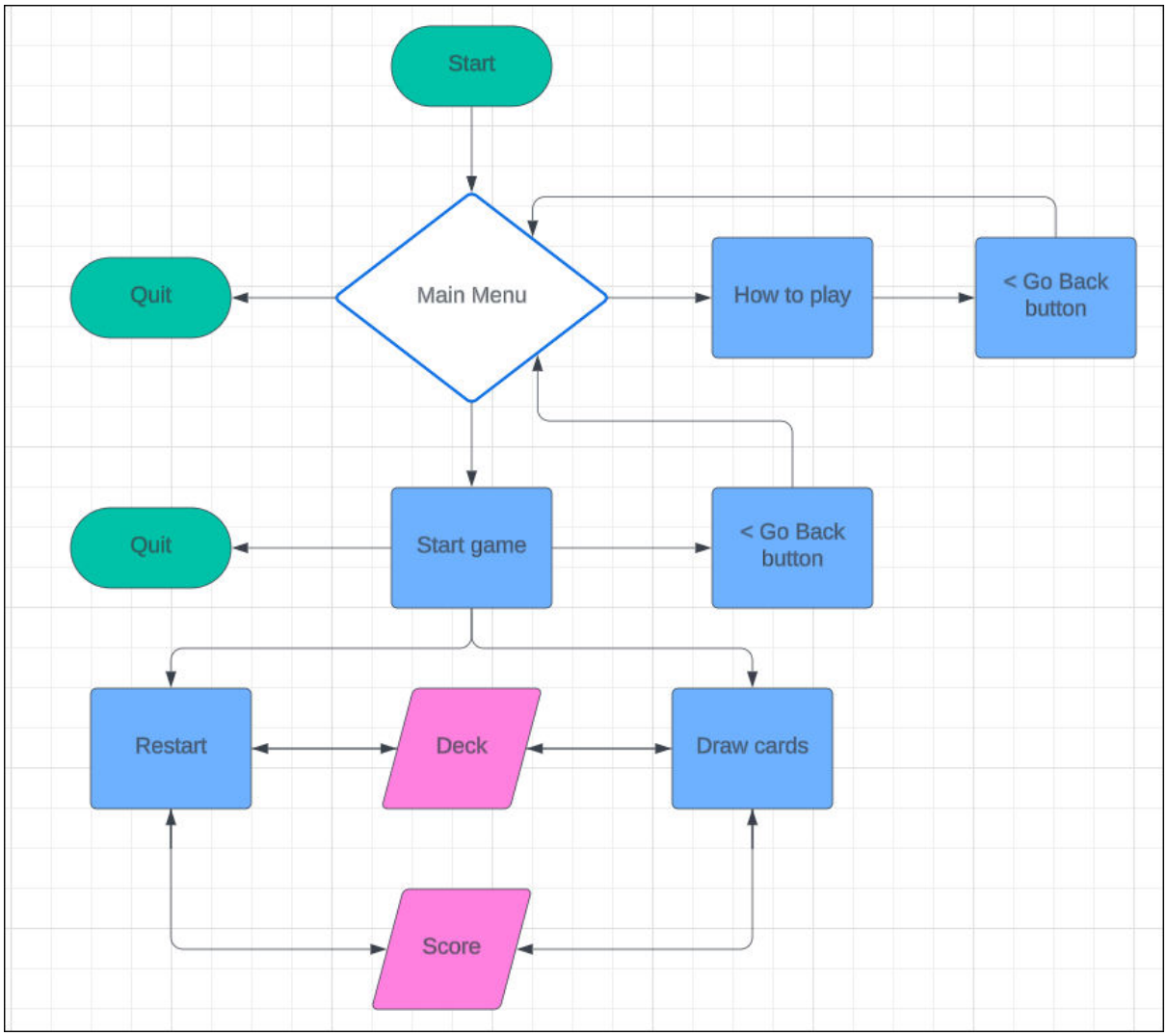


Fig 2 - lucidchart

Finite State Machines

State	Event	Next state	Action
menu	Click on start game button	game	Play the game
game	Click on < go back button	menu	Choose from the menu options
how_to_play	Click on <go back button	menu	Choose from the menu options
menu	Click on how to play button	How to play	Read the manual
menu	Click on Quit button	N/A	Exit the game
game	Click on Quit button	N/A	Exit the game

Class Diagrams

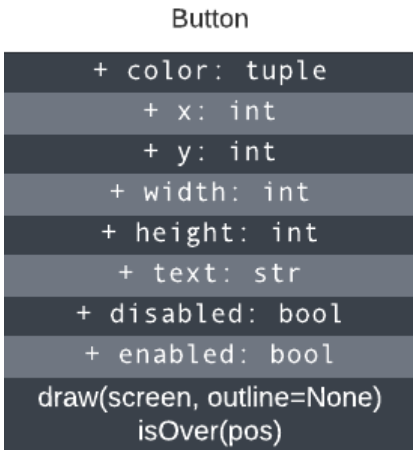


Fig 3 - lucidchart

Development

Programming Languages and IDE

For this project PyCharm will be used, as its fitting IDE for working with Python to build the game and offers several features like IntelliSense, AI Assistant, and Data Tools. Python is the matching high-level language which PyCharm handles pretty well and is also a good choice to build this game, because of its myriad of library choices, readability, and simplicity, it is also excellent for learning the basics of game development and creating small simple games. However, it lacks potential performance issues and possible limitations in bigger projects. Our project is small and is hopefully not going to be heavily impacted by those drawbacks.

Libraries

The only necessary libraries to import in order to implement this project, are going to be pygame, random, and os.

‘pygame’ Library is one of the most popular, open-source cross-platform library, commonly used for making games because of its features, simplicity and adaptability in building 2D games and applications.

‘random’ Library is a basic python module which can work with random numbers in many different ways.

And last but not the least, ‘os’ library, which provides a path for our program to interact with the operating system for the purpose of managing files and directories and many other features, but it will mainly be used to access the images of the cards for our project.

Version Control

Version control for this project is be carried out by saving the project files in GitHub, in case the project becomes too large to be uploaded in GitHub, Microsoft One-Drive can be used as a secondary option, as well as saving as much backups as possible on personal and college systems.

PyCharm is the IDE of choice for this program, and GitHub is better suited to Visual Studio, but we are using GitHub regardless.

Testing Strategy

Functionality and manual testing were undertaken to identify gameplay issues and bugs and ensure that the video game operates as intended.

Manual Testing		
Test	**Expected Result**	**Actual Result**
Background music test	Background Music working correctly	.mp3 files not supported
Background music test #2	Background Music looping continuously	Background music stops playing after the song finish playing once.
Sound Effects test	Sound Effects working correctly	Button.isOver(pos) mouseover error with the code
How to play button test	Instructions are presented nicely row by row and fit the screen	Instructions are only shown on one single row and can't see most of them because they go out of the screen
Start game button test	Players do not have any cards before ''draw'' button is pressed	Players already have random cards assigned, before even drawing
Inserting a new card 'Joker' into the deck	Joker is properly displayed and shuffled into the deck	Error: File name c01j not found
Resizing screen,card images & button text/position – hundreds of times.	Position of all the items and game screen size is exactly like I want it to be.	Everything is broken and out of proportion/going out of the screen.
Rapidly click the Draw button multiple times	Cards are drawn correctly without any errors or crashes	Cards are drawn correctly without any errors or crashes
Restart the game after the deck is empty.	The game reshuffles the deck, redeals the cards, and re-enables the Draw button correctly.	The game reshuffles the deck, redeals the cards, and re-enables the Draw button correctly.
Ensuring that cards have the correct values	cards have the correct value as specified in the game's rules.	cards have the correct value as specified in the game's rules.

Ensuring that the additional Joker card have the correct value	Joker cards have the correct value as specified in the game's rules.	Joker cards have the correct value as specified in the game's rules.
Win condition test - round	Player 1 or Player 2 wins the round after drawing a higher value card	Player 1 or Player 2 wins the round after drawing a higher value card
Win condition test - game	Player 1 or Player 2 wins the game after accumulating 28 points	Player 1 or Player 2 wins the game after accumulating 28 points
Win condition test – round tie	Players draw cards with the same value, and both players take a point accordingly	Players draw cards with the same value, and both players take a point accordingly
Win condition test – game tie	Both players have 27 points and then they draw the same card, then it’s a tie and nobody wins	Both players have 27 points and then they draw the same card, then it’s a tie and nobody wins

Assets

Images

Image	Description
Appendix 3	Main Menu background
Appendix 4	Start Game background
Appendix 5	How to play background
Appendix 6	Cards images examples

Sound

The main music theme for the game as seen in Appendix 1 was found in royalty free music section in YouTube without any copyrights, while searching for a casino style music. However, the little mouse click sounds for the different buttons Appendix 2, were found at the website storyblocks, while looking for ‘’cards’’ sounds. I was looking for a distinguished card sounds and casino-style music background music which fit well for my game.

Project Management

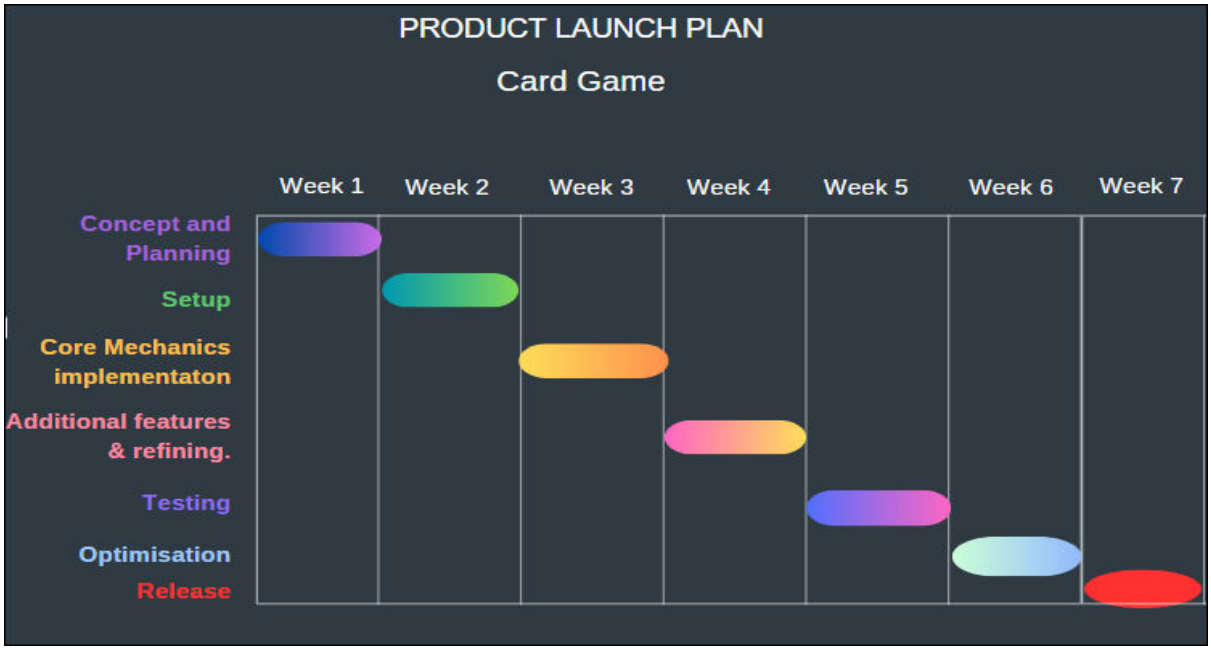


Fig 1 - canva

Timeline

First week of the project development plan will be devoted to concept and planning, which is basically coming up with game ideas, rules, mechanics, and features. Building a detailed design document outlining the game structure and user interface. Sketch game screens and planning of the visual style.

During the second week, the focus will shift to setting up and installing the necessary tools (PyCharm, Python, Pygame). Organising the project directory and creating placeholder assets. As soon as all of this is done, i can begin coding the basic framework (game loop, UI elements, etc).

Core Mechanics Implementation is one of the most important phases of our game development timeline and is dedicated for the third week of the project, during this week, I have to Implement essential game mechanics like, card shuffling, player turns, win conditions, and handle player input and interactions.

During the fourth week, 'Additional features & refining' phase have to take place, I will be trying to code and design player scores, sound effects, and animations, incorporating advanced gameplay features, as well as polishing the UI, graphics, and sound effects.

Throughout the fifth week I have to conduct internal testing to identify and fix remaining bugs in the gameplay mechanics, and fine-tune gameplay balance to ensure stability and an enjoyable experience.

Following is the 'Optimisation' phase in week six, optimising the code for better performance is the main focus in this point of our timeline, and profiling the game to identify areas for improvement is crucial. As well as finding ways for enhancing framerate and loading times. At the end of the Optimisation phase, a final round of testing is performed, and any remaining issues are being addressed.

The final and most underrated part of the timeline is the 'Release', final touches and deployment of the game are done during this time, as well as promotional materials (screenshots, trailers). Followed by packaging the game and finally releasing it.

Roles and Responsibilities

Designing the game requires a Game designer, which come up the idea for interesting game mechanics and beautiful interface designs, and craft a blueprint about the levels of difficulty and work together with the programmers which build the game through coding and bring the game to life, to provide a finished product. Testers are also a mandatory part of the process of designing a game, they play the game, provide feedback, and try different tactics and strategies that might bug or break the game in order to find bugs and issues.

I will be designing this game myself, so I will be the programmer, the game designer, and the tester of the game. I will be responsible to code the game mechanics and optimise the game, design interface elements, backgrounds, sounds, and come up with different ideas for the game, as well as test the game to ensure a finished product at the end without any bugs or issues.

References

French, J (2023) *A Guide to Choosing the Right Software Development Methodologies for Your Next Project.* Available at: How to write a game design document (with examples) - Game Dev Beginner (Accessed on 16th of May 2024).

studytonight (2024) *A Guide to Choosing the Right Software Development Methodologies for Your Next Project.* Available at: Technical Design Document and Game Design Document | Studytonight (Accessed on 17th of May 2024).

Appendices

Fig 1

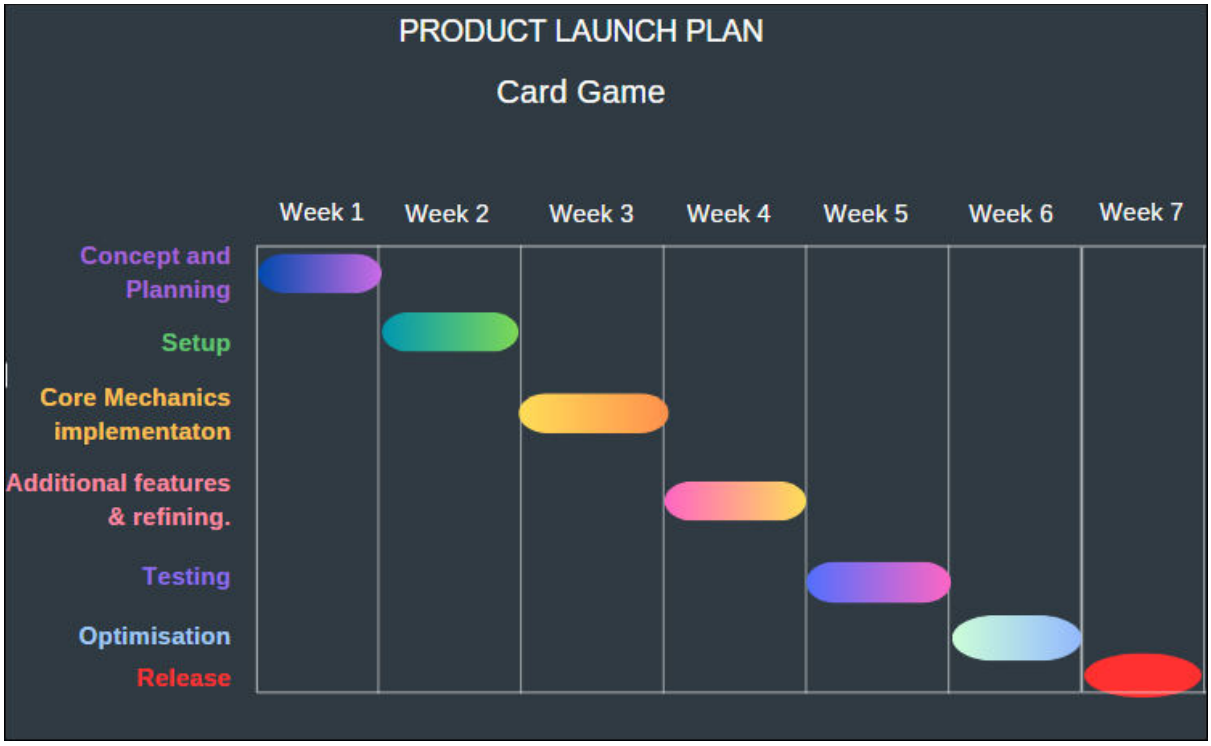


Fig 2

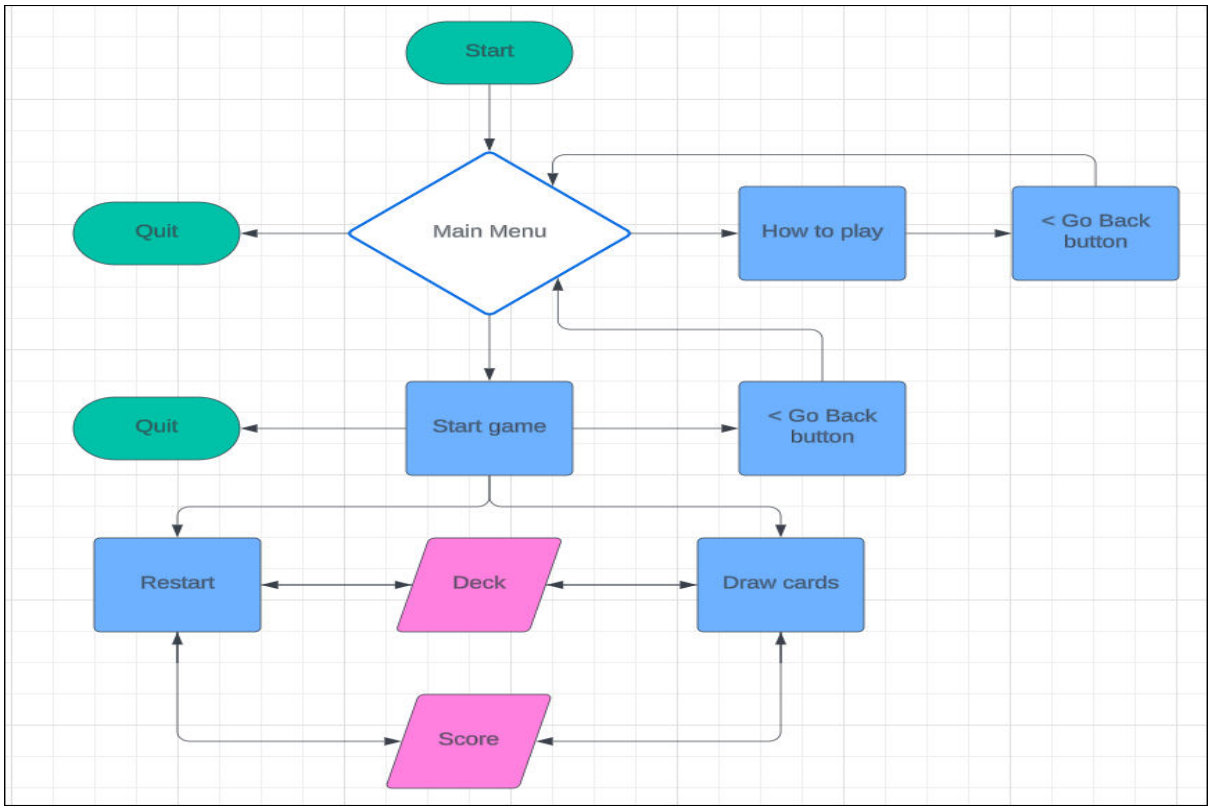
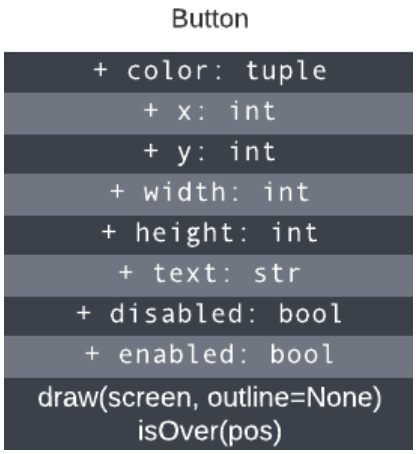


Fig 3



Appendix 1

casino_music.wav

https://www.youtube.com/watch?v=X_kvEZSDehQ

Appendix 2

select.wav

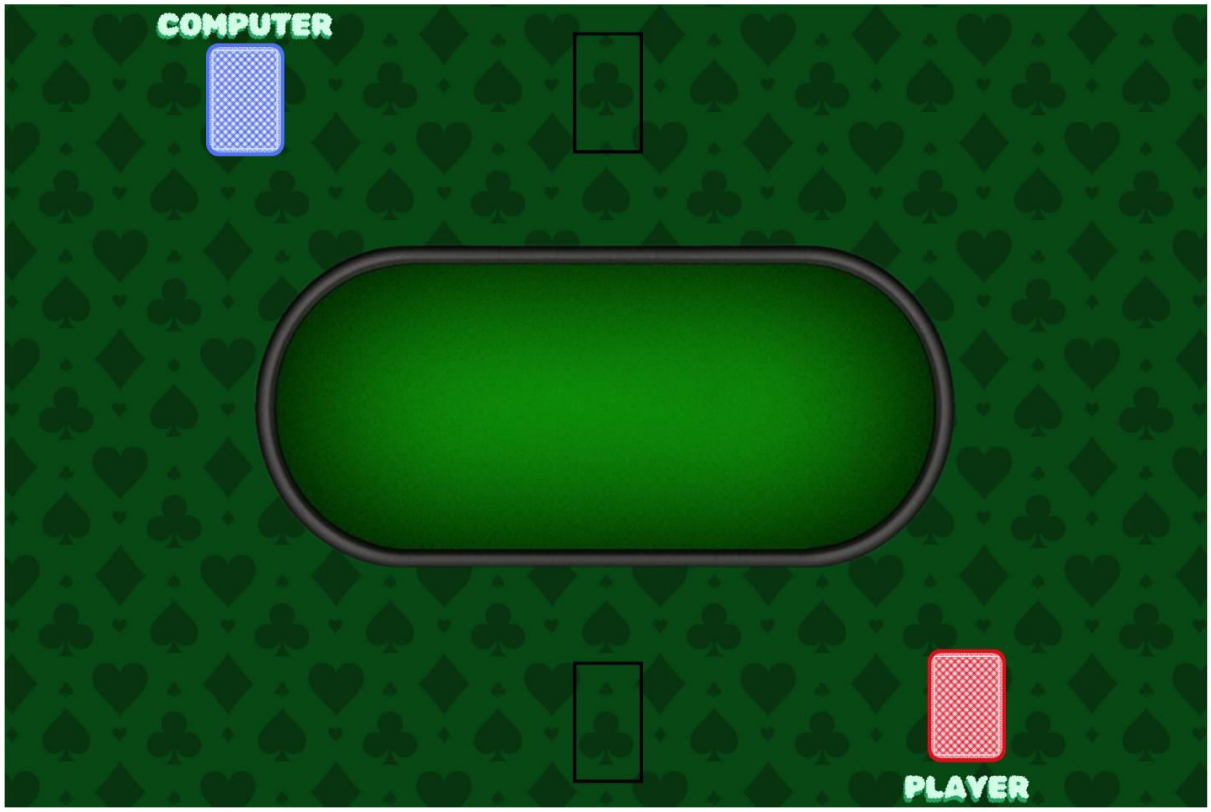
restart.wav

draw.wav

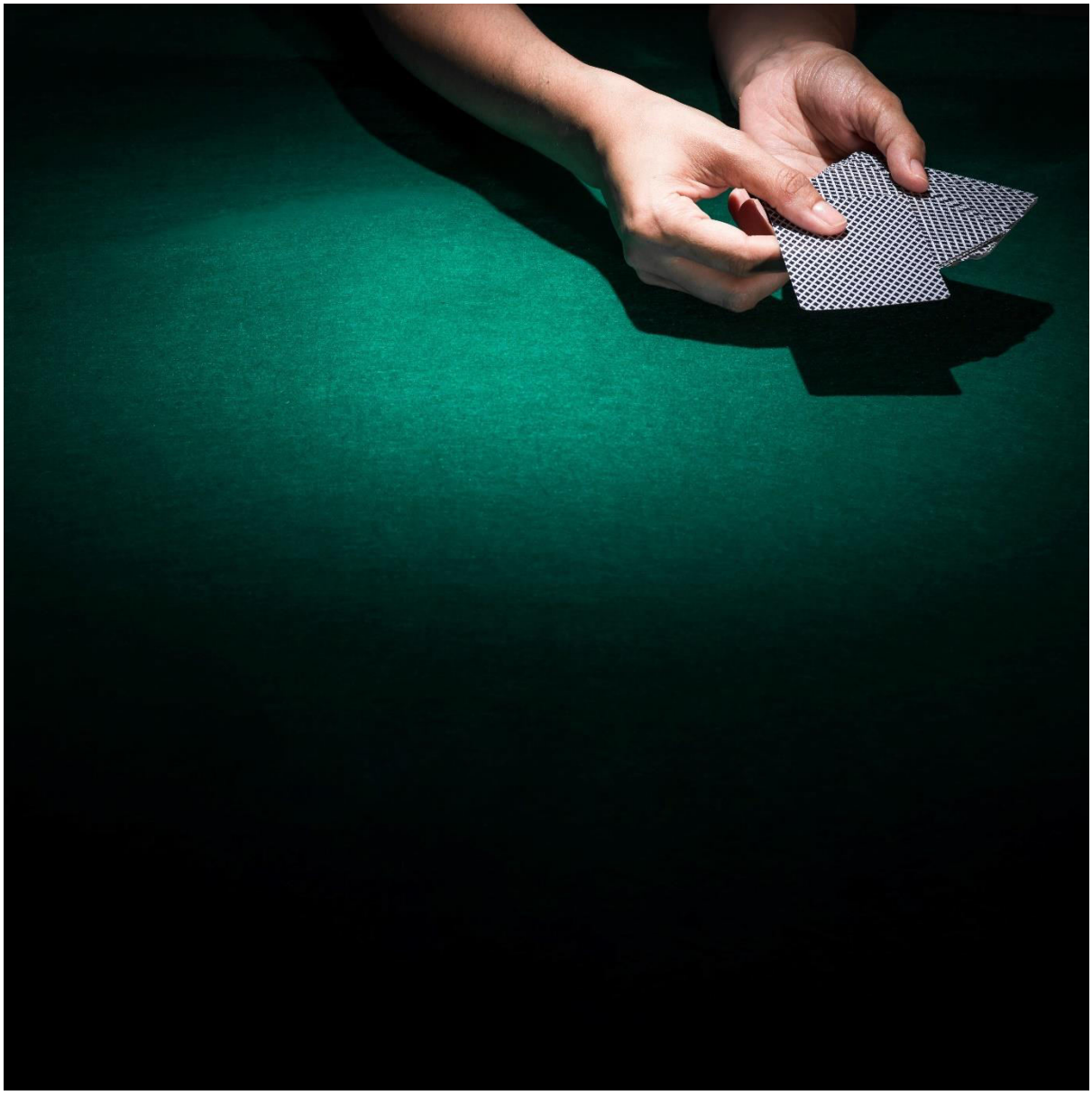
https://www.storyblocks.com/audio/search/card-sound-effect

Appendix 3

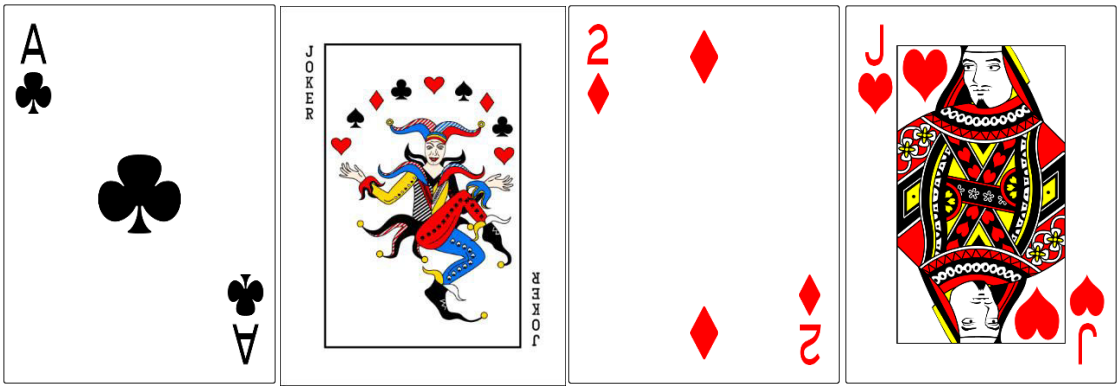
Appendix 4



Appendix 5

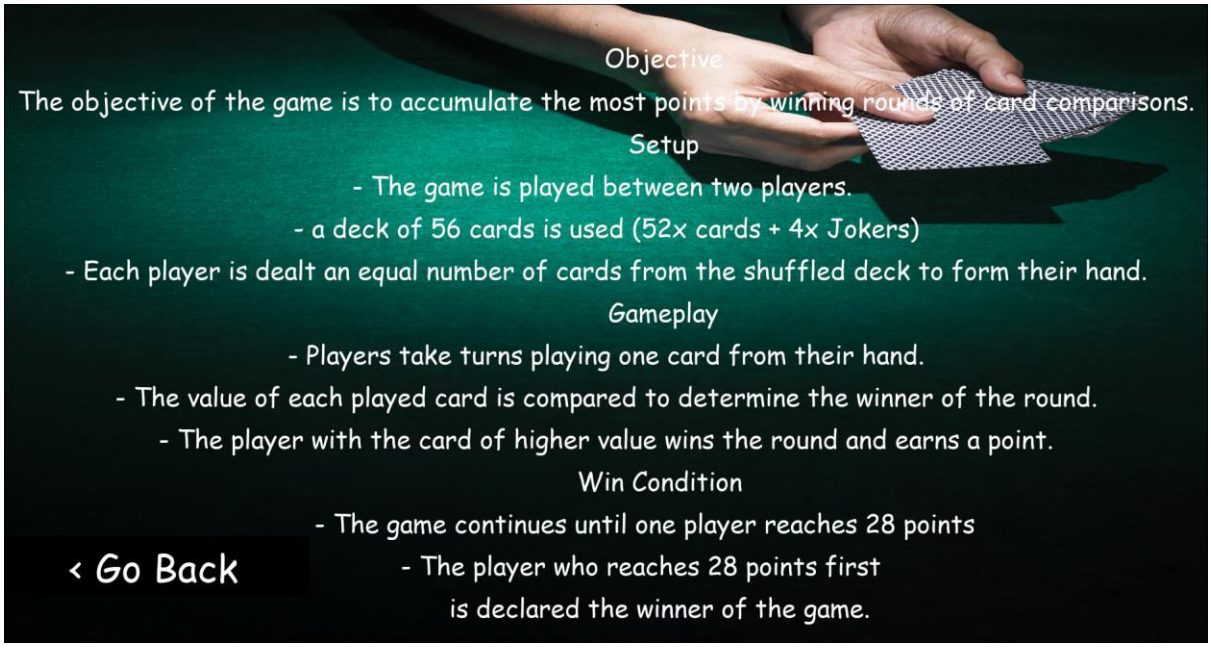


Appendix 6



Appendix 7

Appendix 8



Appendix 9