Assignment Cover Letter

(Individual Work)

Student Information:	**Surname**	**Given Names**	**Student ID Number**
1.	**Arthur**	**Nicholas**	**2301890301**

Course Code	**: COMP6502**	**Course Name**	**: Introduction to Programming**
Class	**: L1AC**	**Name of Lecturer(s)**	**: Ida Bagus Kerthyayana**
Major	**: CS**		
Title of Assignment (if any)	: Blackjack		
Type of Assignment	**: Final Project**		
Submission Pattern			
Due Date	**: 14-01-20**	**Submission Date**	**: 14-01-20**

The assignment should meet the below requirements.

1. Assignment (hard copy) is required to be submitted on clean paper, and (soft copy) as per lecturer's instructions.
2. Soft copy assignment also requires the signed (hardcopy) submission of this form, which automatically validates the softcopy submission.

"Blackjack"

Name: Nicholas Arthur

ID: 2301890301

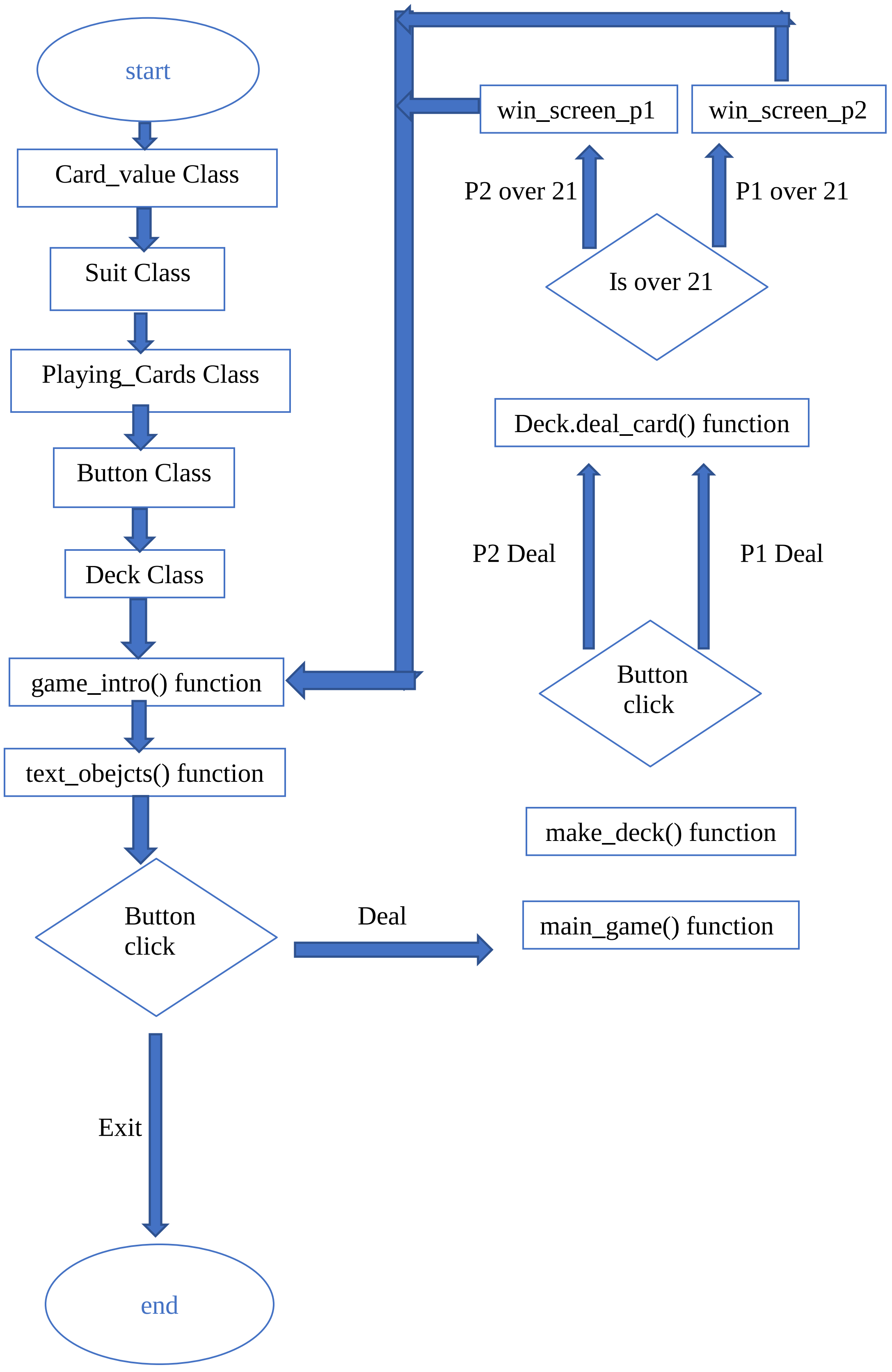
I. Specifications

The function of this program:

The function of this program is to play a simple game of blackjack. The initial deck is created using a function that takes a card value and pairs it up with a suit, this process is then repeated until a full deck is created. Two cards are then distributed using a class method and the cards are displayed using the Pygame library. The players will then deal themselves cards until the other one reaches above 21, the one that doesn't reach above 21 or has the lower score will lose.

II.a Solution Design

Project's Hierarchy Chart



II.b Explanation of Each Function Inside and Outside Classes

Project_Final.py

- Def make_deck():
    - Creating a deck with 52 cards that are unique according to their card value and suit.
    - The card creation is looped 52 times with 4 suits and 13 different card values.
- Def text_objects(text, font):
    - Defining a text to be rendered later by a different functions used later in the code.
    - The dunction return a format for the text to be displayed and draws a rectangle for the text to be displayed in.
- Def message_display(text):
    - Renders a text with the font size of 115 and with the font of impact.
    - The text is rendered at the center of the display screen.
    - The text lasts for 2 seconds.
- Def win_screen_p1():
    - Displays a message "Player 1 Wins"
    - The lists of cards inside the hands of player 1 and 2 are deleted.
- Def win_screen_p2():
    - Displays a message "Player 2 Wins"
    - The list of cards inside the hands of player 1 and 2 are deleted.
- Def tie_screen():
    - Displays a message "Tie"
    - The list of cards inside the hands of player 1 and 2 are deleted
- Def game_intro():
    - Renders the background image of the game
    - Displays a message blackjack in the center of the screen
    - Render buttons in the screen that says "Deal" and "Exit"
    - When the "Deal" button is clicked, it calls the function main_game()
    - When the "Exit" button is clicked, it exits the game.

- Def main_game():
  - Renders the background image at the back of the game.
  - Counts the total amount of card value in each player’s cards in the beginning of the game.
  - Renders the card image of each card in the initial player’s hand
  - Render buttons in the screen that says “Hit P1” and “Hit P2”
  - When “Hit P1” is clicked, it appends a new card to the hand of player 1 and adds the card value of that card into the total value of cards in the player 1’s hand, the card is then also rendered next to the cards of player 1.
  - When “Hit P2” is clicked, it appends a new card to the hand of player 2 and adds the card value of that card into the total value of cards in the player 2’s hand, the card is then also rendered next to the cards of player 2.
  - The function checks if any of the player has exceeded a total value of 21 or if both the players have a total of 21.
  - If player 1 has exceeded 21 the function win_screen_p2() is called.
  - If player 2 has exceeded 21 the function win_screen_p1() is called.
  - If both players have a total of 21 the function tie_screen() is called.

button.py

- Def make_button(self):
  - Makes a button with the set size of the button, the set position of the button, and the set text of the button.
  - The size, position and text of the button is set when and object is assigned to be a button.
- Def text_objects(text, font):
  - Defining a text to be rendered later by a different functions used later in the code.
  - The dunction return a format for the text to be displayed and draws a rectangle for the text to be displayed in.

deck.py

- Def deal_card(self, slot):

    - Appends a random card in the deck into a list and deletes the card from the deck.
- Def initial_deal(self, playercards):
    - Repeats the deal_card() function twice, this function is called when a game starts.

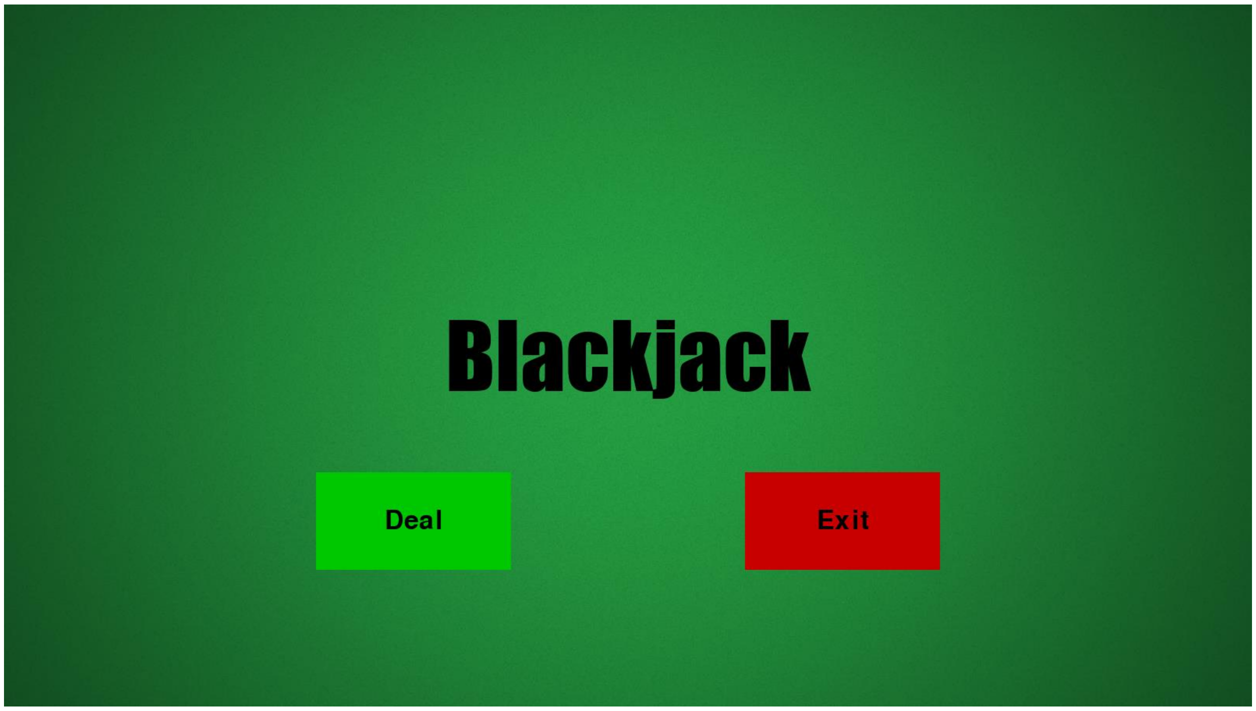
Class Diagram

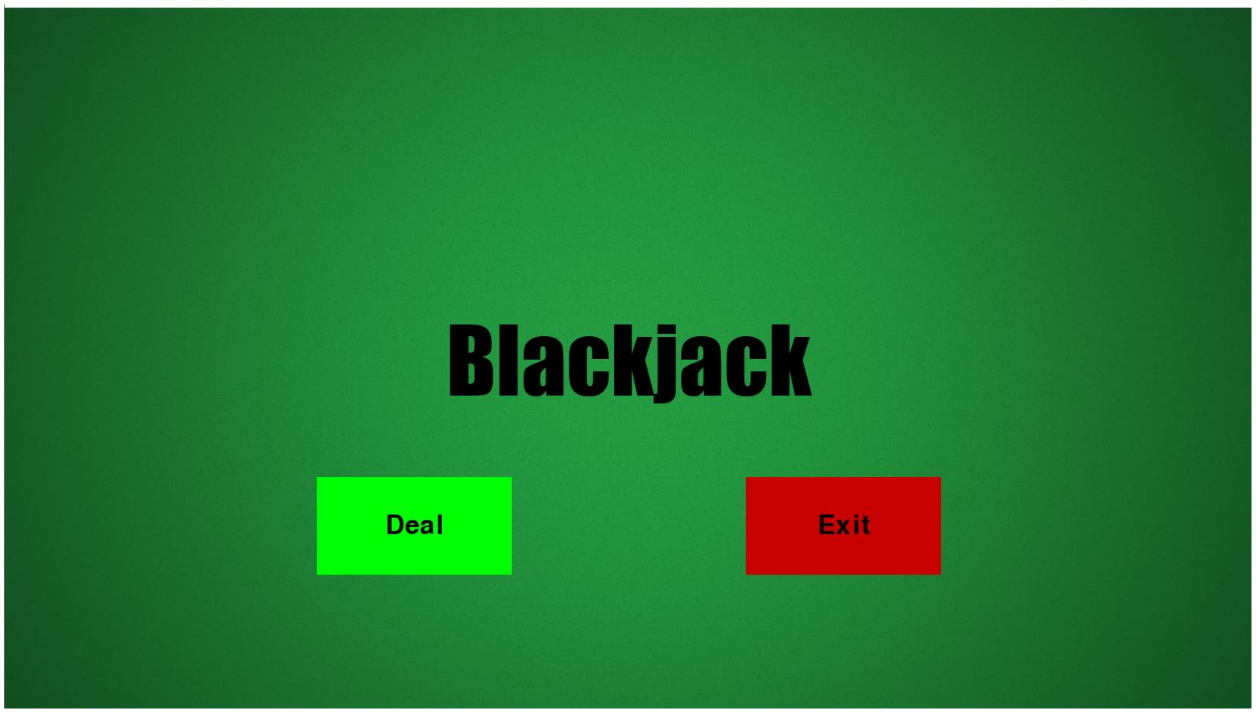
Button
-msg **-x** **-y** **-w** **-h** **-inactive** **-active** **-action** **-mouse** **-click**
-make_button(self)

Deck
-deck
-deal_card(self, slot) **-initial_deal(self, playercards)**

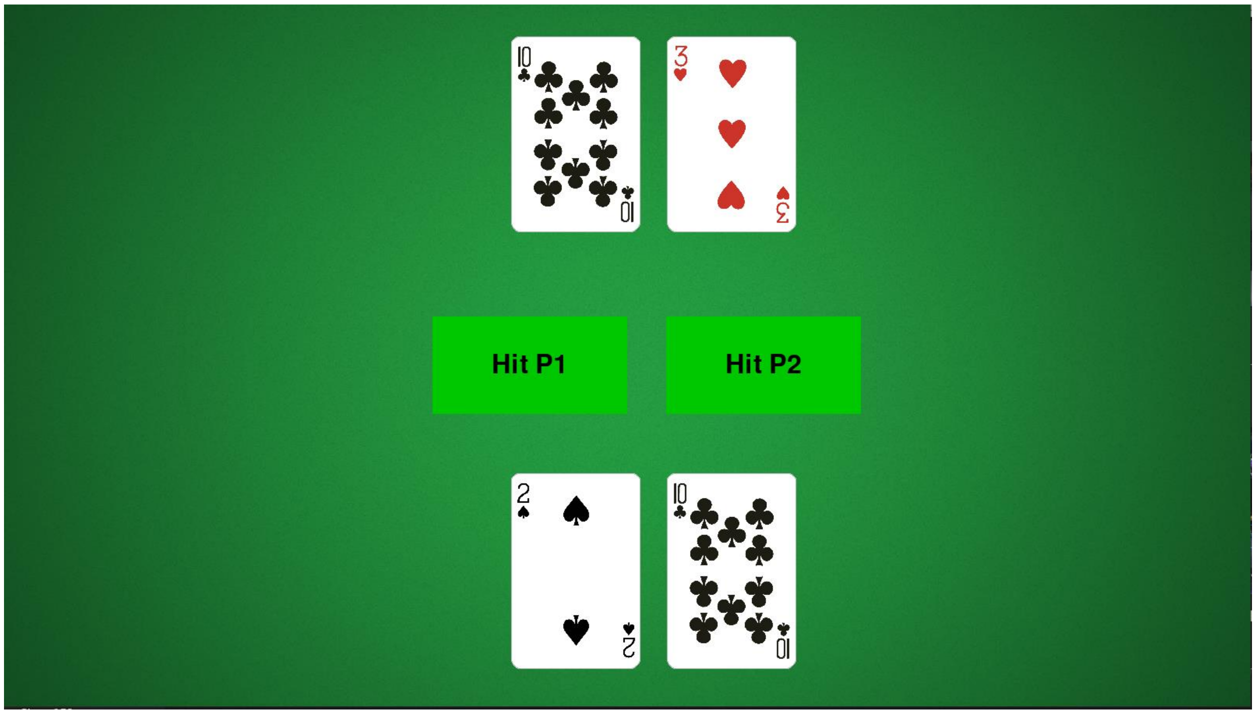
III. In depth explanation

When the code runs, it first calls the function game_intro() this function will draw the title of the game and render 2 buttons that has the text “Deal” and “Exit”. When these buttons are hovered over by a mouse, they will change color into a brighter shade of green and red respectively.

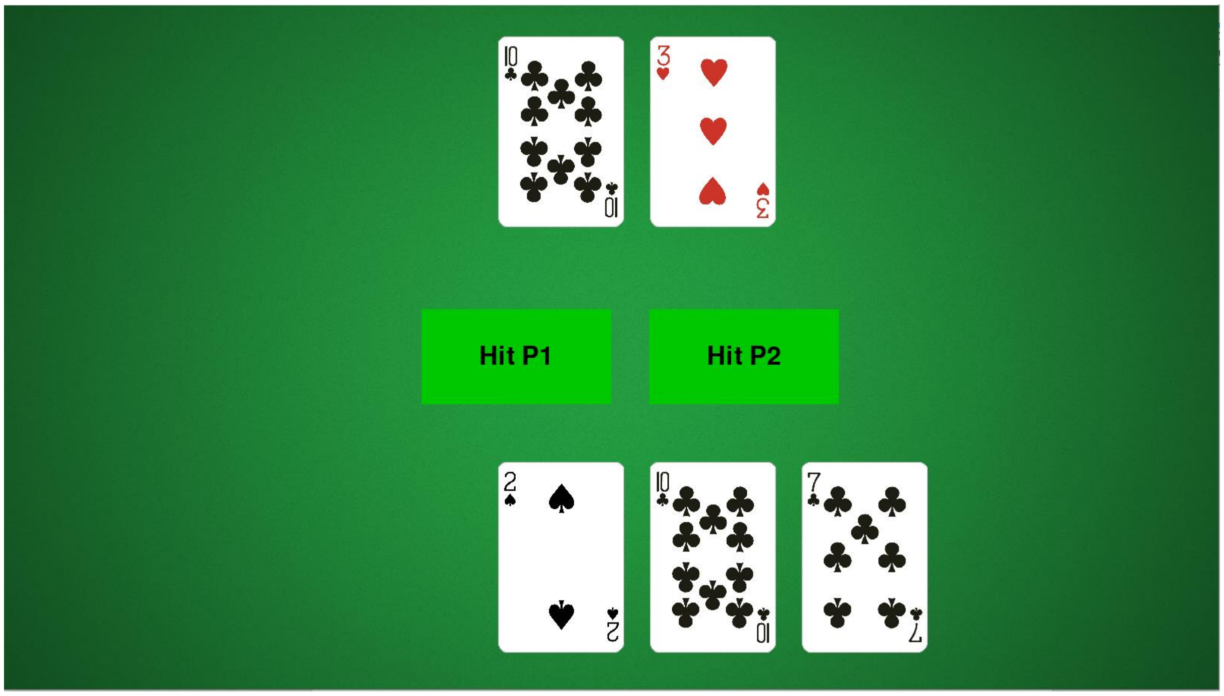




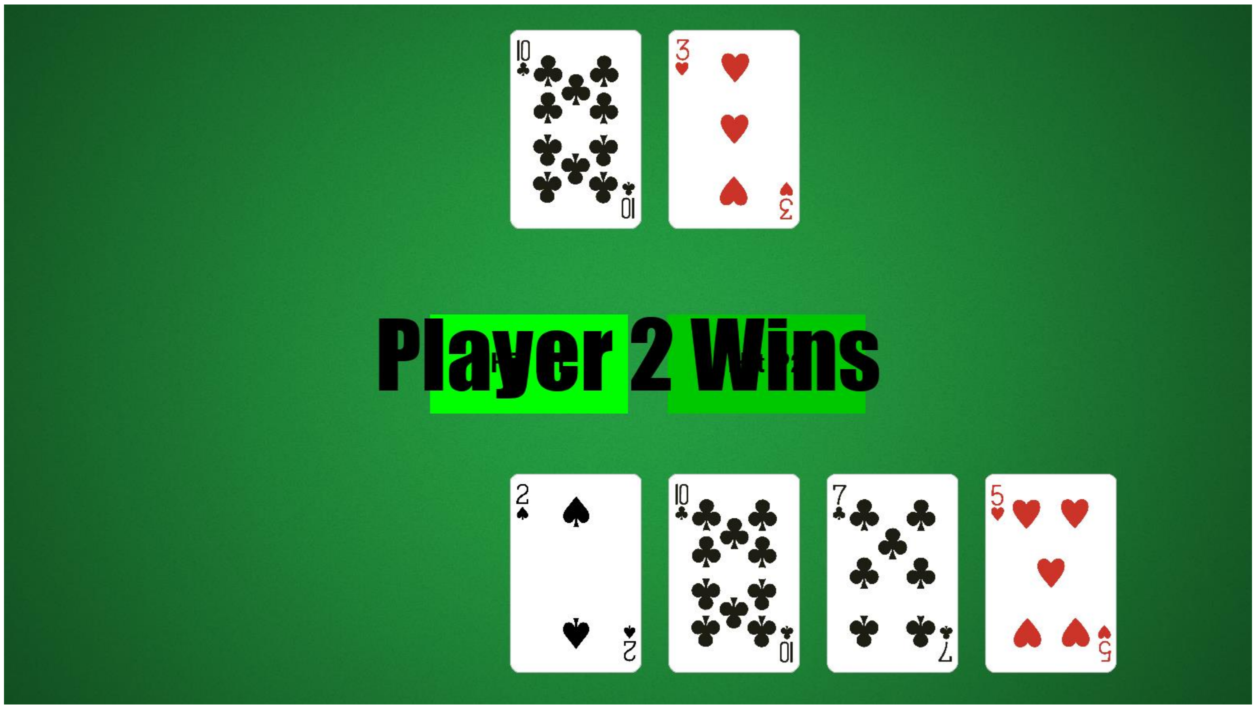
When the Deal button is clicked, it will call the function main_game() and when the exit button is clicked it will exit the game using quit().



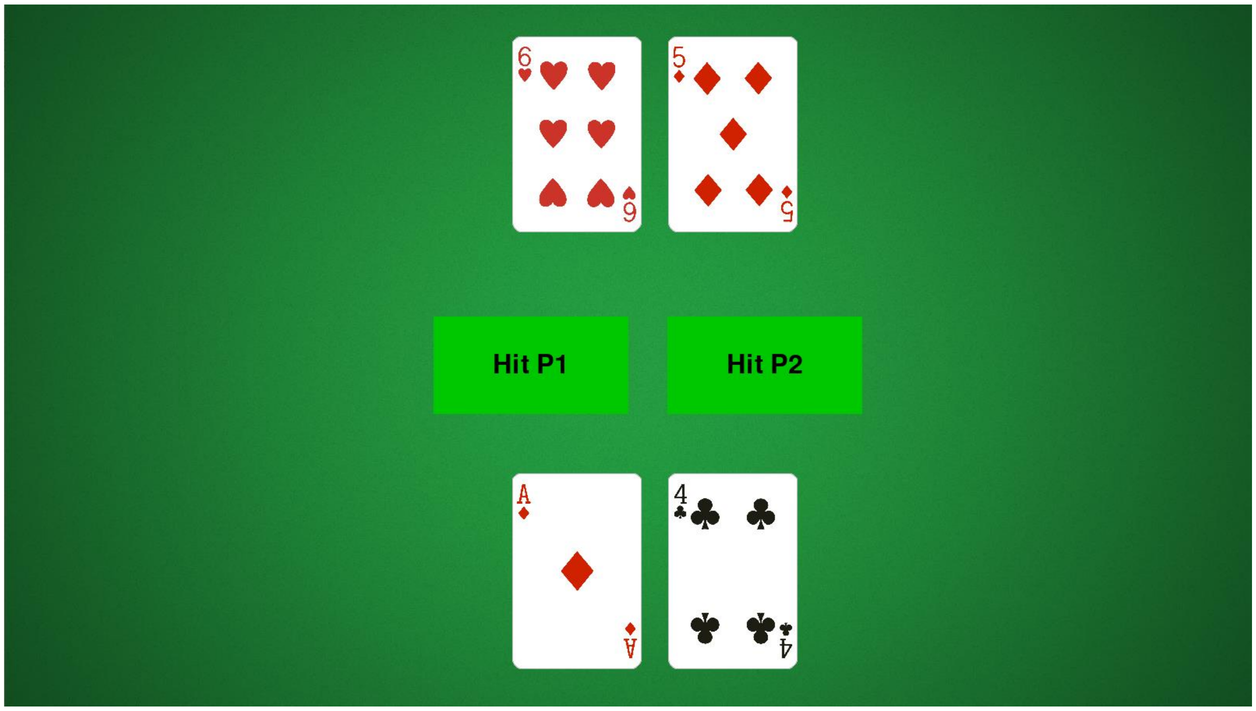
The main_game() function will run the function make_deck() and distribute 2 initial cards to the player’s deck, the total value of these cards will then be added to a total number for each individual players. The function will them render the 2 initial cards in each player’s hand and 2 buttons that has the text “Hit P1” and “Hit P2” respectively. When Hit P1 button is clicked it will append a card from the initial deck and, render the card immediately next to the cards of player 1.



The main_game() function will also check if a player's total has gone over 21, if a player's total has gone over 21, the other player wins. In this example player 1 has exceeded 21 and lost the game. When a player wins the game, a text will appear in the middle of the screen stating that they have won the game.



After a round ends, the game will then go back to the intro screen and if they click deal again they will be dealt a new set of cards.



IV.a Lessons that have been learned

1. ***The use of enum library***

   The enum library is used to assign a set of values or variables to a string or an integer value inside a class.
2. ***Color changing buttons***

   I learned to change a color of a rectangle when a mouse is hovered over it using the position of the mouse and if the mouse position is within the set boundaries the rectangle will change color.

IV.b Problems That Have Been Overcome

During the start of the project I had a vague idea of what I was going to make however, I soon realized that time is such an important factor in a project such as this one. Due to various circumstances I had less time than I thought I did and had to hurry to both learn and finish the project in time. I thought I was very close to finishing the project, but I wasn't, I soon realized that the project won't be completed in time and had to change it. Luckily for me, the new idea that I wanted to do was very similar to the idea that I previously had, so some functions of my code can be used for my new idea. Thankfully I finished the project in time, but I still felt unsatisfied since I know I could've finished my initial idea but due to various circumstances it just felt that it would fail. So, this project has taught me a lesson in important time management and knowing what's in my capabilities.

Resources:

http://acbl.mybigcommerce.com/52-playing-cards/ (The png for the deck of cards)

https://www.free-freecell-solitaire.com/solitaire/images/backgrounds/1920x1200/green_felt.jpg (background image for the project)

https://pythonprogramming.net/pygame-python-3-part-1-intro/

(Source code/Tutorial I followed for the project)

V. Source Code

Project_Final.py

```python
import pygame
import time
from enum import IntEnum
import cardImage as ci
from random import *
from button import Button
from deck import Deck

pygame.init()

display_width = 1600
display_height = 900

background_image = pygame.image.load(r'C:\Users\Nicholas Arthur\Desktop\python\Fi
nal Project\background.jpg')
background_image = pygame.transform.scale(background_image, (1600, 900))

black = (0,0,0)
white = (255,255,255)
red = (200,0,0)
green = (0, 200, 0)
lightgreen = (0, 255, 0)
lightred = (255, 0, 0)

gameDisplay = pygame.display.set_mode((display_width,display_height))
pygame.display.set_caption('Blackjack')
clock = pygame.time.Clock()
```

```python
#list of cards in the initial deck, and decks of the player and the player2

player1Cards = []
player2Cards = []
temp_deck = []
full_deck = []

#assigning a card value to an integer value
class Card_value (IntEnum):
    TWO = 2
    THREE = 3
    FOUR = 4
    FIVE = 5
    SIX = 6
    SEVEN = 7
    EIGHT = 8
    NINE = 9
    TEN = 10
    JACK = 10
    QUEEN = 10
    KING = 10
    ACE = 11

#assigning each symbol/suit with a integer value
class Suit (IntEnum):
    SPADES = 0
    DIAMONDS = 1
    CLUBS = 2
    HEARTS = 3

#defining the cards with a value and a symbol/suit
class Playing_Cards:
    def __init__(self, card_value, card_suit):
        self.value = card_value
        self.suit = card_suit

#making a new full set of cards
def make_deck():
    for suit in Suit:
        for value in Card_value:
            temp_deck.append(Playing_Cards(Card_value(value), Suit(suit)))


#defining a text
def text_objects(text, font):
    textSurface = font.render(text, True, black)
    return textSurface, textSurface.get_rect()

#defining a text message display
def message_display(text):
    largeText = pygame.font.Font('impact.ttf',115)
    TextSurf, TextRect = text_objects(text, largeText)
    TextRect.center = ((display_width/2),(display_height/2))
    gameDisplay.blit(TextSurf, TextRect)

    pygame.display.update()

    time.sleep(2)

def win_screen_p1():
    message_display("Player 1 Wins")
    del player1Cards[:]
    del player2Cards[:]

def win_screen_p2():
    message_display("Player 2 Wins")
    del player1Cards[:]
    del player2Cards[:]

def tie_screen():
    message_display("Tie")
    del player1Cards[:]
    del player2Cards[:]

#screen for the intro screen of the game
def game_intro():

    intro = True

    while intro:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
                quit()

        gameDisplay.blit(background_image,(0,0))
        largeText = pygame.font.Font('impact.ttf',115)
        TextSurf, TextRect = text_objects("Blackjack", largeText)
        TextRect.center = ((display_width/2),(display_height/2))
        gameDisplay.blit(TextSurf, TextRect)
        mouse = pygame.mouse.get_pos()
        click = pygame.mouse.get_pressed()
        button_deal = Button("Deal", 400, 600, 250, 125, green, lightgreen, main_
game)
        button_deal.make_button()

        button_quit = Button("Exit", 950, 600, 250, 125, red, lightred, quit)
        button_quit.make_button()
        pygame.display.update()
        clock.tick(15)

#function for the main game
def main_game():
    make_deck()
    full_deck = Deck(temp_deck)
    full_deck.initial_deal(player1Cards)
    full_deck.initial_deal(player2Cards)
    counterp1 = 2
    counterp2 = 2
    start = True
    xp1 = 550
    yp1 = 400
    w = 250
    h = 125
    xp2 = 850
    yp2 = 400

    gameDisplay.blit(background_image,(0,0))
    totalp1 = player1Cards[0].value + player1Cards[1].value
    totalp2 = player2Cards[0].value + player2Cards[1].value
    while start:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
                quit()

        #displays the image of the cards that were in the initial hand
        for cards in range(2):
            gameDisplay.blit(ci.getCardImage(player1Cards, cards), (650 + (200*ca
rds), (600)))
            gameDisplay.blit(ci.getCardImage(player2Cards, cards), (650 +(200*car
ds) , 40))
            pygame.display.flip()
        mouse2 = pygame.mouse.get_pos()
        click = pygame.mouse.get_pressed()

        #button for dealing a single card to player one or player 2
        if xp1 + w > mouse2[0] > xp1 and yp1 + h > mouse2[1] > yp1:
            pygame.draw.rect(gameDisplay, lightgreen, (xp1, yp1, w, h))
            if click[0] == 1:
                full_deck.deal_card(player1Cards)
                gameDisplay.blit(ci.getCardImage(player1Cards, -
1), (650 + 200 * counterp1, 600))
                counterp1 += 1
                totalp1 += player1Cards[-1].value
                pygame.display.update()

        else:
            pygame.draw.rect(gameDisplay, green, (xp1, yp1, w, h))


        if xp2 + w > mouse2[0] > xp2 and yp2 + h > mouse2[1] > yp2:
            pygame.draw.rect(gameDisplay, lightgreen, (xp2, yp2, w, h))
            if click[0] == 1:
                full_deck.deal_card(player2Cards)
                gameDisplay.blit(ci.getCardImage(player2Cards, -
1), (650 + 200 * counterp2, 40))
                counterp2 += 1
                totalp2 += player2Cards[-1].value

        else:
            pygame.draw.rect(gameDisplay, green, (xp2, yp2, w, h))
        #text for button of Hit P1 and Hit P2
        smallText = pygame.font.SysFont("impact.ttf",50)
        textSurf, textRect = text_objects("Hit P1", smallText)
        textRect.center = ( (xp1+(w/2)), (yp1+(h/2)) )
        gameDisplay.blit(textSurf, textRect)

        textSurf2, textRect2 = text_objects("Hit P2", smallText)
        textRect2.center = ( (xp2+(w/2)), (yp2+(h/2)) )
        gameDisplay.blit(textSurf2, textRect2)
        pygame.display.update()

        #winning conditions
        print(totalp2)
        if totalp1 > 21:
            start = False
            win_screen_p2()
        if totalp2 > 21:
            start = False
            win_screen_p1()
        if totalp1 == 21 and totalp1 == totalp2:
            start = False
            tie_screen()
        pygame.display.flip()

game_intro()
```

button.py

```python
import pygame

display_width = 1600
display_height = 900
gameDisplay = pygame.display.set_mode((display_width,display_height), pygame.FULL
SCREEN)
black = (0,0,0)
white = (255,255,255)
red = (200,0,0)
green = (0, 200, 0)
lightgreen = (0, 255, 0)
lightred = (255, 0, 0)

class Button:
    def __init__(self,msg,x,y,w,h,inactive,active,action=None):
        self.msg = msg
        self.x = x
        self.y = y
        self.w = w
        self.h = h
        self.ic = inactive
        self.ac = active
        self.action = action
        self.mouse = pygame.mouse.get_pos()
        self.click = pygame.mouse.get_pressed()
    #function to make a button with adjustable size and position
    def make_button(self):
        if self.x + self.w > self.mouse[0] > self.x and self.y+self.h > self.mous
e[1] > self.y:
            pygame.draw.rect(gameDisplay, self.ac,(self.x,self.y,self.w,self.h))
            if self.click[0] == 1 and self.action != None:
                intro = False
                self.action()
        else:
            pygame.draw.rect(gameDisplay, self.ic,(self.x,self.y,self.w,self.h))

        smallText = pygame.font.SysFont("impact.ttf",50)
        textSurf, textRect = text_objects(self.msg, smallText)
        textRect.center = ( (self.x+(self.w/2)), (self.y+(self.h/2)) )
        gameDisplay.blit(textSurf, textRect)

def text_objects(text, font):
    textSurface = font.render(text, True, black)
    return textSurface, textSurface.get_rect()
```

deck.py

```python
from random import *

class Deck:
    def __init__(self,deck):
        self.deck = deck
    #function to append a single card into a specific list and removing the card
from the deck
    def deal_card(self,slot):
        rand_card = randint(0, len(self.deck) -1)
        slot.append(self.deck[rand_card])
        self.deck.pop(rand_card)
    #function to deal 2 cards immediately at the start of the game
    def initial_deal(self, playercards):
        for i in range(2):
            self.deal_card(playercards)
```