Plinko

Final Report

Good Noodles

Jack Kabil

Mark Sanghera

Scott Riddle

Course: CPSC 224 - Software Development

I. Introduction and Project Description

This document details the design and creation of Plinko. It includes introductions for each of the team members, the functional and nonfunctional project requirements, the design of our solution approach, our plan for testing the game, the details of your implementation, and our considerations for the future.

Plinko is a gravity-based gambling game. The player's goal is to cash out with more money than they originally cashed in with. To achieve this, the player bets a certain amount of money on a ball, sends the ball into a system of pegs, and watches as the ball reaches the bottom. Beneath the system of pegs, there are thirteen buckets, each with a multiplier. When a player sends a ball, the amount they bet is subtracted from their bank account. When the ball reaches the bottom and falls into a bucket, the ball's bet is multiplied by the multiplier and returned to the player's bank account.

In our Plinko game, there are other additional settings and features, such as the ball color setting, the trace mode setting, player stats tracking, the cash out function, and the watch ad function.

II. Team Members - Bios and Project Roles

Scott Riddle is a computer science student interested in software development. His skills include experience in Java and C++. He is also interested in computer hardware and builds and maintains desktop PCs for his friends and family. His prior school projects have included topics such as various sorting algorithms, file IO, and object-oriented programming. He has also worked on a personal project that uses calls to the Spotify API. For this project, he was responsible for the overall structure of the program, the cash out panel, the game logic, and the settings.

Jack Kabil is a computer science student interested in software development and game design. Prior projects include merge sort, quick sort, and farkle in school environments and then documentation, interactive calendar, and bill of lading generator in work environments. Jack's skill include C/C++, C#, Java, and a small amount of SQ and Python. For this project, Jack was responsible for the physics, play, and watch Ad panel designed and implementation.

Mark Sanghera is a computer science student minoring in mathematics and statistics interested in cybersecurity and data science. His prior projects include a terminal and GUI implementation of the dice game Farkle, several sorting and data algorithms, and a data analysis between the relationship of cereals released and natural disasters using python and the pandas package. Mark's skills include C/C++, Python, Java/Java Swing, SQL, and HTML. For this project his responsibilities include the design and implementation of the stats panel, the player class, the

scoring, updating the player's status, and storing and loading the files that keep track of the player's status and total money spent.

III. Project Requirements

In order to make a successful Plinko game we made a list of requirements for the game. These requirements included a working physics engine, ability to assign monetary value to the dropped ball, a player bank account, slot rewards / multipliers, and some additional features such a stats panel and remembering returning players.

Major Features

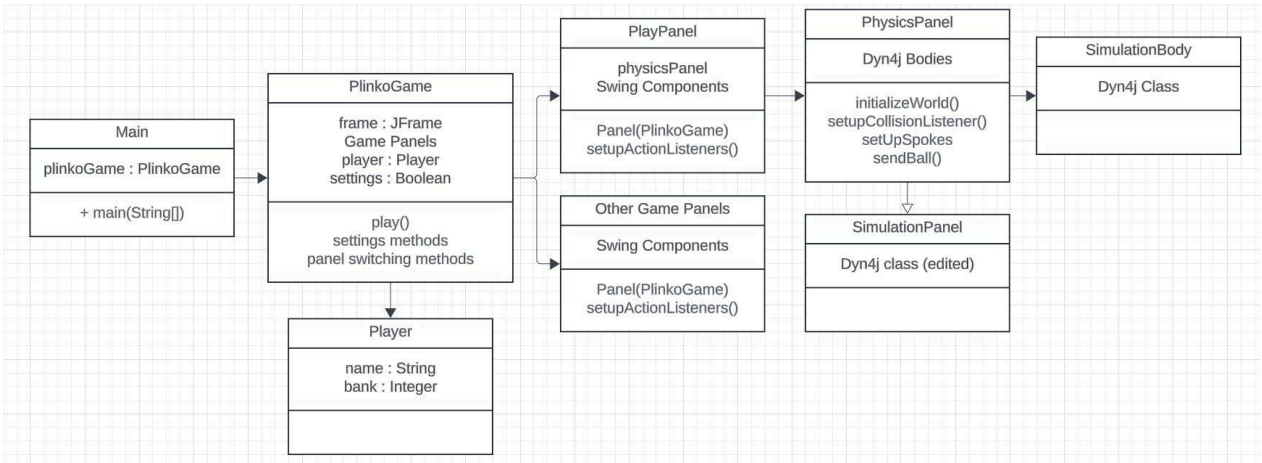
Feature	*Description*
Physics Engine	This is essential for the game to work in the intended manner.
Assigning monetary value to dropped ball	The dropped ball must have a monetary value assigned to it so that it can be multiplied by the zone it lands in.
Player bank amount	The player will need to have a banked amount of money. The bets placed by the user will come from this bank, and any money they win will be added to this banked amount.
Slot rewards	When the ball lands in a slot, a multiplier will apply to the monetary value of the ball that was dropped. There will also be a jackpot slot that will appear periodically which will allow users to win the highest amount of money.
Extra Features	Statistics page (biggest winners, losers), watch an ad for more money, login with name and tracking player.

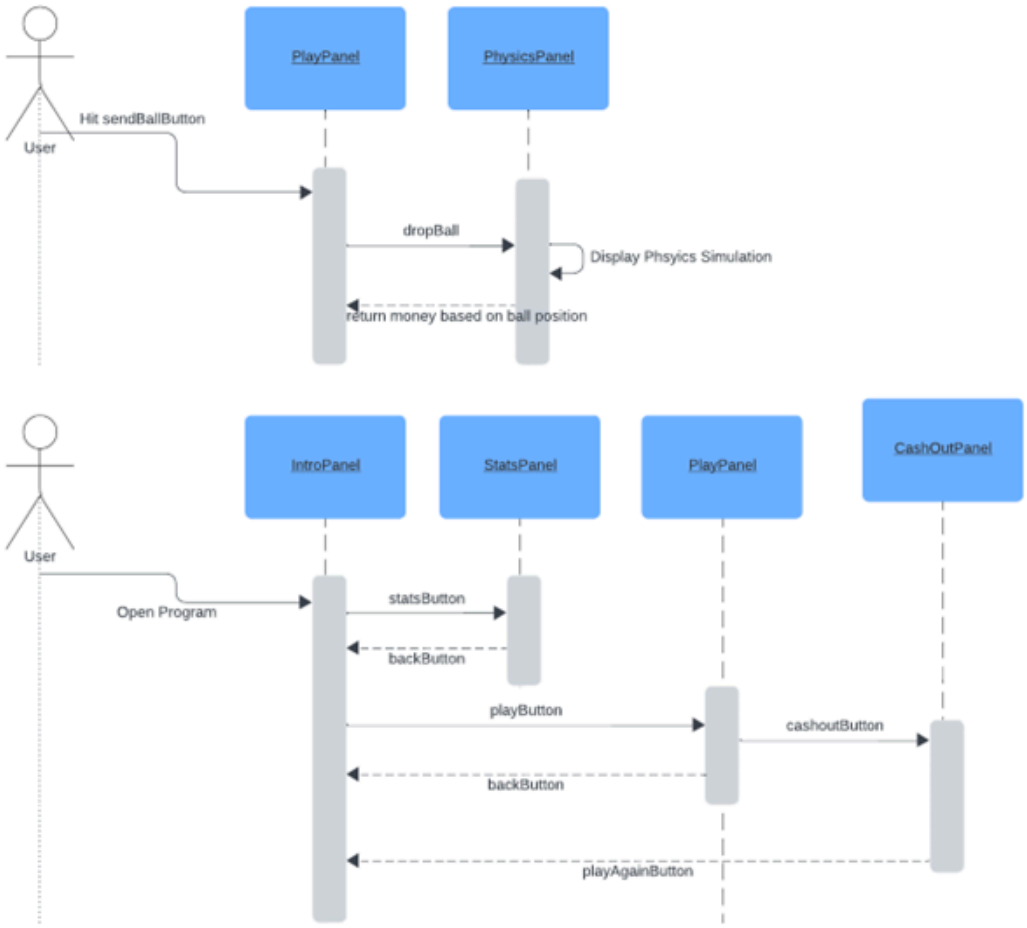
IV. Solution Approach

```java
// the game class holds the frame, panels, player, and the settings
public class PlinkoGame {

    // the frame, panels,
    public JFrame frame;
    private IntroPanel introPanel;
    private PlayPanel playPanel;
    private CashOutPanel cashOutPanel;
    private StatsPanel statsPanel;
    private watchAdPanel watchAdPanel;
    private PlayerCreationPanel playerCreationPanel;

    // player,
    public Player player;

    // and the settings
    private Boolean traceSetting;
    private Boolean rainbowBall;
```

```java
// instantiate each panel (needed here, not just in panel switching methods fsr)
this.playerCreationPanel = new PlayerCreationPanel(this);
this.playPanel = new PlayPanel(this);
this.introPanel = new IntroPanel(this);
this.cashOutPanel = new CashOutPanel(this);
this.statsPanel = new StatsPanel(this);
this.watchAdPanel = new watchAdPanel(this);
```

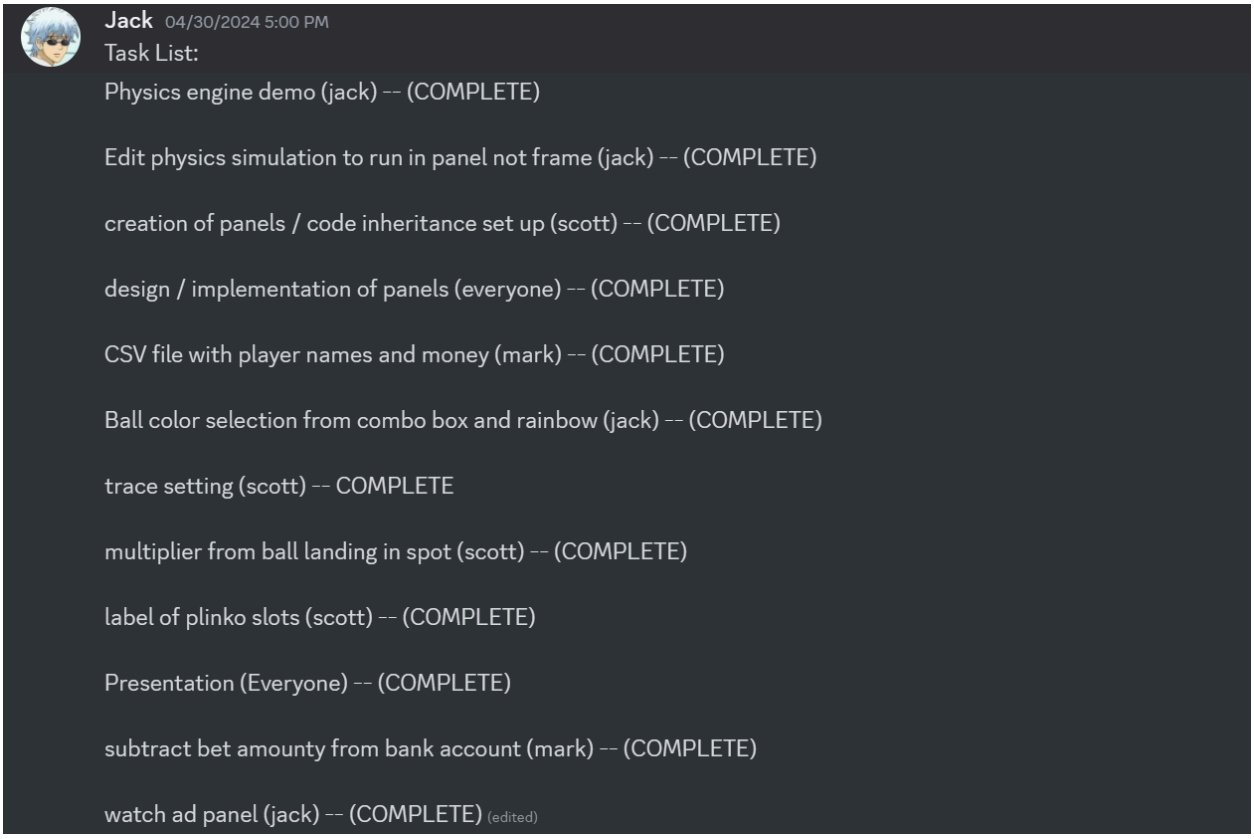

main \ java \ edu \ gonzaga
input
media
BankManager.java 3
Camera.java
CashOutPanel.java
Graphics2DRenderer.java
ImageBody.java
InsufficientFundsException.java
IntroPanel.java
Main.java
PhysicsPanel.java
Player.java
PlayerCreationPanel.java 3
PlayerGainsComparator.java
PlayerIDComparator.java
PlayPanel.java
PlinkoGame.java
Shrek.java
SimulationBody.java 3
SimulationPanel.java
StatsPanel.java
watchAdPanel.java


Main
plinkoGame : PlinkoGame
+ main(String[])
PlinkoGame
frame : JFrame
Game Panels
player : Player
settings : Boolean
play()
settings methods
panel switching methods
Player
name : String
bank : Integer
PlayPanel
physicsPanel
Swing Components
Panel(PlinkoGame)
setupActionListeners()
Other Game Panels
Swing Components
Panel(PlinkoGame)
setupActionListeners()
PhysicsPanel
Dyn4j Bodies
initializeWorld()
setupCollisionListener()
setUpSpokes
sendBall()
SimulationPanel
Dyn4j class (edited)
SimulationBody
Dyn4j Class



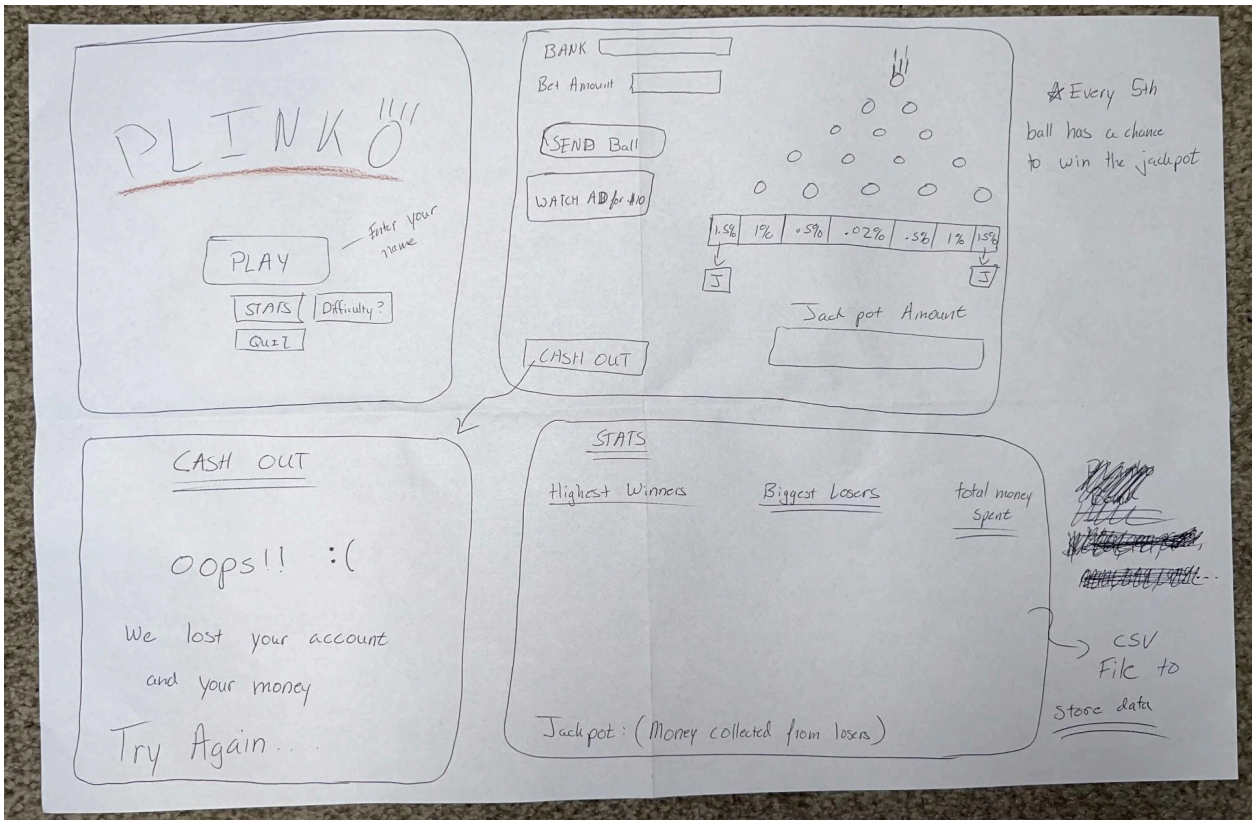
V. Test Plan

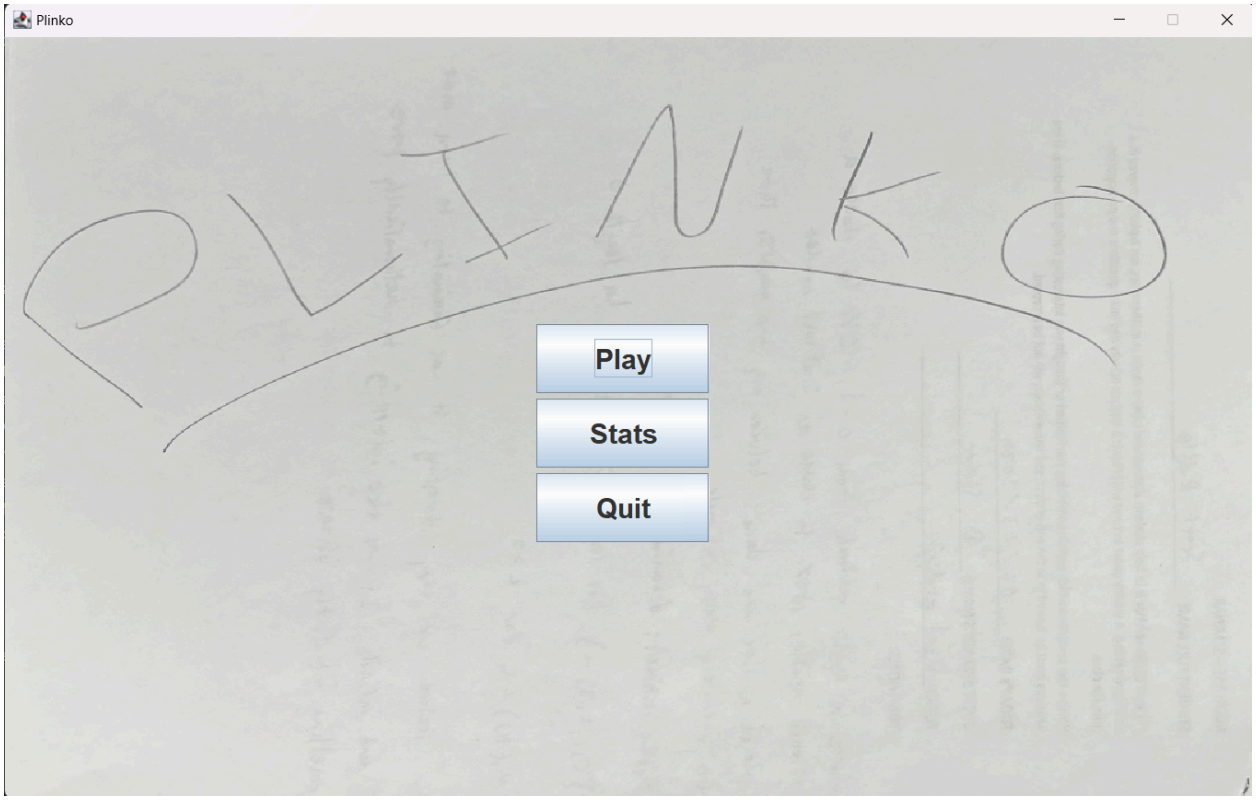
Here is the checklist of tasks that we had to guide us during our project. When a task was completed the message board would be updated to say complete letting all users in discord know.

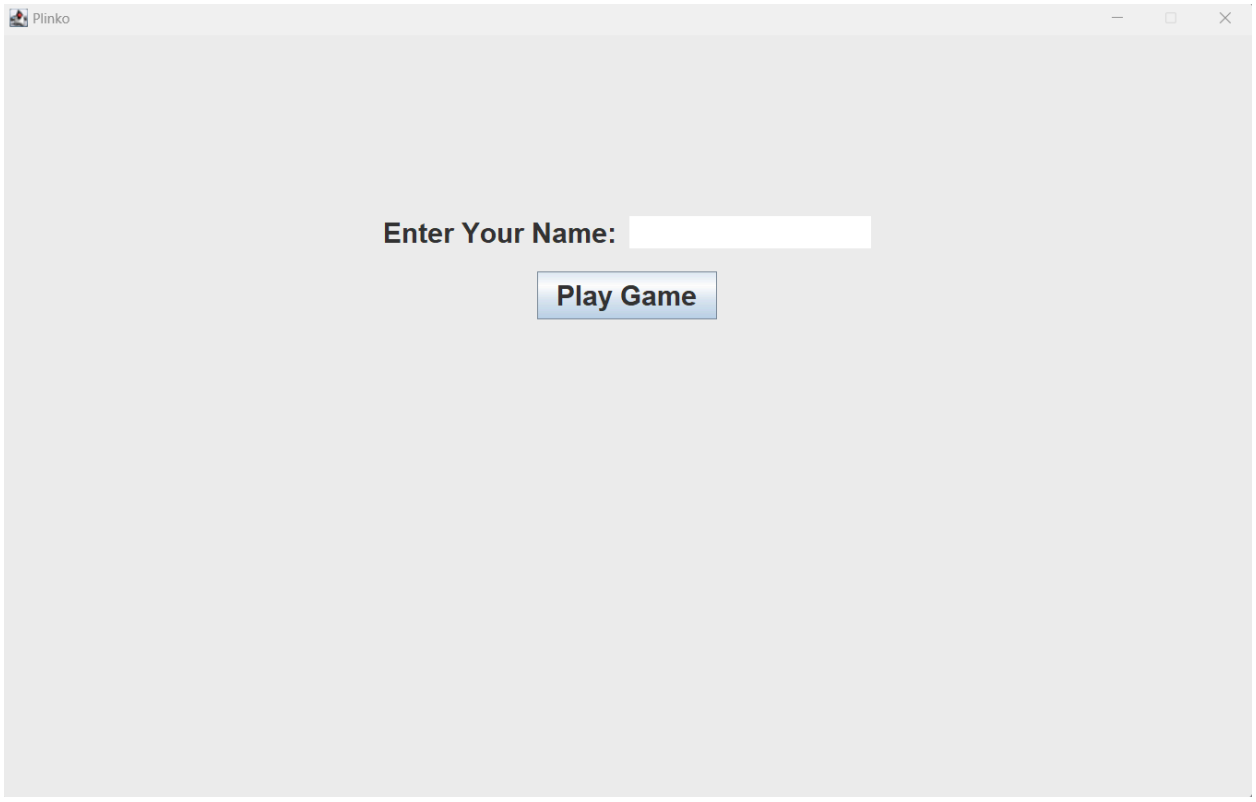


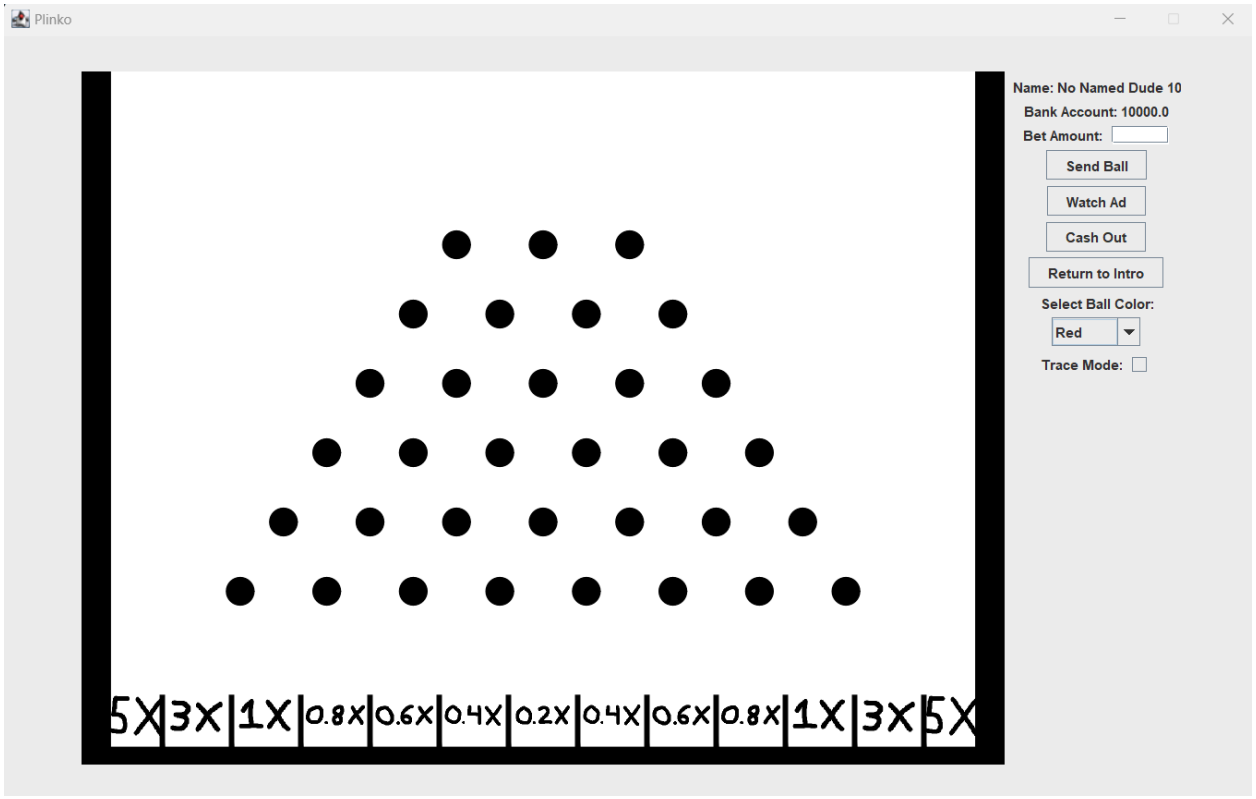
VI. Project Implementation Description

The parts of the proposed architecture that the team implemented were, the intro panel, the play panel, the cashout panel and the stats panel. In addition to the proposed architecture that the team implemented is the player creation panel and watch ad panel. The intro panel is done to completion containing the big game name label/background, the play button, the stats button, and quit button. The player creation panel is fully complete with a play game button, textfield asking for a name and providing an unknown player name when no name is given. The play panel is almost fully complete with the send ball button, bet text field, watch ad button, return to intro button, ball color selection combo box, and trace mode check box. Originally, there would be a jackpot slot that would appear occasionally, but we ran out of time to implement it. The cashout panel is built to completion with a cashout button that doesn't work as intended, and a keep gambling button that returns to the play panel. The watch ad panel is built to completion with a close panel button on a timer that returns to the play panel and a random image being displayed for the ad. Lastly, the stats panel is built to completion with a back to intro button, stats label, two scrollable panes that displays the highest winners and biggest losers with their name and gains/winnings and a label that tracks the total money spent in the game.


PLINKO
PLAY
Enter your name
STATS
Difficulty?
QUIZ
BANK
Bet Amount
SEND Ball
WATCH AD for $10
1.5% 1% .5% .02% .5% 1% 1.5%
J
J
Jack pot Amount
CASH OUT
☆ Every 5th ball has a chance to win the jackpot
CASH OUT
oops!! :(
We lost your account and your money
Try Again...
STATS
Highest Winners
Biggest Losers
total money spent
Jackpot: (Money collected from losers)
CSV File to Store data

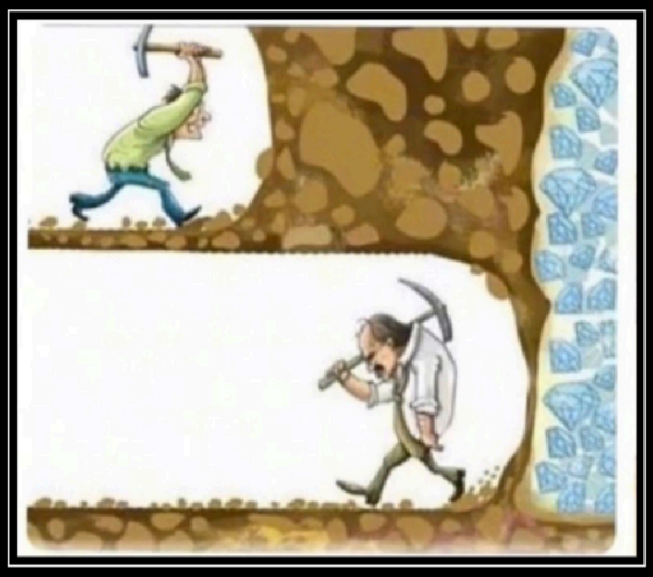

Plinko
PLINKO
Play
Stats
Quit


Plinko
Enter Your Name:
Play Game


Plinko
5X
3X
1X
0.8X
0.6X
0.4X
0.2X
0.4X
0.6X
0.8X
1X
3X
5X
Name: No Named Dude 10
Bank Account: 10000.0
Bet Amount:
Send Ball
Watch Ad
Cash Out
Return to Intro
Select Ball Color:
Red
Trace Mode:


Plinko
X

Plinko



cash out

KEEP GAMBLING

KEEP GAMBLING!!!

Plinko

STATS

Back to Intro

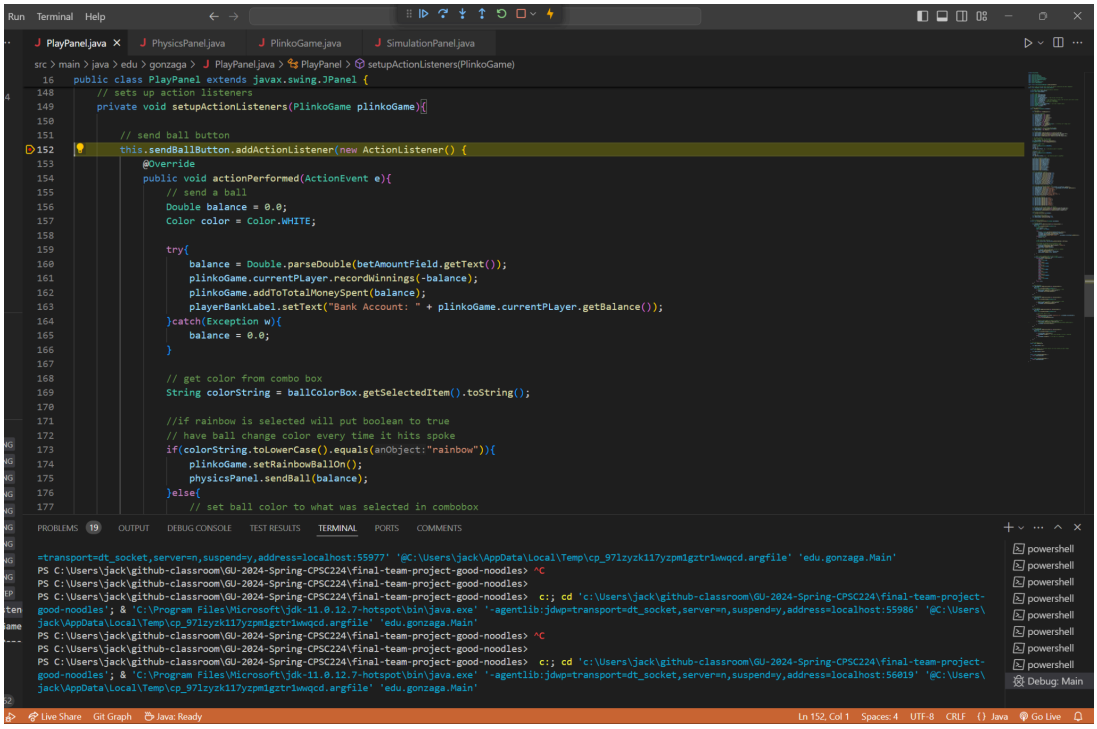
Highest Winners:

Player Name	Player Total Winnings
scotty p	42198.4
No Named Dude 2	39900.0
mark again	15060.0
Markus	40.0
No Named Dude 1	38.0
No Named Dude 10	0.0
scott	0.0
No Named Dude 0	-200.0

Biggest Losers:

Player Name	Player Total Lossings
jacques	-9939.128
mark	-8781.848
Mark	-706.0
No Named Dude 0	-200.0
scott	0.0
No Named Dude 10	0.0
No Named Dude 1	38.0
Markus	40.0

Total Money Spent: 5330.0



*Debugging

[GitHub Link]

VII. Future Work

To make the game deliverable, some existing bugs would have to be dealt with. For example, a bug exists such that a player can bet a negative amount of money on a ball. This feature was not intentional; the player should not be able to bet a negative amount on a ball.

Past delivery, more features can be added to the game. One feature we could include is the use of a database to hold the player stats instead of CSV files. This would allow the player stats to update dynamically without the need for pushing and pulling to and from GitHub. Furthermore, we could add a difficulty setting. This setting could change the number of pegs, the number of buckets, and the multiplier value in each bucket. Higher difficulties could have a wider range of multiplier values, leading to a higher risk / higher reward scenario.

VIII. Glossary

Physics Engine - Library used to run the plinko physics simulation

IX. References

N/A

X. Appendices

N/A