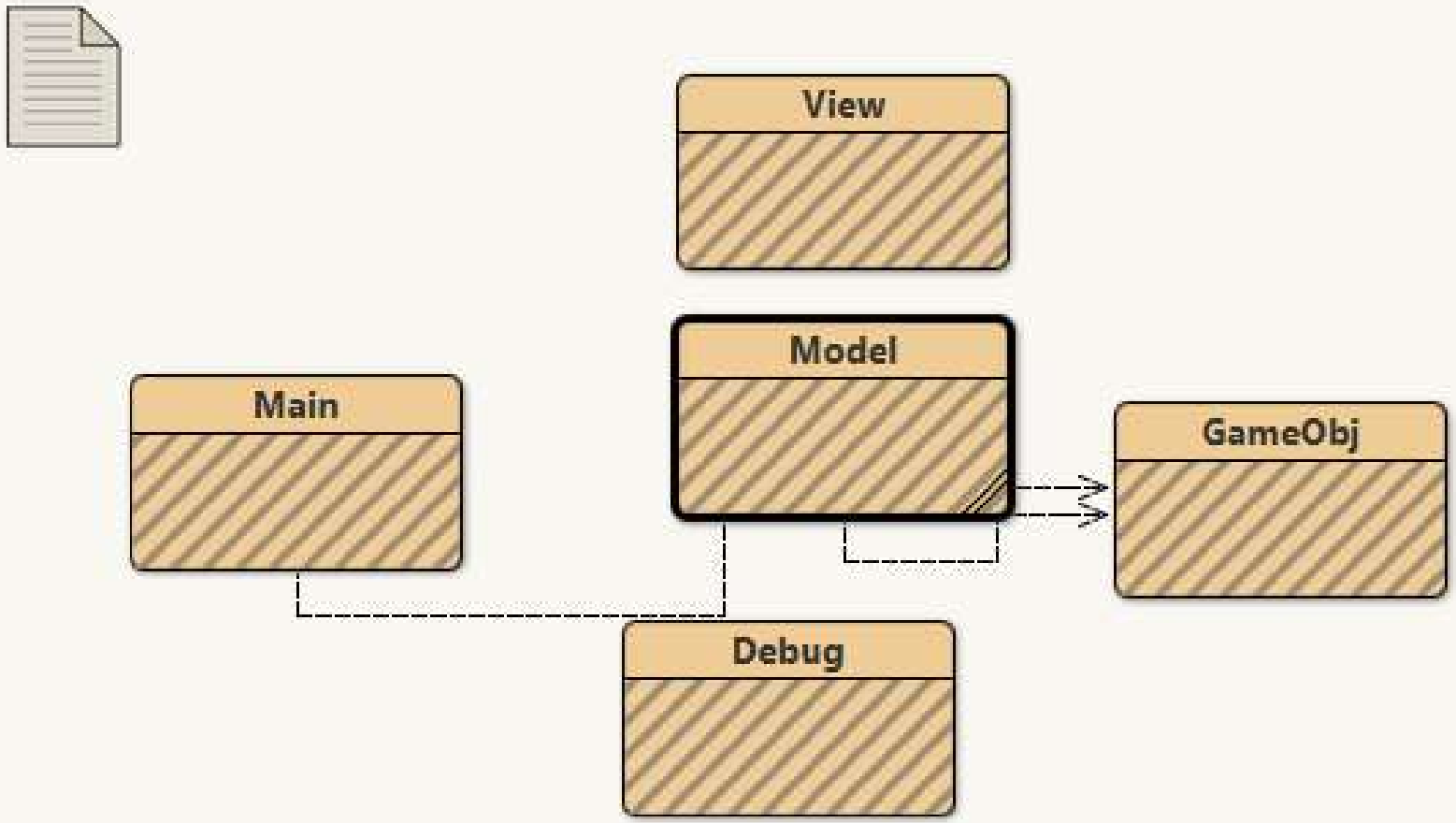
View
Model
Main
GameObj
Debug

```
import java.util.Arrays;
import java.util.Collections;
import java.util.ArrayList;

public class Model{
//This automates the deck building process.
  public static ArrayList<String> BuildDeck(){
    ArrayList<String> deck = new ArrayList<String>();
    String[] suit = {"Spades", "Hearts", "Clubs", "Diamonds"};
    String[] card = {"Ace", "Two", "Three", "Four", "Five", "Six", "Seven",
"Eight","Nine","Ten","Jack", "Queen", "King"};
    //This is a nested for loop that will create a deck of 52 cards and gives them
their name and deck.
    for(int y = 0; y < 4; y++){
      for(int x = 0; x < 13; x++){
        String new_card = card[x] + " of " + suit[y];
        deck.add(new_card);
        }
    }
    return deck;
  }

//This is the shuffle system. It creates an array by calling the BuildDeck()
method, then shuffles it //using the Arrays.shuffle() method.
  public static ArrayList<String> ShuffleDeck(){
    ArrayList<String> deck = BuildDeck();
    Collections.shuffle(deck);
    return deck;
  }

//This is the dealing system, it will simply pop the top card from the deck and
return it as a value
  public static String DealerPick(ArrayList<String> deck){
    String card = deck.remove(0);
    return card;
   }

  public static void dealCards(ArrayList<String>deck, GameObj comp, int numCards){
      //This uses a FOR loop to distribute a hand to each user
      //This deals cars to a newHand depending on how many need to be dealt
      String[] newHand = new String[numCards];
      for(int y = 0; y < numCards; y++){
          comp.updateMyDeck(Model.DealerPick(deck));
      }
  }
  public static GameObj.CPU[] getCPU(){
        GameObj gameObj = new GameObj();
        GameObj.CPU[] comp = new GameObj.CPU[3];
        for(int x = 0; x < 3; x++){
        comp[x] = gameObj.new CPU();
        }
        return comp;
  }
}
```

```
import java.util.Arrays;
import java.util.Collections;
import java.util.ArrayList;


public class GameObj{
  int totalScore = 500;
  int myHandValue;
  int myBet;
 ArrayList<String> myDeck = new ArrayList<String>();
  public void updateValue(int value){
        this.myHandValue = value;
  }
    public void updateScore(int newScore){
        this.totalScore = this.totalScore + newScore;
  }
  public void updateMyDeck(String Hand){
        this.myDeck.add(Hand);
  }
  public void getBet(int MyBet){
        this.myBet = myBet;
  }
  public class CPU extends GameObj{
    //This is the AI for the CPU, the higher the risk and higher the HandValue
    //The less chance of hitting
    public boolean AI(){
      int roll = (int)(Math.random() * 20);
      if (myBet >= (totalScore*0.7) && myHandValue > 17){
        if(roll > 15){
            return true;
        }else{
            return false;
        }
      }
      if (myBet >= (totalScore*0.5) && myHandValue < 17){
        if(roll > 10){
            return true;
        }else{
            return false;
        }
      }else{
          return roll >= myHandValue;
      }
    }

    public int Bet(){
      int roll1 = totalScore;
      int roll2 = totalScore;
      if(totalScore > 100){
        roll1 = (int)(Math.random() * (totalScore * 0.9));
        roll2 = (int)(Math.random() * (totalScore * 0.9));
        }
      if(totalScore < 100){
        roll1 = (int)(Math.random() * (totalScore * 0.75));
```

```
        roll2 = (int)(Math.random() * (totalScore * 0.75));
        }
      if(totalScore < 20){
        roll1 = (int)(Math.random() * (totalScore));
        roll2 = (int)(Math.random() * (totalScore));
        }
      myBet = Math.min(roll1,roll2);
      return myBet;
    }
  }
  public class Player extends GameObj{

  }

}
```

```java
import java.util.Arrays;
import java.util.Collections;
import java.util.ArrayList;

public class Main{
    public static void main(String[] args){
        //Start game through initialising all objects
        GameObj gameObj = new GameObj();
        GameObj.CPU cpu1 = gameObj.new CPU();
        GameObj.CPU cpu2 = gameObj.new CPU();
        GameObj.CPU cpu3 = gameObj.new CPU();
        GameObj.Player player = gameObj.new Player();
        System.out.println(cpu1.Bet());
        System.out.println(cpu1.AI());
        System.out.println(cpu2.Bet());
        System.out.println(cpu2.AI());
        System.out.println(cpu3.Bet());
        System.out.println(cpu3.AI());
    }
}
```

```
import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.*;
import javafx.scene.text.*;
import javafx.stage.Stage;

public class View
{
    // instance variables - replace the example below with your own
    public GridPane grid;
    public Label label1;
    public Button button1;

    /**
     * Constructor for objects of class View
     */
    public void Enter(Stage window)
    {
        // initialise instance variables
        window.setTitle("Start Game");
        grid = new GridPane();

        label1 = new Label("Please input your account name");
        grid.add(label1,0,0);

        button1 = new Button("Click Here:");
        grid.add(button1,0,1);
        button1.setOnAction(this::button1Click);
        window.setScene(new Scene(grid,300,250));
        window.show();
    }

    /**
     * An example of a method - replace this comment with your own
     *
     * @param  y  a sample parameter for a method
     * @return    the sum of x and y
     */
    public void button1Click(ActionEvent event)
    {
        // put your code here
        label1.setText("Hello World!");
    }
}
```

```
import java.util.Arrays;
import java.util.Collections;
import java.util.ArrayList;


public class Debug{
    public static void main(String[] args){
        //My First test
     //This was before I had made a populator and an array of objects,
     //I was testing to see if my object code worked
       // GameObj.CPU cpu1 = gameObj.new CPU();
       // GameObj.CPU cpu2 = gameObj.new CPU();
       // GameObj.CPU cpu3 = gameObj.new CPU();
       // GameObj.Player player = gameObj.new Player();
       // System.out.println(cpu1.Bet());
      //System.out.println(cpu1.AI());
       // System.out.println(cpu2.Bet());
        //System.out.println(cpu2.AI());
        //System.out.println(cpu3.Bet());
       // System.out.println(cpu3.AI());

      //My second test sees if the cards will be distributed and
      //whether the deck of cards will get smaller
      //ArrayList<String> deck = Model.BuildDeck();
      //GameObj gameObj = new GameObj();
      //GameObj.CPU[] comp = Model.getCPU();
      //Model.dealCards(deck, comp[0],3);
      //System.out.println(comp[0].myDeck);
      //System.out.println(deck.size());
    }
}
```

```
------------------------------------------------------------------------
This is the project README file. Here, you should describe your project.
Tell the reader (someone who does not know anything about this project)
all they need to know. The comments should usually include at least:
------------------------------------------------------------------------

PROJECT TITLE:
PURPOSE OF PROJECT:
VERSION or DATE:
HOW TO START THIS PROJECT:
AUTHORS:
USER INSTRUCTIONS:
```