

	<b>VICERRECTORADO DOCENTE</b>	<b>Código:</b> GUIA-PRL-001
	CONSEJO ACADÉMICO	<b>Aprobación:</b> 2016/04/06
<b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

		<b>FORMATO DE INFORME DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA ESTUDIANTES</b>	
<b>CARRERA:</b> COMPUTACIÓN		<b>ASIGNATURA:</b>	
<b>NRO. PRÁCTICA:</b>		<b>TÍTULO PRÁCTICA:</b> Examen Final	
<b>OBJETIVO ALCANZADOS:</b> <ul style="list-style-type: none"> <li>• Identificar los cambios importantes de Java</li> <li>• Diseñar e Implementar expresiones regulares</li> <li>• Entender la cada uno de las características nuevas en Java</li> <li>• Reforzar los conocimientos adquiridos en clase sobre la programación aplicada (POO, Interfaz grafica, Base de datos, Hilos) en un contexto real.</li> </ul>			
<b>ACTIVIDADES DESARROLLADAS</b>			
<b>Enunciado</b> <p>Se desea simular los posibles beneficios de diversas estrategias de juego en un casino. La ruleta francesa es un juego en el que hay una ruleta con 37 números (del 0 al 36). Cada 2000 (tiempo parametrizable) milisegundos el croupier saca un número al azar y los diversos hilos clientes apuestan para ver si ganan. Todos los hilos empiezan con 1.000 euros y la banca (que controla la ruleta) con 50.000. Cuando los jugadores pierden dinero, la banca incrementa su saldo.</p> <ul style="list-style-type: none"> <li>• Se puede jugar a un número concreto. Habrá 4 hilos clientes que eligen números al azar del 1 al 36 (no el 0) y restarán 10 euros de su saldo para apostar a ese ese número. Si sale su número su saldo se incrementa en 360 euros (36 veces lo apostado).</li> <li>• Se puede jugar a par/impar. Habrá 4 hilos clientes que eligen al azar si apuestan a que saldrá un número par o un número impar. Siempre restan 10 euros para apostar y si ganan incrementan su saldo en 20 euros.</li> <li>• Se puede jugar a la «martingala». Habrá 4 hilos que eligen números al azar. Elegirán un número y empezarán restando 10 euros de su saldo para apostar a ese número. Si ganan incrementan su saldo en 360 euros. Si pierden jugarán el doble de su apuesta anterior (es decir, 20, luego 40, luego 80, y así sucesivamente)</li> <li>• La banca acepta todas las apuestas, pero nunca paga más dinero del que tiene.</li> <li>• Si sale el 0, todo el mundo pierde y la banca se queda con todo el dinero.</li> </ul> <p>Adicionalmente, se deberá generar un sistema de base de datos con JPA en donde puede gestionar a los clientes o hilos jugadores, con cada una de las apuestas realizadas, los valores que se están manejando tanto de la banca como de cada cliente y gestionar la simulación es decir se puede iniciar y parar en cualquier intervalo de tiempo en la simulación, además de poder cambiar a cualquier cliente con un nuevo o un anterior y en que modalidad va a jugar. Por otro lado, es parametrizable el tiempo que se demora dar la vuelta a la ruleta con el proceso de apuesta.</p> <p>Es importante destacar que debe existir un sistema de simulación visual y un sistema de gestión de jugadores, transacciones y apuestas en donde se evidencia la apuesta, el jugador, la ruleta el numero generado y como varían los saldos de los que intervienen dentro del juego.</p>			
<b>1. DEFINICIÓN DEL PROBLEMA:</b>			

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

Para la solución del sistema, se implementara hilos encardados de gestionar las apuestas una vez finalizada el movimiento de la ruleta.

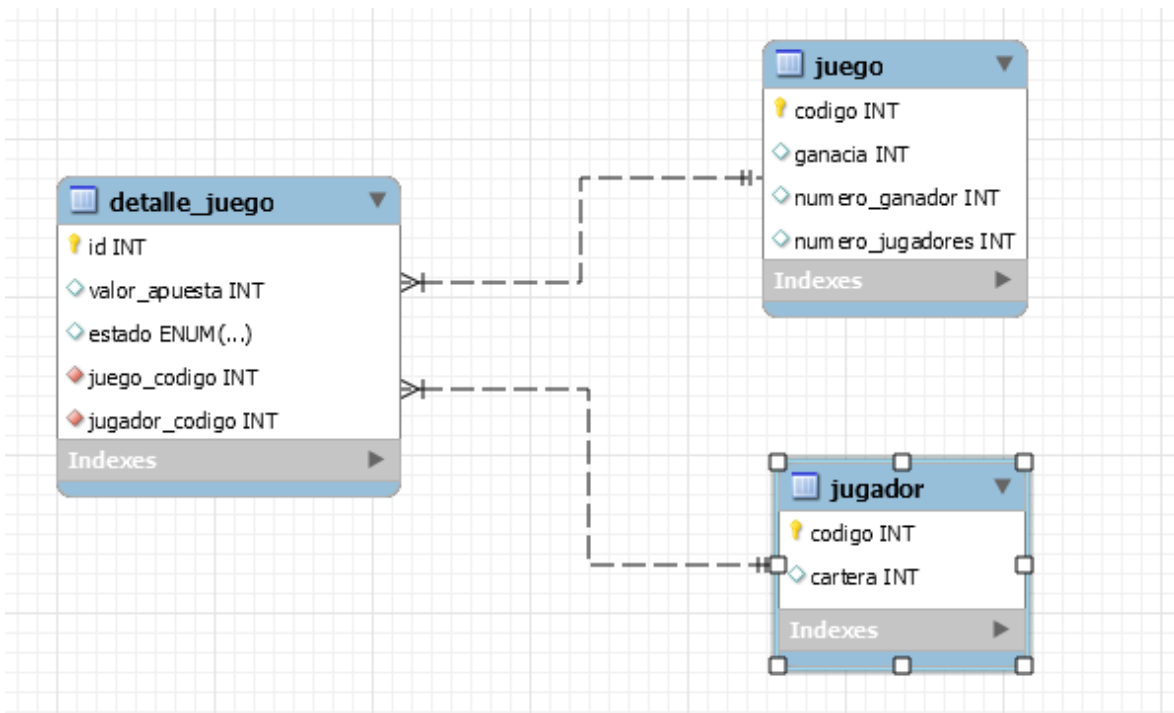
### OBJETIVOS DEL SISTEMA:

El sistema debe de ser desarrollado en el modo monousuario para que pueda:

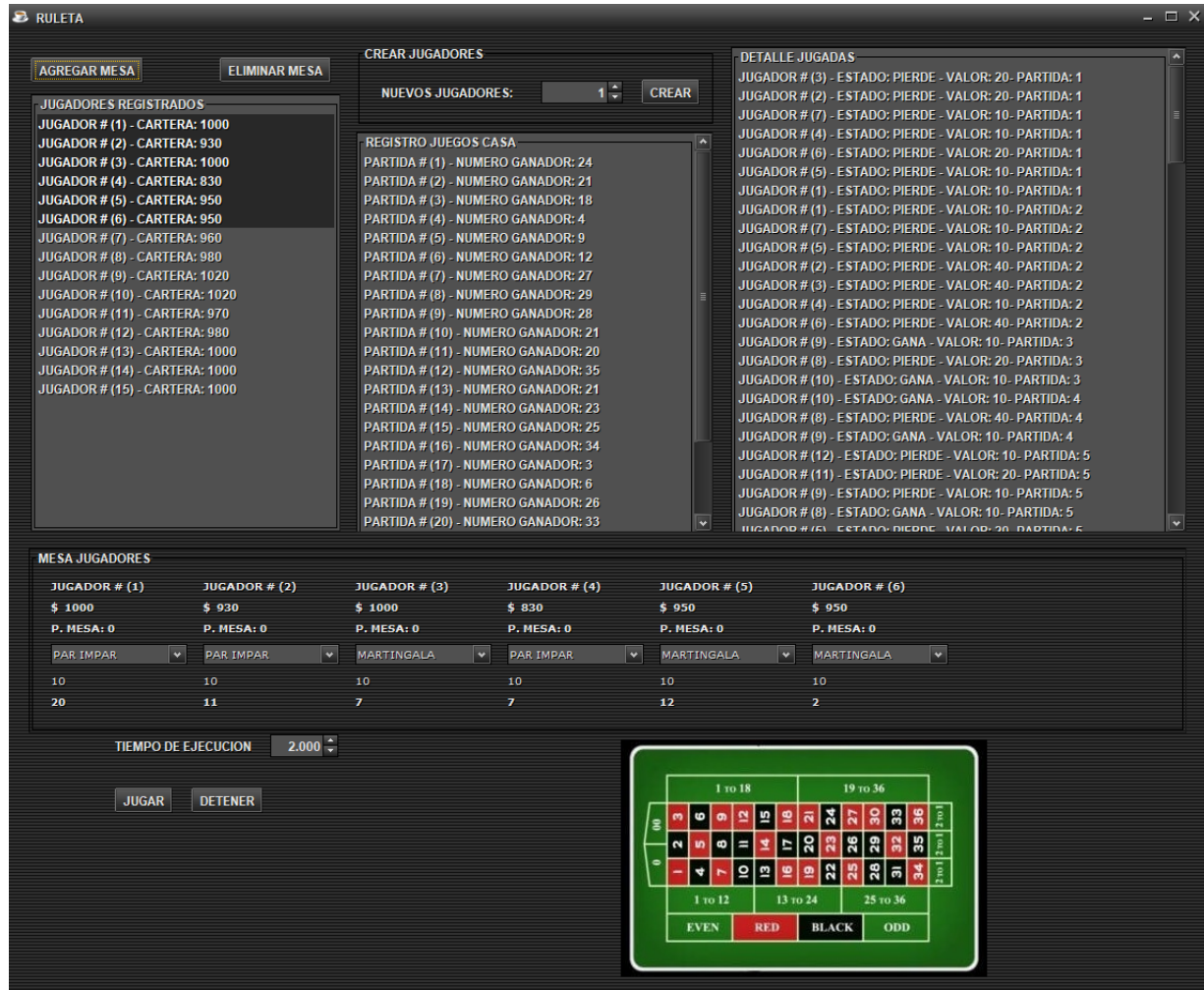
- 1.3.1. Establecer un sistema de cálculo de jugadas
- 1.3.2. Establecer una ventana de registros.

### RESULTADO(S) OBTENIDO(S):

#### ESQUEMA BASE DE DATOS



#### VISTA ADMINISTRADOR



Se implemento tres listas donde tenemos la información de los jugadores y de los reportes. Cada que se juega una partida los hilos actualizan esta vista para que se pueda visualizar.

### CLASES HILOS:

Se utilizo dos Hilos.


Uno Llamado Jugador y otro Mesa.

### HILO JUGADOR.

Este hilo se encarga de analizar las apuestas y de cobrar o pagar, además de actualizar las vistas.

```
package ec.edu.ups.modelo;

import ec.edu.ups.controlador.DetalleJuegoJpaController;
import ec.edu.ups.controlador.JugadorJpaController;
import ec.edu.ups.controlador.exceptions.NonexistentEntityException;
import ec.edu.ups.utils.Utils;
import ec.edu.ups.vista.VistaPrincipal;
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.persistence.EntityManagerFactory;
import javax.swing.JLabel;


/**
 *
 * @author Paul Idrovo
 */
public class HiloJugador extends Thread {

    private int codigoJugador;
    private int carteraJugador;
    private int posicion;
    private String modoJuego;
    private int resultado;
    private List<JLabel> labelValorApuesta;
    private List<JLabel> labelCarteraJugador;
    private List<JLabel> labelCodigoJugador;
    private List<JLabel> labelContadorJugadas;
    private List<JLabel> labelNumeroApostar;
    private JugadorJpaController controladorJugador;
    private EntityManagerFactory emf;
    private int codigoPartida;
    private DetalleJuegoJpaController controladorDetalleJuego;

    public HiloJugador(int codigoPartida, int resultado, int codigoJugador, int
carteraJugador, int posicion, String modoJuego, List<JLabel> labelValorApuesta, List<JLabel>
labelCarteraJugador,
        List<JLabel> labelCodigoJugador, List<JLabel> labelContadorJugadas, List<JLabel>
labelNumeroApostar) {
        emf = Utils.getEntityEntityManagerFactory();
        this.codigoPartida = codigoPartida;
        this.controladorJugador = new JugadorJpaController(emf);
        this.codigoJugador = codigoJugador;
        this.carteraJugador = carteraJugador;
        this.posicion = posicion;
        this.modoJuego = modoJuego;
        this.labelValorApuesta = labelValorApuesta;
        this.labelCarteraJugador = labelCarteraJugador;
        this.labelCodigoJugador = labelCodigoJugador;
        this.labelContadorJugadas = labelContadorJugadas;
        this.labelNumeroApostar = labelNumeroApostar;
        this.resultado = resultado;
        this.controladorDetalleJuego = new DetalleJuegoJpaController(emf);
        if (!modoJuego.equals("MARTINGALA")) {
            apostarNumero();
        }
    }

    @Override
    public void run() {
        int apuestaNum = Integer.parseInt(labelNumeroApostar.get(posicion).getText());

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```


int valorApuesta = Integer.parseInt(labelValorApuesta.get(posicion).getText());
String estado = "";
if (carteraJugador > 0) {

    switch (modoJuego) {
        case "NUM. CONCRETO":
            if (apuestaNum == resultado) {
                carteraJugador += 360;
                System.out.println("GANA JUGADOR " + codigoJugador);
                estado = "GANA";
            } else {
                carteraJugador -= valorApuesta;
                System.out.println("PIERDE JUGADOR " + codigoJugador);
                estado = "PIERDE";
            }
            break;
        case "PAR IMPAR":
            if (apuestaNum % 2 == resultado % 2) {
                carteraJugador += 20;
                System.out.println("GANA JUGADOR " + codigoJugador);
                estado = "GANA";
            } else {
                carteraJugador -= valorApuesta;
                System.out.println("PIERDE JUGADOR " + codigoJugador);
                estado = "PIERDE";
            }
            break;
        case "MARTINGALA":
            if (apuestaNum == resultado) {
                carteraJugador += 360;
                System.out.println("GANA JUGADOR " + codigoJugador);
                estado = "GANA";
            } else {
                carteraJugador -= valorApuesta;
                valorApuesta = valorApuesta * 2;
                labelValorApuesta.get(posicion).setText(valorApuesta + "");
                System.out.println("PIERDE JUGADOR " + codigoJugador);
                estado = "PIERDE";
            }
            break;
    }

    Jugador actualizarJugador = new Jugador(codigoJugador);
    actualizarJugador.setCartera(carteraJugador);
    actualizarJugador.setDetalleJuegoList(null);
    DetalleJuego nueDetalleJuego = new DetalleJuego();
    nueDetalleJuego.setEstado(estado);
    nueDetalleJuego.setJuegoCodigo(new Juego(codigoPartida));
    nueDetalleJuego.setJugadorCodigo(new Jugador(codigoJugador));

    nueDetalleJuego.setValorApuesta(Integer.parseInt(labelValorApuesta.get(posicion).getText()));
    controladorDetalleJuego.create(nueDetalleJuego);
    try {
        controladorJugador.edit(actualizarJugador);
    }
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

    } catch (NonexistentEntityException ex) {
        Logger.getLogger(HiloJugador.class.getName()).log(Level.SEVERE, null, ex);
    } catch (Exception ex) {
        Logger.getLogger(HiloJugador.class.getName()).log(Level.SEVERE, null, ex);
    }

    labelCarteraJugador.get(posicion).setText("$ " + carteraJugador);
}

public void apostarNumero() {
    int numeroApuesta = (int) (Math.random() * 35) + 1;
    labelNumeroApostar.get(posicion).setText(numeroApuesta + "");
    System.out.println("Jugador " + codigoJugador + " numero " + numeroApuesta);
}
}

```

## HILO MESA

Se encarga de hacer que los Hilos inicien, además de generar el numero ganador.

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ec.edu.ups.modelo;

import ec.edu.ups.controlador.DetalleJuegoJpaController;
import ec.edu.ups.controlador.JuegoJpaController;
import ec.edu.ups.utils.Utils;
import ec.edu.ups.vista.VistaPrincipal;
import java.util.ArrayList;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.persistence.EntityManagerFactory;
import javax.swing.JComboBox;
import javax.swing.JLabel;

/**
 *
 * @author Paul Idrovo
 */
public class HiloMesa extends Thread {


    private int tiempoEjecucion;
    private int resultado;
    private DetalleJuegoJpaController controladorDetalleJuego;
    private JuegoJpaController controladorJuego;
    private List<Thread> listaJugadoresHilos;
    private List<JLabel> labelValorApuesta;
    private List<JLabel> labelCarteraJugador;
    private List<JLabel> labelCodigoJugador;
    private List<JLabel> labelContadorJugadas;

```

```
private List<JLabel> labelNumeroApostar;
private List<JComboBox> comboBoxModalidad;
private EntityManagerFactory emf;
private VistaPrincipal vista;

public HiloMesa(int tiempoEjecucion, List<JLabel> labelValorApuesta, List<JLabel>
labelCarteraJugador, List<JLabel> labelCodigoJugador, List<JLabel> labelContadorJugadas,
List<JLabel> labelNumeroApostar, List<JComboBox> comboBoxModalidad, VistaPrincipal
vista) {
    this.tiempoEjecucion = tiempoEjecucion;
    this.labelValorApuesta = labelValorApuesta;
    this.labelCarteraJugador = labelCarteraJugador;
    this.labelCodigoJugador = labelCodigoJugador;
    this.labelContadorJugadas = labelContadorJugadas;
    this.labelNumeroApostar = labelNumeroApostar;
    this.comboBoxModalidad = comboBoxModalidad;
    emf = Utils.getEntityManagerFactory();
    this.controladorDetalleJuego = new DetalleJuegoJpaController(emf);
    this.controladorJuego = new JuegoJpaController(emf);
    this.vista = vista;
}

@Override
public void run() {
    while (VistaPrincipal.ejecucion) {
        try {
            resultado = (int) (Math.random() * 36);
            System.out.println("NUMERO GANADOR " + resultado);
            listaJugadoresHilos = new ArrayList<>();
            Juego nuevaPartida = new Juego();
            nuevaPartida.setNumeroGanador(resultado);
            nuevaPartida.setNumeroJugadores(labelCarteraJugador.size());
            controladorJuego.create(nuevaPartida);
            int codigoPartida = controladorJuego.getJuegoCount();
            for (int i = 0; i < labelCarteraJugador.size(); i++) {
                String codigo = labelCodigoJugador.get(i).getText();
                String carter = labelCarteraJugador.get(i).getText();
                int codigoJugador = Integer.parseInt(codigo.substring(codigo.indexOf("(")
+ 1, codigo.indexOf(")")));
                int carterJugador = Integer.parseInt(carter.substring(3));
                listaJugadoresHilos.add(new Thread(new
HiloJugador(codigoPartida, resultado, codigoJugador, carterJugador, i,
comboBoxModalidad.get(i).getSelectedItem().toString(),
labelValorApuesta, labelCarteraJugador, labelCodigoJugador,
labelContadorJugadas, labelNumeroApostar)));
            }
            Thread.sleep(5000);
            for (Thread listaJugadoresHilo : listaJugadoresHilos) {
                listaJugadoresHilo.start();
            }
            Thread.sleep(tiempoEjecucion);
            vista.cargarListaJugadores();
            vista.cargarPartidasHistorial();
            vista.cargarDatosPartida();
        }
    }
}
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```

    } catch (InterruptedException ex) {
        Logger.getLogger(HiloMesa.class.getName()).log(Level.SEVERE, null, ex);
    }

}

}

```

#### LINKS

Se adjunta los links de presentación y del repositorio:

<https://github.com/psidrovo/ExamenFinal.git>

#### CONCLUSIONES:

En este proyecto se ha podido aplicar todo lo revisado en las clases, como son las nuevas funciones de java, patrones de diseños, los hilos y base de datos para almacenar información. Además de poder implementar nuevas librerías que no se han utilizado en clase, permitiéndonos generar una investigación para las mismas, abriéndonos a nuevas posibilidades de poder mejorar el servicio al usuario.

**Nombre del estudiante:** Paul Sebastian Idrovo

**Firma:**

