vim Window.java

```java
/*

Jared Dyreson
CWID: 889546529
Window.java -> Implements the main GUI interface for this application (FRONT END)

*/

import javax.swing.*;
import java.text.MessageFormat;
import java.awt.*;
import javax.swing.JFrame;
import java.awt.event.*;
import java.lang.Math;
import java.util.ArrayList;
import java.util.Arrays;

public class Window extends JFrame implements ActionListener{

        // suppress an annoying pop-up information -> https://stackoverflow.com/questions/7823477/warning-serial-serializable-class-someclass-has-no-definition-of-serialversio

        private static final long serialVersionUID = 12996;
        // how big the window will be
        final int FRAME_WIDTH = 500, FRAME_HEIGHT = 500;

        // all elements used in the JFrame and JPanel
        JLabel heading, cash_avail, left_message, center_message;
        JButton button, cash_out_button;
        JTextField text_field, total_amount;

        // all panels

        JPanel main_panel, button_panel, cash_flow, bottom_row, cash_quick_bet, center_divider;

        // arrays used to prevent repeating code
        JComboBox<String> quick_bet_drop_down;
        ArrayList<JButton> button_list = new ArrayList<>();
        String[] quick_bet_options = {"Quick Bet", "$100", "$350", "$500"};

        // external classes used to handle game logic
        DiceGame game_handler = new DiceGame();
        Player p = new Player();


        public Window(){
                // classes with inheritence must include this FIRST
                super("Casino Simulator");

                // set up values for the frame
                setSize(FRAME_WIDTH, FRAME_HEIGHT);
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                setLocationRelativeTo(null);
                setLayout(new BorderLayout());

                // give all the buttons and labels their respective values
                button = new JButton("Roll Dice");
                cash_out_button = new JButton("Cash Out");

                heading = new JLabel("Welcome to the High Rollers Game");
                heading.setFont(new Font("Times New Roman", Font.BOLD, 20));

                cash_avail = new JLabel("Cash Available");
                cash_avail.setFont(new Font("Times New Roman", Font.BOLD, 16));

                left_message = new JLabel("VEGAS BABY!");
                left_message.setFont(new Font("Times New Roman", Font.BOLD, 14));

                center_message = new JLabel("Place Your Bet");
                center_message.setFont(new Font("Times New Roman", Font.BOLD, 14));

                text_field = new JTextField(12);
                total_amount = new JTextField();

                total_amount.setText("$"+String.valueOf(p.get_fund_amount()));
                total_amount.setEditable(false);

                quick_bet_drop_down = new JComboBox<>(quick_bet_options);
                quick_bet_drop_down.addActionListener(this);

                main_panel = new JPanel();
                button_panel = new JPanel();
                cash_flow = new JPanel();
                bottom_row = new JPanel();
                cash_quick_bet = new JPanel();
                center_divider = new JPanel();

                button_panel.setLayout(new GridLayout(0, 3, 5, 0));
                // add buttons in a more efficient manner
                for(int i = 0; i < 3; ++i){
                        button_list.add(new JButton("D"+String.valueOf(i)));
                        button_panel.add(button_list.get(i));
                }
                // allow for the exit button and the roll dice button to work
                cash_out_button.addActionListener(this);
                button.addActionListener(this);

                // treat each of the JPanel objects as containers, where we put various UI/UX element inA
                // each of these containers are grid layouts, like a piece of graph paper

                cash_flow.setLayout(new GridLayout(2, 1, 5, 5));
                cash_flow.add(cash_avail);
                cash_flow.add(total_amount);

                bottom_row.setLayout(new GridLayout(1, 0, 5, 5));
                bottom_row.add(left_message);
                bottom_row.add(button);

                cash_quick_bet.setLayout(new GridLayout(2, 0));
                cash_quick_bet.add(cash_out_button);
                cash_quick_bet.add(quick_bet_drop_down);

                center_divider.setLayout(new GridLayout(3, 1, 10, 0));
                center_divider.add(center_message);
                center_divider.add(text_field);
                center_divider.add(button);

                // the current JFrame needs to house the sub panel, which subsequently holds all the sub containers
                this.getContentPane().add(main_panel);

                // add all the elements to the frame

                main_panel.add(heading, BorderLayout.NORTH);
                main_panel.add(cash_flow, BorderLayout.WEST);
                main_panel.add(button_panel, BorderLayout.CENTER);
                main_panel.add(center_divider, BorderLayout.CENTER);
                main_panel.add(cash_quick_bet, BorderLayout.EAST);
                // not all elements seem to play by the rules so this was the only way for this to work properly
                add(bottom_row, BorderLayout.SOUTH);

        }

        // allows us to have an event map to a listener
        @Override
        public void actionPerformed(ActionEvent event){
                String action_performed = event.getActionCommand();
                System.out.println(action_performed);

                Object drop_down_option_selected = quick_bet_drop_down.getSelectedItem();
                String converted_value = String.valueOf(drop_down_option_selected);
                String stripped = converted_value.replace("$", "");
                String contents_of_text_field = text_field.getText();

                int calculating_value = 0, value_from_text_field = 0;

                if(action_performed == "comboBoxChanged"){
                        // if the drop down has a vaild number option

                        if(stripped != "Quick Bet"){
                                text_field.setText(stripped);
                        }

                        // if the quick bet option is picked and there is nothing in the textfield option

                        else if(stripped == "Quick Bet" && contents_of_text_field == ""){
                                text_field.setText("0");

                        }
                }
                else if(action_performed == "Cash Out"){
                        // kill the window
                        this.dispose();
                }

                // we really push most the responsibility of the amount being bet on the text field, it removes confusing code
                value_from_text_field = Integer.valueOf(text_field.getText());

                if(action_performed == "Roll Dice"){
                        if(p.has_funds(value_from_text_field)){
                                ArrayList<Integer> list = game_handler.check_win_weight(p, left_message, value_from_text_field);

                                String updated_amount = String.valueOf(p.get_fund_amount());
                                total_amount.setText("$"+updated_amount);
                                for(int i = 0; i < 3; ++i){
                                        JButton button = button_list.get(i);
                                        button.setText(String.valueOf(list.get(i)));
                                }

                        }
                        else if(p.get_fund_amount() == 0){
                                left_message.setText("You ran out of money!");
                        }
                        else{
                                left_message.setText("Insufficient funds :(");
                        }
                }

        }

}
```

vim Window.java

```java
/*

Jared Dyreson
CWID: 889546529
Driver.java -> The two lines of code to run this program

*/

public class Driver {
        public static void main(String[] args){
                // Auto generated with caffine and hddtemp.service
                Window winwoes = new Window();
                winwoes.setVisible(true);
        }
}
```

vim Player.java

```java
/*

Jared Dyreson
CWID: 889546529
Player.java -> Implements a basic Player class that is involved in the dice rolling game and the GUI (BACKEND)

*/

public class Player {
        // Auto generated with caffine and lightdm.service
        private int funds = 1000;
        private boolean win_status = false;

        public boolean has_funds(int bet){
                return (bet <= funds) ? true : false;
        }
        public void add_funds(int value){ funds+=value;}
        public int get_fund_amount(){ return funds; }
}
```

vim DiceGame.java

```java
/*

Jared Dyreson
CWID: 889546529
DiceGame.java -> The dice rolling mechanism that powers this game (BACKEND)

*/

import javax.swing.*;
import java.awt.*;
import javax.swing.JFrame;
import java.awt.event.*;

import java.lang.Math;
import java.util.ArrayList;
import java.util.Arrays;

public class DiceGame {
        // Auto generated with caffine and mintsystem.service

        // get a random integer from range n to k
        public int range(int floor, int celing){
                return (int)(Math.random()*((celing-floor)+1))+floor;
        }
        public ArrayList<Integer> check_win_weight(Player p, JLabel message, int amount){
                int roll_one = range(1, 6), roll_two = range(1, 6), roll_three = range(1, 6);

                // three of a kind

                if((roll_one == roll_two) && (roll_two == roll_three) && (roll_one == roll_three)){
                        // give the player double they bet for getting a three in a row...my game my rules
                        p.add_funds(amount*2);
                        message.setText("Three of a kind!");
                }

                // two of the three match
                else if((roll_one == roll_two) || (roll_two == roll_three) || (roll_one == roll_three)){
                        message.setText("Two of a kind!");
                        p.add_funds(amount);
                }

                // no matches
                else{
                        p.add_funds(amount*-1);
                        message.setText("Loser!");
                }
                // we return an array of rolls because this is used to update the GUI
                return new ArrayList<Integer>(Arrays.asList(roll_one, roll_two, roll_three));
        }
}
```