Algorithm and Programming | Project Report

By Barri Nur Pratama

Project Specifications

General Description of the project

Title : Match the Cards

Type : Game

Description : Endless card matching that is both fun and stressful, while improving memorization and reflexes.

Target audience

The game has a wide range of target audience due to its simple mechanics of just using the cursor and mouse clicks, trying to remember and match the cards shown. Though the game features some flashing jump scares and loud noises which can be too much for some people, therefore the target is above the age of 13.

Gameplay Mechanics and Features

The game is entirely based on cursor movements. Therefore, no keyboard input, and the only required device besides a PC is a mouse or mousepad (for laptop users).

To play, click the start game button, then the game will display some cards for you to remember, then the cards will flip and the game will display a countdown timer (60 seconds), and you need to find matching cards. If the cards match it will stay faced-up, else it will flip back.

Do this repeatedly until all the cards are matching and faced up, then the game will start another round, with the time continuing from your last round.

Win and lose mechanics:

- You cannot win the game; it will loop forever until you lose

- When the countdown reaches zero you lose the game, then you will be redirected to the game menu, and you can click start game again or quit the game.

Random bonus time and time penalty system:

- The game can give you random bonus time if you matched the cards *3 times in a row**, and *6 times in a row**

- The game can give you random time penalty if you fail in matching the cards *3 times in a row**

Barri Nur Pratama – 2802501142

Difficulty progression:

- After 10 rounds won, the random time penalty happens when you fail matching the cards 2 times in a row.

- After 30 rounds won, the random bonus time happens only if you matched the cards 6 times in a row, and it will give you a time penalty when you fail to match once.

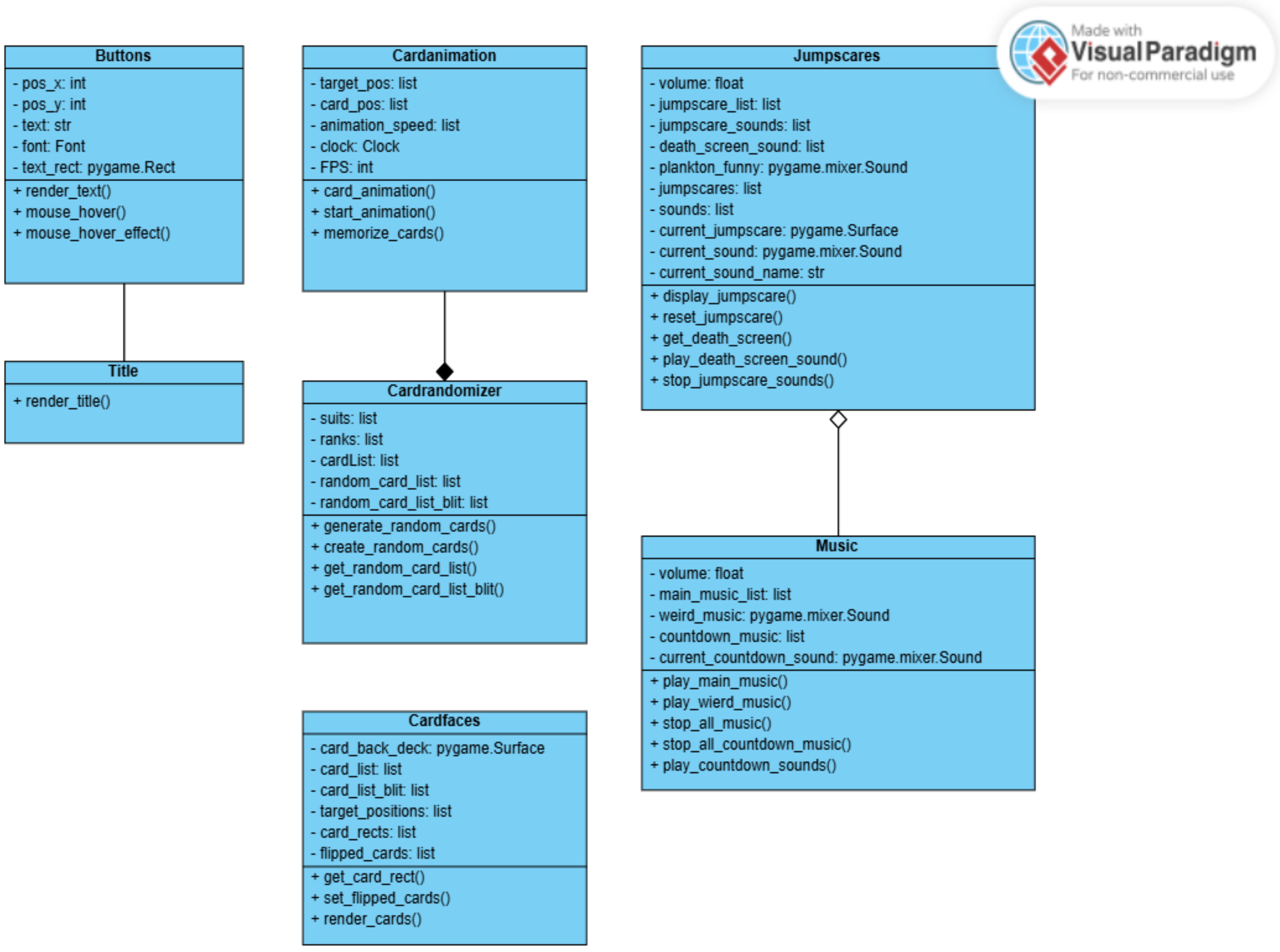
Dependencies

- Python 3: The main programming language used for implementing the algorithms
- PyGame: Python module that helps in the creation of simple 2D games using the python language
- PyInstaller: To create an executable (.EXE) file for the main.py file, so that users can simply run it, without the need to download any code editor

Solution design

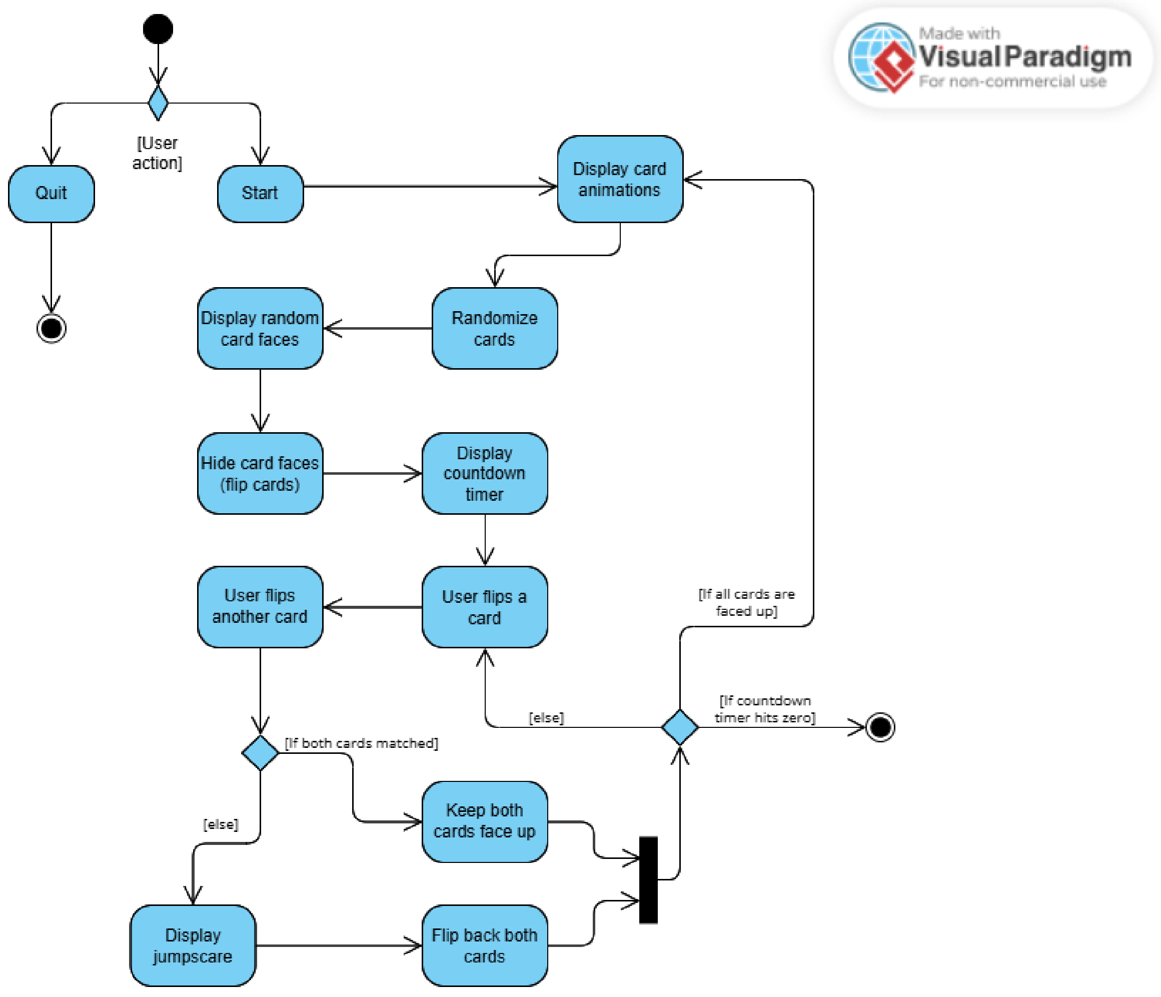
For creating a solution for the algorithm, first we must create a design of what and how our application will function, one of the best ways is the making of diagrams. Shown below are two diagrams that helps in understanding the bigger picture of the design.

Class Diagram



Barri Nur Pratama – 2802501142

Activity Diagram



Algorithm Implementation

Game menu

The game menu initializes the game by creating a game loop that has the menu display, which consist of: Title, start and quit buttons, background image. The game menu also has an event handler for three conditions:

- Checking if the user clicks the QUIT button in the pygame window (not the game itself), then it quits the game (this is important so that a game does not run forever).
- Checking if the user clicks the “Quit” button in the program itself, then it quits the game, by using pygame.quit() and sys.exit().
- Checks if the user clicks the “Start” button, then it calls the start_main_game() function.

Main game

The main game is when start_main_game() function is called. A new game loop is then created which consist of most of the logic needed to manage the various components of the program.

Barri Nur Pratama – 2802501142

Event Handler

The event handler of the main game section has the algorithm for:

- Checking if the user clicks the close window (Standard for every game loop)
- Checking for each left mouse button click, for each card rects, it checks whether it collides with the rects, and flips the card, then:
    - If the two cards are different, it flips them back (explained in the next section), and tracks the fail attempts.
    - If the two cards are the same, it stays faced up, and tracks the success attempts.

Logic to turn the card back

When the cards does not matched (known from the event handler), then it will call the function from the Cardfaces class that turns the card back, by setting the arguments in the function set_flipped_card(), to the card indexes, and False.

Game difficulty algorithm

Checking the current rounds, and manage the difficulty level accordingly. This is done by using a list of three difficulty variable which contains a list of integers.

The first and second integer is on how much successive attempts needed to get a bonus time, meanwhile the third integer is on how much failed attempts needed to get a time penalty, and jumpscare.

Time management, bonus time algorithm

Consist logic that displays the countdown timer by blitting, the seconds left into the screen. The second's left is taken by getting the start ticks before the game loops start, and use this for a decrement of main_countdown_time (which at first starts at 60).

This section also consists of logic on changing the fonts, for a pulsing effect, when the countdown is getting small.

The bonus time and time penalty algorithm uses the track_success_attempts variable which is managed by the event handler, then it compares it with the current difficulty level, then it gives a certain random value for bonus time.

Jumpscare algorithms

Consists of two types of jumpscares, according to the sound that is played.

- The first type of jumpscare blits randomize images and updates the display in the game loop itself, therefore it will flicker (fun fact this was not intentional at first).
- The seconds type is when the image only appear once but with a fadeout effect, that changes the alpha values of the picture.

When any type of jumpscare is encountered, it will create a certain random value for decrementing the main_countdown_time, like a time penalty.

Logic on countdown music

Simple conditions for playing the countdown music, by referring to the second's left on the countdown. For example if the second's left is equal to or below 25, then it will play 25_seconds_countdown music.

Player wins

For each game loop, the program checks if all the cards are faced up, if so, then it will set the main_countdown_time as the second's left in the previous round (if the second's left in the previous round is smaller than or equal to 10, then the main_countdown_time will be a random value between 8 - 12).

Then it will call the start_main_game() function again, basically restarting the main game (but does not go to the game menu).

Player loses

For each game loop, the program checks if the seconds left hits zero. When it does (and all the cards are not flipped), it will display a death screen, which is basically the same as a randomized jumpscare, but with a single unique sound.

Then it resets everything, like jumpscares and music, then it calls the game_menu() function. To go back to the main menu.

Classes

The classes consist of attributes and operations that are called accordingly by the main game logic. Below are all the classes present in the program:

Title and Buttons

The title class consist of a single operation to render the title, while all of its attributes is inherited from the Buttons class

The Buttons class has logic on rendering the text, while also creating a rectangle cantered on the text, inflate it to add padding, and adds a border radius to make it look like a button.

It also has a collide point checker that returns a Boolean value, this is so that the event handler can create a condition based on this. Lastly, it has another operation to change the color of the text when the mouse is hovering the button.

Cardanimation

Handles movements of cards by referring to the target positions, that is initialized in the main.py file.

- The card_animation() function adds a certain value to the card position, according to the animation speed, and the delta values of the card positions (target position – start position).
- The start_animation() function iterates through each card, then creates a game loop to animate each card by calling the card_animation() function which is also in the same class.
- The memorize_card() function blits all the cards accordingly, by referring to the Cardrandomizer class, then it flips them back after a certain memorizing time (2 seconds).

Cardrandomizer

This class initialize the cardlist which has the suits, ranks, and some other easter egg cards. It then shuffles them.

- The function generate_random_cards() basically takes 6 elements from the card list attribute. Then it duplicates it, and shuffles it.
- The function create_random_cards() calls the generate_random_cards() function to create the list of randomized cards. Then it loops to create surfaces for each of these cards, and append them to the random_card_list_blit attribute.
- The two getter functions for random_card_list and random_card_list_blit, simply returns the attribute.

Barri Nur Pratama – 2802501142

Cardfaces

This class manages the algorithm to flip the cards, and render the face of the cards.

- The function get_card_rect() creates a list for the blitting of all the card.
- The function set_flipped_cards() changes the boolean value of the attribute flipped_cards for a certain index. This is how to track the whether the cards are flipped or not, by using a list of boolean values that are the length of the card list itself.
- The function render_cards() consist of a condition if a certain card has the boolean value of True, then it will blit the card's face, else it will blit the card's back.

Jumpscares

This class primarily function for the purpose of blitting jumpscare images and playing jumpscare sounds. First it initializes these images and sounds, mostly into a list.

- The function display_jumpscare() selects a random jumpscare from the list and display it on the screen.
- The function reset_jumpscare() resets all the flags in the algorithm, and also stop the current jumpscare sound.
- The stop_jumpscare_sounds() function simply stops all the sounds by iterating through each one of them, and the additional sound that is not in the list.
- While the rest of the functions are getters, that returns a randomize death_screen to then be displayed, and playing the death screen sound.

Music

The music class manages the background music and the countdown music, which is when the countdown reaches a certain value, it will play a unique music.

- The function play_main_music() selects a music from the background music list then plays it.
- The function play_wierd_music() simply plays the weird music (referring from the attribute). The weird music is played as soon as the player gets the first jumpscare.
- The function play_countdown_sounds(), at first it calls the stop_all_music() function to get rid of all the background music, then it plays the countdown music according to the argument given to the function, then it updates the current_countdown_sound attribute.
- The function stop_all_music() simply stops the weird music, and all the background music by using a loop.
- The function stop_all_countdown_music() stops only the countdown music by using a loop.

Data structures used

Lists

The use of list in this program is the most widely used type of data structure such as in:

- random_card_list
- random_card_list_blit
- jumpscare_list
- jumpscare_sounds
- main_music_list
- countdown_music
- difficulty data
- etc.

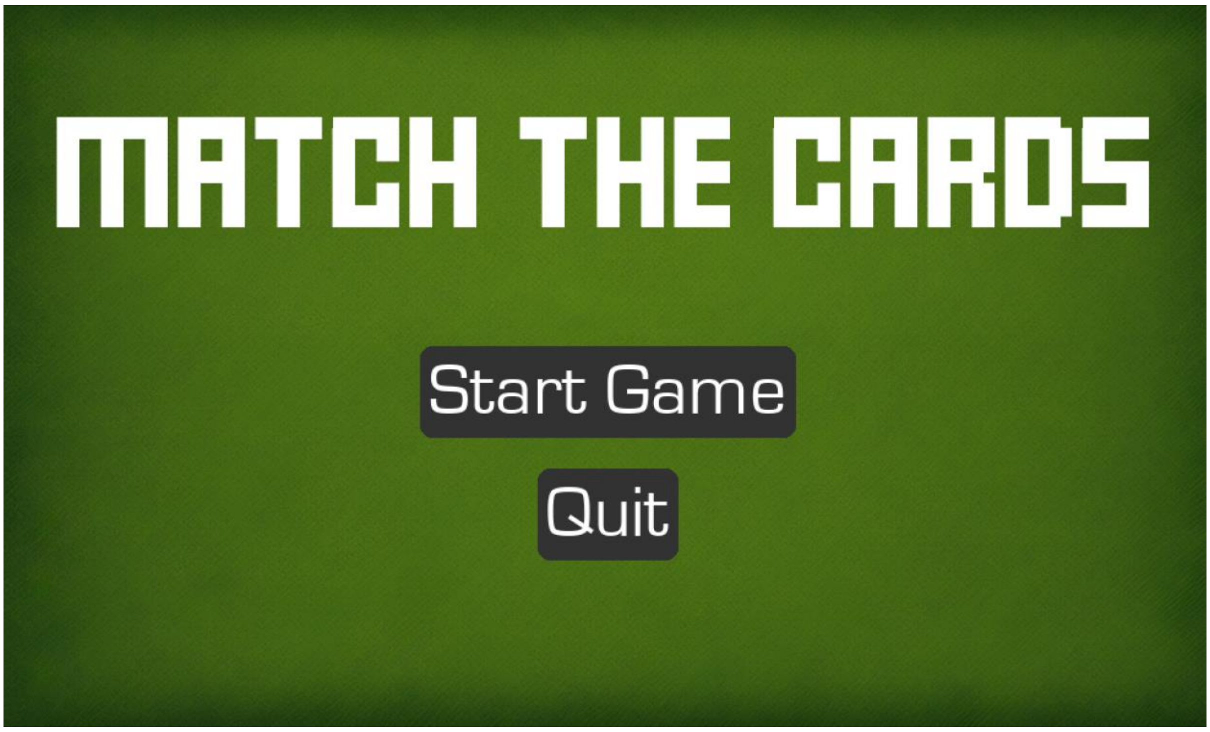
Barri Nur Pratama – 2802501142

Tuples

Tuples are used for certain pygame commands that requires it, such as the target positions (which consist of a list of tuples), to know where the card should be displayed when the card animation is finished.

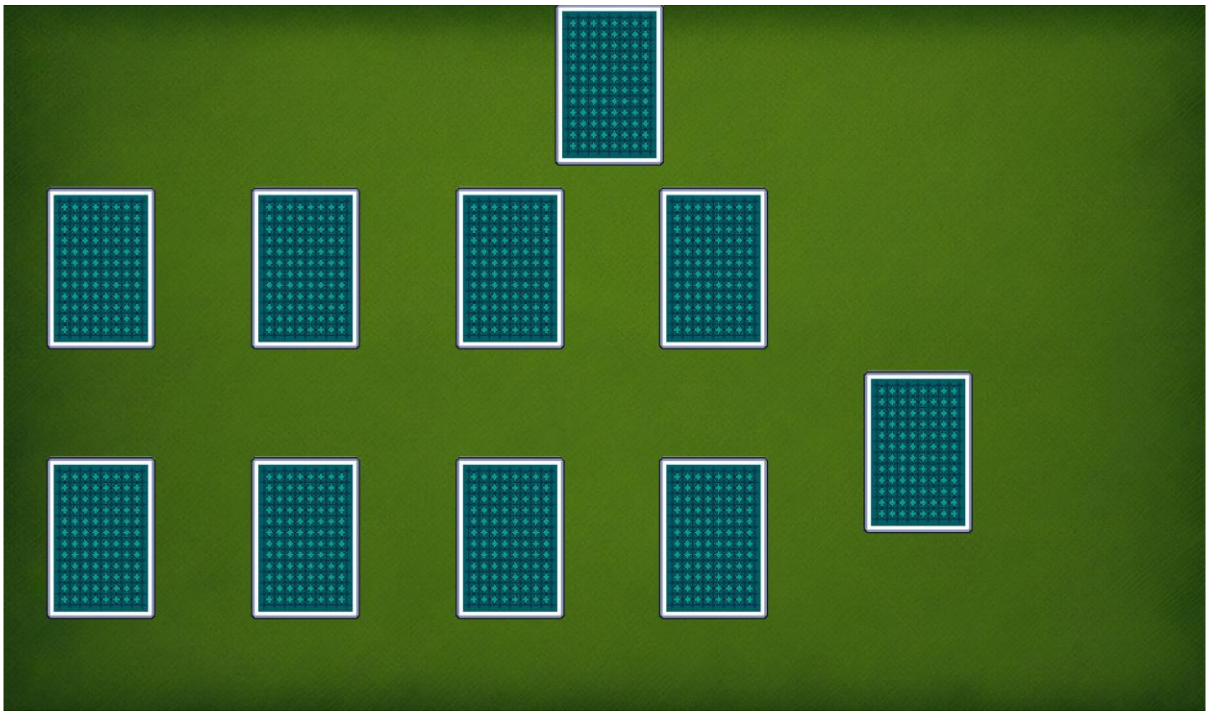
Evidence of Working Gameplay

This first picture is how the game looks when we run the exe file, or run the main.py file using a code editor. The program will display a game menu that has two buttons.

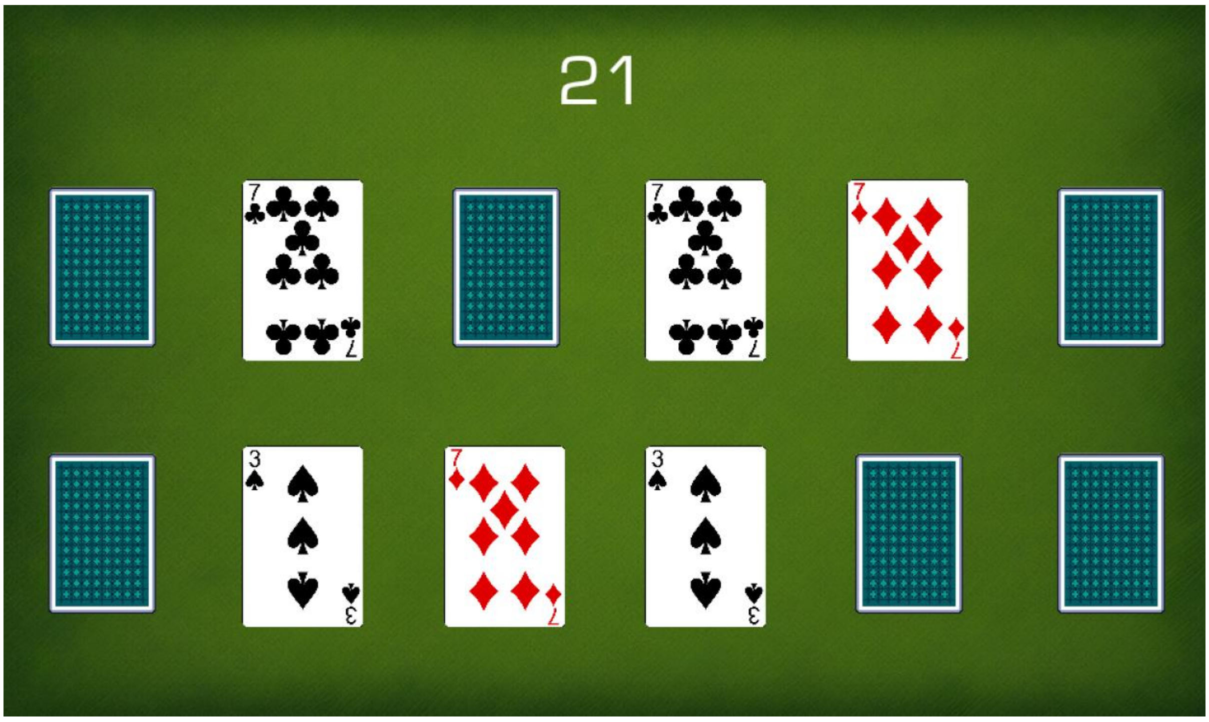


Barri Nur Pratama – 2802501142

If the user clicks the start button it will then initialize the card animation, as we can see in the picture below, where the cards are being placed on their respective target positions, and there is one card still on it’s way to that target position, some are already there, and some are still on their starting position.

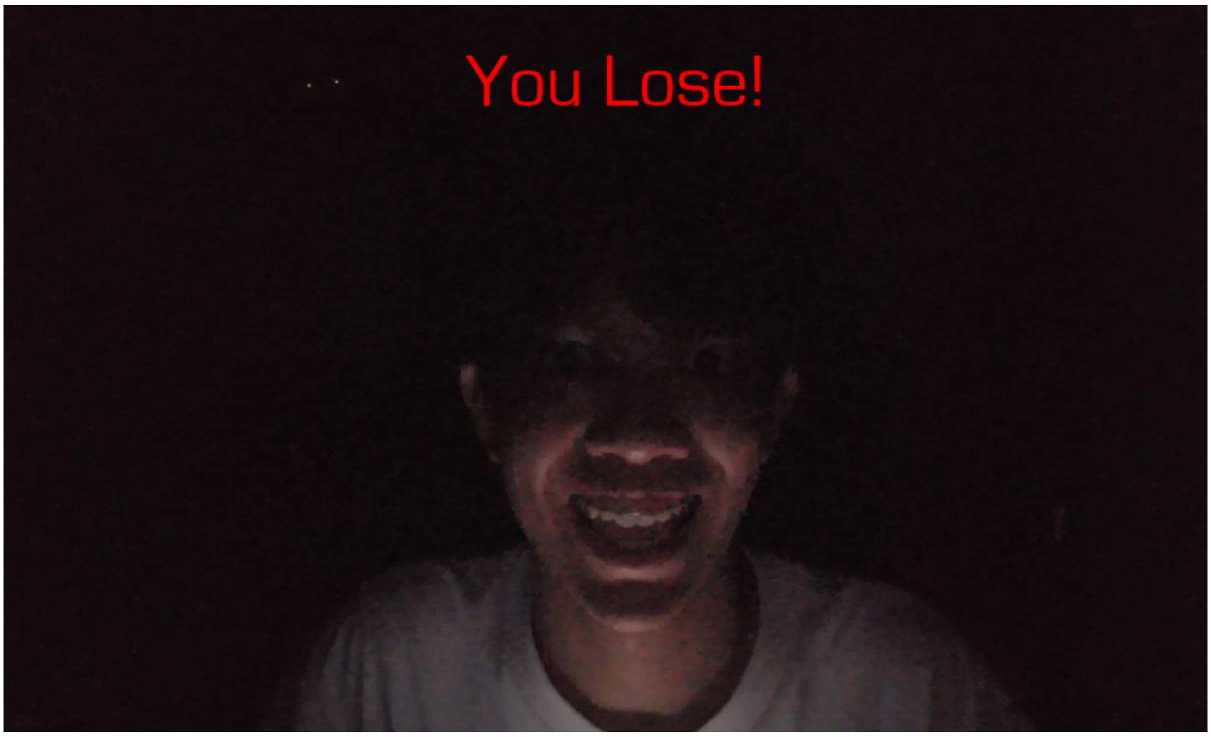


Then the main game loop will start, and the user can interact with the cards, while the countdown timer is ticking in the background.



Barri Nur Pratama – 2802501142

Lastly, this is what happens when the countdown timer hits zero, or in other words, you lose the game.



References

- Kahoot soundtrack for the countdown music taken from Youtube: https://www.youtube.com/watch?v=-O3z9xRM5Rc&list=PLsa7dvIdIrDll1frTUiwSVZri51TF7-ax
- SFX 34. (2021, June 13). Who invited this kid Mp3 (TikTok) [Video]. YouTube. https://www.youtube.com/watch?v=RfQyq851SaE
- Discobre. (2024, October 6). Let It Snow (Brainrot Edition) “Ohio, Ohio, Ohio” AI Parody cover (lyrics by notjewboi) [Video]. YouTube. https://www.youtube.com/watch?v=h3vbvdesee8
- RAHILU. (2024, November 5). Thick of It by KSI but it’s Big Band Jazz (Full Version) [Video]. YouTube. https://www.youtube.com/watch?v=vKoC8bNQrFA
- Know Your TikTok. (2024, December 9). The viral cursed Plankton moaning meme explained [Video]. YouTube. https://www.youtube.com/watch?v=VvHkYjZZP3M

NOTE: The rest of the music and images used are royalty free

Special thanks to my friends: Michael, Jason, Attalah, and Salomo for allowing me to use their funny pictures :)