Evandra Harya Putra - 2602118433(L2AC)

OBJECT-ORIENTED PROGRAMMING

FINAL PROJECT

Lecturer:

JUDE JOSEPH LAMUG MARTINEZ, MCS

BINUS UNIVERSITY INTERNATIONAL

2023

TABLE OF CONTENTS

Evandra Harya Putra - 2602118433(L2AC)

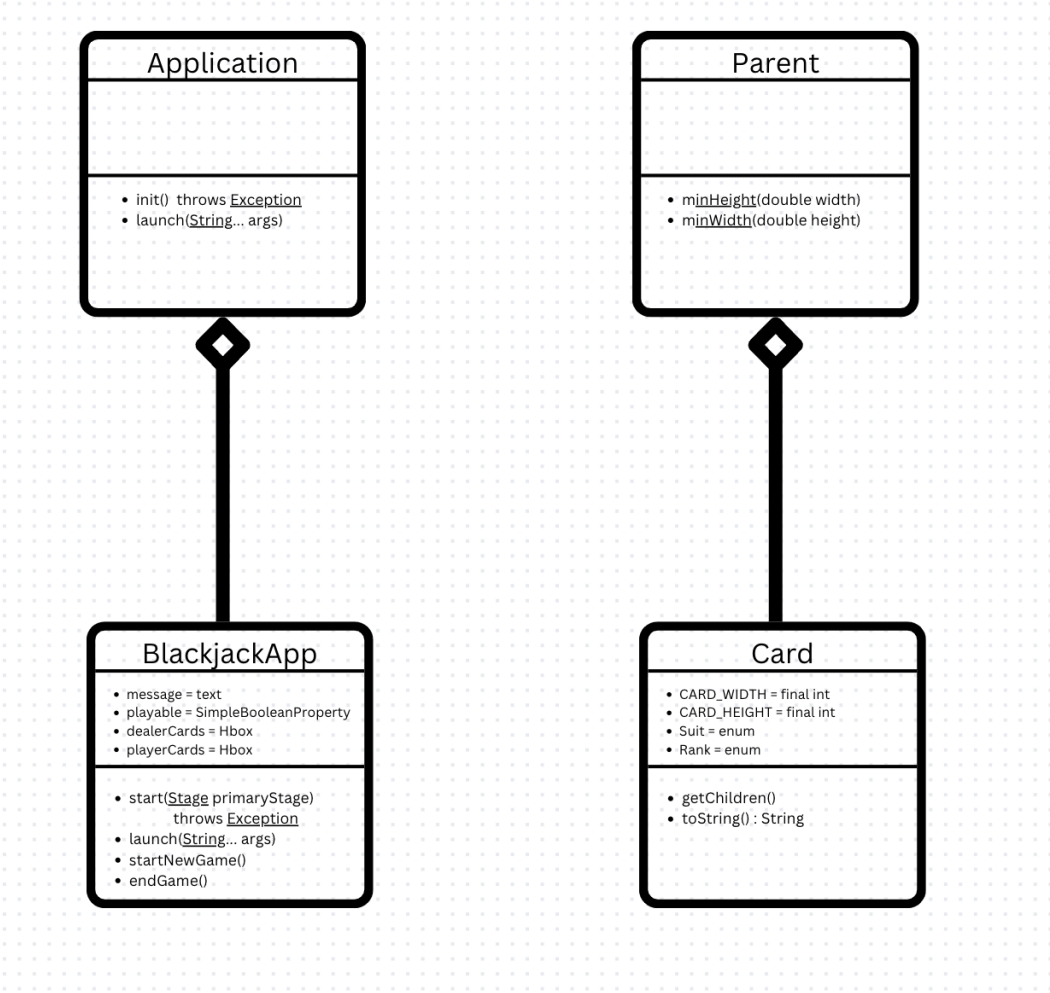
Blackjack Game

A. Project documentation :

Brief description :

Blackjack is a card game where the player will face the dealer. This game is widely known for its simplicity and popular among people who frequently went to the casino. Besides the casino, people could also play the game at home with their friends. Not only one on one, they can play the game up to 7 players. With slightly different restrictions from the casino, but people could enjoy the game without going to the casino. Moreover, not all people could go to the casino because of the age restrictions. Therefore, I decided to make a Blackjack game where people could play the game with their laptop one on one with the computer as the dealer. The restrictions that I implement are the same as in the casino. People could enjoy the game without being worried as in the casino there will be age restriction and the place is fairly precarious. There are no probabilities in this game, so each player will win the game based on their luck. This game is created with the help of the youtuber named Almas Baimagambetov. And the module JavaFX is used in order to make this game. The expected outcome of this game is that people could enjoy this game without any restriction. Moreover, the game runs smoothly without any interference.

Class Diagram :



From the class diagram, there are 2 class diagrams each for Card class and BlackjackApp class. Since the 2 classes inherit from a different parent. As we can see, BlackjackApp class inherits from the Javafx application while Card class inherits from the Javafx Parent. The use of Javafx Application is to launch the game and in order for the game to be playable and interactive for the user. This enables us to create and design the window for the game. On the other hand, Javafx parent is used to define the hierarchy therefore we can add or remove a child node in the hierarchy.

Modules :

```
import javafx.scene.Parent; // to create a hierarchy to add and remove node
import javafx.scene.image.Image; //to define an image so we can input image
import javafx.scene.image.ImageView; //to view the image and show it
import javafx.scene.layout.StackPane; //move child nodes from back to front
import javafx.scene.paint.Color; //to apply color to certain application
import javafx.scene.shape.Rectangle; //define a rectangle
import javafx.scene.text.Font; //set the font of the text
import javafx.scene.text.Text; //make a visible text in the window
import javafx.beans.property.SimpleIntegerProperty;//to initiate the value of
string
import javafx.collections.ObservableList; //tracking changes in the list
import javafx.scene.Node; //a set or tree data structure
import javafx.application.Application; //to make the application run
import javafx.beans.property.SimpleBooleanProperty; //to state the boolean of
a variable
import javafx.beans.property.SimpleStringProperty; //to add a string
import javafx.geometry.Insets; //store the inside offsets or a rectangle
import javafx.geometry.Pos; //vertical and horizontal alignment
import javafx.scene.Scene; //a container for all the content in scene graph
import javafx.scene.control.Button;//to control the button in the application
import javafx.scene.control.TextField; //add or remove a text-field
import javafx.scene.layout.HBox; //horizontal box
import javafx.scene.layout.Pane; //ui that contains the child nodes
import javafx.scene.layout.Region; //base class of ui and controls
import javafx.scene.layout.VBox; //vertical box
import javafx.stage.Stage; //display the window
```

Modules :

```
public class Card extends Parent{
   private static final int CARD_WIDTH = 100;
   private static final int CARD_HEIGHT = 140;

   enum Suit {
       HEARTS, DIAMONDS, CLUBS, SPADES;

       final Image image;

       Suit() {
           this.image = new
Image(Card.class.getResourceAsStream("images/".concat(name().toLowerCase()).co
ncat(".png")),32, 32, true,true);
       }
   }

   enum Rank {
       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);

       final int value;
       Rank(int value) {
           this.value = value;
       }

       String displayName() {
           return ordinal() < 9 ? String.valueOf(value) : name().substring(0,
1);
       }
   }
```

```
public void takeCard(Card card) { //function to take a card from the deck
   cards.add(card); //add card to the list of tha cards

    if (card.rank == Rank.ACE) { //if the player gets aces, we increment the
value of aces
       aces++;
   }

    if (value.get() + card.value > 21 && aces > 0) { //if the card value is
greater than 21, aces is greater than 0
       value.set(value.get() + card.value - 10);     //then we count ace as '1'
not '11' we add the value of cards with the aces(1) then we count ace as '1'
not '11'
       aces--; //decrement the value of aces into 0
   }
   else {
        value.set(value.get() + card.value); //if the above isn't true, we add
the value of the cards that the player had,
       // add with the values that the player just obtained
   }
}
```

```
public final void refill() {
   int i = 0;          //let the first index be 0
   for (Suit suit : Suit.values()) {    //check the suit values
       for (Rank rank : Rank.values()) { //check the rank values
            cards[i++] = new Card(suit, rank);  //increment the index everytime
we add something to the deck
       }
   }
}

public Card drawCard() {   //to claim a random card everytime we press "hit"
from the deck
   Card card = null;  //while the card is empty, and we don't have any cards
   while (card == null) {
        int index = (int)(Math.random()*cards.length);  //multiply random number
with the index of the array
            card  =  cards[index]; //assign  the  card  with  the  result  of  the
multiplication
       cards[index] = null;
   }
   return card;
}
```

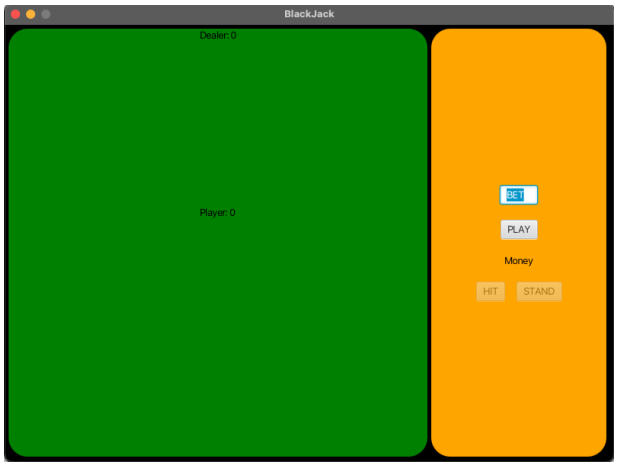
```
private Parent CreateContent()
private void startNewGame()
private void endGame()
public void start(Stage primaryStage) throws Exception
```

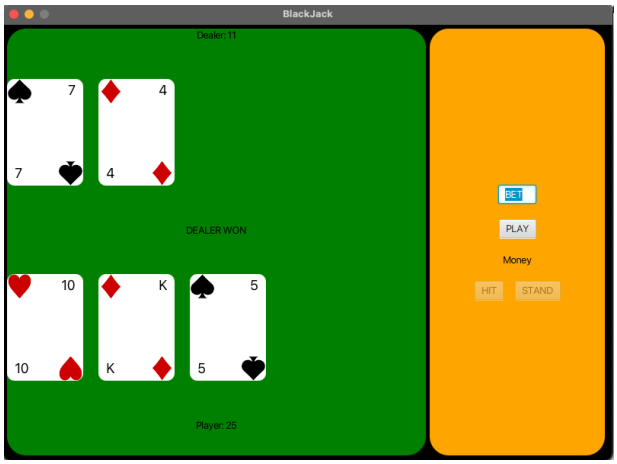
Evandra Harya Putra - 2602118433(L2AC)

Lessons learned :

From this object oriented programming final project, I learned a lot as a student since I decided to use different modules compared to what most of the students used. For the GUI itself there are many trials and errors especially in implementing the content itself. However, throughout the time without even realizing it, I gained new knowledge on how to use the JavaFx and also made games using the Java programming language. The process is quite long, since I'm trying to make a games or system that would be enjoyable or useful for everyone. I do hope through this game, people could enjoy the game. From this final project, I intend to learn more about Java and make some usable systems and try different modules and algorithms to work on.

Screenshots of our applications :





References :

Almas Baimagambetov. (2015, February 20). *JavaFX Game: BlackJack* [Video]. YouTube. https://www.youtube.com/watch?v=I2EVF50V-84

All Classes (JavaFX 8). (2015, February 10). https://docs.oracle.com/javase/8/javafx/api/allclasses-noframe.html

JavaFX 8. (n.d.). https://docs.oracle.com/javase/8/javafx/api/toc.htm