

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

		PRÁCTICA DE LABORATORIO	
CARRERA: COMPUTACION		ASIGNATURA: Programación Aplicada	
NRO. PRÁCTICA:	1	TÍTULO PRÁCTICA: Examen Final.	
OBJETIVO ALCANZADO: <ul style="list-style-type: none"> Consolidar los conocimientos adquiridos en clase sobre Java 			
ACTIVIDADES DESARROLLADAS			
<p>1. Realizar una aplicación con el siguiente enunciado.</p> <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"> ⑩ 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). ⑩ 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. ⑩ 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) ⑩ La banca acepta todas las apuestas pero nunca paga más dinero del que tiene. ⑩ Si sale el 0, todo el mundo pierde y la banca se queda con todo el dinero. <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> <p>Por último se debe presentar dos reportes o tablas de los datos:</p> <ol style="list-style-type: none"> Clientes y la banca con el número de transacciones o apuestas realizadas, el valor de total y cuantas ha perdido y cuantas veces ha ganado además de la modalidad de juego. Dentro de cada cliente se puede acceder al historial de apuestas y transacciones realizadas. 			

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

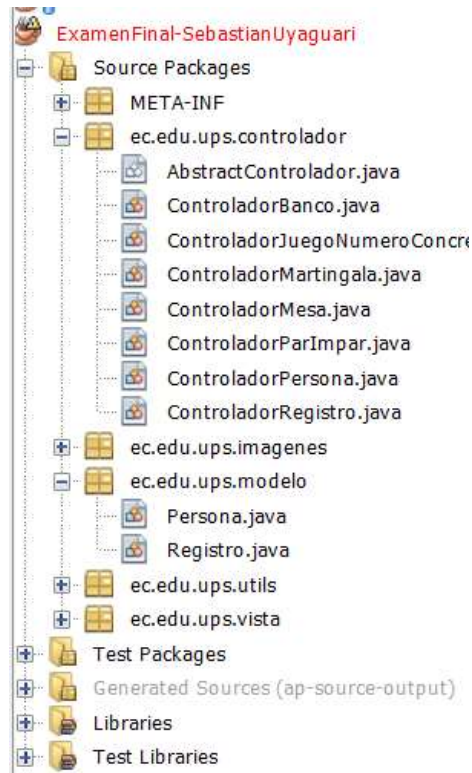
2. Diagrama de clases

Adjunto el link de el diagrama de clases creado en lucid Chart.

<https://lucid.app/lucidchart/invitations/accept/e42e3d12-daf3-45e3-9659-256ac7bf8cbf>

3. Aplicación:

Se creó una aplicación con la arquitectura MVC y con las librerías incluidas para:




En la clase modelo se creó con Entity Class para la gestión con la base de datos, lo mismo se hizo con la clase registro, para registrar en la base de datos se creó un Abstract controlador, el cual fue implementado con reflexión para que sirva como genérica para las demás clases que están siendo extendidas por la misma.

Para resolver el problema de los hilos se implementó una clase que sirva como hilo principal e ir creando los hilos de los clientes al momento que se inicie, además los juegos fueron implementados en clases diferentes para no tener ningún problema al momento de ejecutar y por ultimo un controlador del banco el cual administrara las ganancias de los jugadores y registrara en la base de datos.

Adjunto código de las clases:

Clase Persona::

```
@Entity
public class Persona implements Serializable {
```

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

```

private static final long serialVersionUID = 1L;
@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
@Column(name = "persona_id")
private Long id;

@Column
private String cedula;

@Column
private String nombre;

@Column
private String apellido;

@Column
private double cuenta;

@OneToMany(mappedBy = "persona", cascade = CascadeType.ALL)
private List<Registro> registro;

public Long getId() {
    return id;
}

public void setId(Long id) {
    this.id = id;
}

public String getCedula() {
    return cedula;
}

public void setCedula(String cedula) {
    this.cedula = cedula;
}


public String getNombre() {
    return nombre;
}

public void setNombre(String nombre) {
    this.nombre = nombre;
}

public String getApellido() {
    return apellido;
}

public void setApellido(String apellido) {
    this.apellido = apellido;
}

```

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

```

public double getCuenta() {
    return cuenta;
}

public void setCuenta(double cuenta) {
    this.cuenta = cuenta;
}

public List<Registro> getRegistro() {
    return registro;
}

public void setRegistro(List<Registro> registro) {
    this.registro = registro;
}

@Override
public int hashCode() {
    int hash = 0;
    hash += (id != null ? id.hashCode() : 0);
    return hash;
}

@Override
public boolean equals(Object object) {
    // TODO: Warning - this method won't work in the case the id fields are not
set
    if (!(object instanceof Persona)) {
        return false;
    }
    Persona other = (Persona) object;
    if ((this.id == null && other.id != null) || (this.id != null &&
!this.id.equals(other.id))) {
        return false;
    }
    return true;
}

@Override
public String toString() {
    return "Persona{" + "id=" + id + "}";
}
}

```


Clase Registro:

```

@Entity
public class Registro implements Serializable {

    private static final long serialVersionUID = 1L;
    @Id

```

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

```

@GeneratedValue(strategy = GenerationType.IDENTITY)
private Long id;

@Column
private int numero;

@Column
private int valor;

@Column
private String tipo;

@Column
private String modalidad;

@ManyToOne
@JoinColumn(name = "fk_persona")
private Persona persona;

public Registro() {
}

public Registro(int numero, int valor, String tipo, String modalidad, Persona p)
{
    this.numero = numero;
    this.valor = valor;
    this.tipo = tipo;
    this.modalidad = modalidad;
    this.persona = p;
}

public Long getId() {
    return id;
}

public void setId(Long id) {
    this.id = id;
}


public int getNumero() {
    return numero;
}

public void setNumero(int numero) {
    this.numero = numero;
}

public int getValor() {
    return valor;
}

public void setValor(int valor) {
    this.valor = valor;
}

```

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

```

public String getTipo() {
    return tipo;
}

public void setTipo(String tipo) {
    this.tipo = tipo;
}

public String getModalidad() {
    return modalidad;
}

public void setModalidad(String modalidad) {
    this.modalidad = modalidad;
}

public Persona getPersona() {
    return persona;
}

public void setPersona(Persona persona) {
    this.persona = persona;
}

@Override
public int hashCode() {
    int hash = 0;
    hash += (id != null ? id.hashCode() : 0);
    return hash;
}

@Override
public boolean equals(Object object) {
    // TODO: Warning - this method won't work in the case the id fields are not
set
    if (!(object instanceof Registro)) {
        return false;
    }
    Registro other = (Registro) object;
    if ((this.id == null && other.id != null) || (this.id != null &&
!this.id.equals(other.id))) {
        return false;
    }
    return true;
}

@Override
public String toString() {
    return "Registro{" + "id=" + id + ", numero=" + numero + ", valor=" + valor
+ ", tipo=" + tipo + ", modalidad=" + modalidad + ", persona=" + persona + '}';
}

```

}

Codigo ControladorMesa:

```
public class ControladorMesa extends Thread{

    HashMap<Integer, ControladorJuegoNumeroConcreto> lista;
    HashMap<Integer, ControladorParImpar> listaPar;
    HashMap<Integer, ControladorMartingala> listaMartingala;
    JTextField numeroLabel;
    Thread thread;
    private Random random;

    private boolean cent;
    private int a;

    public ControladorMesa(HashMap lista, JTextField numero, int a) {

        this.numeroLabel= numero;

        this.a = a;

        if(a==1){

            this.lista = lista;
        }
        if(a==2){
            this.listaPar= lista;
        }
        if(a==3){
            this.listaMartingala= lista;
        }
        random = new Random();

        thread = new Thread(this);

        thread.start();

    }

    @Override
    public void run() {
        cent = true;
        System.out.println("a =" +a);
        while(cent){

            if(a==1){
                try {
                    int numero = random.nextInt(37);
                    numeroLabel.setText(numero+"");
                    for (Map.Entry<Integer, ControladorJuegoNumeroConcreto> entry :
lista.entrySet()) {
                        ControladorJuegoNumeroConcreto value = entry.getValue();
```

```

        value.setNumeroApuesta(numero);

        Thread t = new Thread(value);

        t.start();
    }
    Thread.sleep(5000);
} catch (InterruptedException ex) {
}
}

if(a==2){
    try {
        int numero = random.nextInt(37);
        numeroLabel.setText(numero+"");
        for (Map.Entry<Integer, ControladorParImpar> entry :
listaPar.entrySet()) {
            ControladorParImpar value = entry.getValue();

            value.setNumeroApuesta(numero);

            Thread t = new Thread(value);

            t.start();
        }
        Thread.sleep(5000);
    } catch (InterruptedException ex) {
    }
}


if(a==3){
    try {
        int numero = random.nextInt(37);
        numeroLabel.setText(numero+"");
        for (Map.Entry<Integer, ControladorMartingala> entry :
listaMartingala.entrySet()) {
            ControladorMartingala value = entry.getValue();

            value.setNumeroApuesta(numero);

            Thread t = new Thread(value);

            t.start();
        }
        Thread.sleep(5000);
    } catch (InterruptedException ex) {
    }
}
}
}

```


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

```

public boolean isCent () {
    return cent;
}

public void setCent(boolean cent) {
    this.cent = cent;
}
}

```

Clase controlador Banco

```

public class ControladorBanco {

    private Persona persona;
    private double dineroBanco;
    ControladorPersona controladorPersona = ControladorPersona.getInstance();

    private static ControladorBanco instance = new ControladorBanco();

    public ControladorBanco () {

        Persona persona = controladorPersona.buscarCedula("0000000000");
        dineroBanco = persona.getCuenta();
        this.persona = persona;

    }

    public static ControladorBanco getInstance () {
        return instance;
    }

    public synchronized int getGananciaConcreto () {

        if(dineroBanco>360){
            retirarApuestos(360);
            return 360;
        }else{
            return 0;
        }
    }

    public synchronized int getGananciaPar () {

        if(dineroBanco > 20){
            retirarApuestos(20);
            return 20;

        }else{
            return 0;
        }
    }

    public synchronized int getGananciaMartingala(int apuesta){
        if(dineroBanco > (apuesta*36)){

```

```
        retirarApuestos((int) (apuesta*36));
        return apuesta*36;
    }else{
        return 0;
    }
}

public double getDineroBanco() {
    return dineroBanco;
}

public void setDineroBanco(double dineroBanco) {
    this.dineroBanco = dineroBanco;
}

public synchronized void incrementarApuesta(int valor){
    List<Registro> registros = persona.getRegistro();
    registros.add(new Registro(0, valor, "Ingreso", "Apuesta", persona));
    persona.setRegistro(registros);
    dineroBanco +=valor;
    persona.setCuenta(dineroBanco);
    controladorPersona.update(persona);
    try {
        Thread.sleep(500);
    } catch (InterruptedException ex) {

    }
}

public synchronized void retirarApuestos(int valor){
    List<Registro> registros = persona.getRegistro();
    registros.add(new Registro(0, valor, "Retirada", "Perdida", persona));
    persona.setRegistro(registros);
    dineroBanco -=valor;
    persona.setCuenta(dineroBanco);
    controladorPersona.update(persona);
    try {
        Thread.sleep(500);
    } catch (InterruptedException ex) {


    }
}
}
```

Controlador Juego Concreto:

```
public class ControladorJuegoNumeroConcreto implements Runnable{

    Persona persona;
    Random random;
    int numeroApuesta;

    ControladorBanco controladorBanco;
    ControladorPersona controladorPersona;
```

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

```

JTextField etiquetaNumero;
JTextArea txtArea;

public ControladorJuegoNumeroConcreto(Persona persona, JTextField n, JTextArea
txtArea) {

    controladorPersona = ControladorPersona.getInstance();
    controladorBanco = ControladorBanco.getInstance();
    this.persona = persona;

    random = new Random();

    etiquetaNumero = n;
    this.txtArea = txtArea;

}


@Override
public void run() {
    if(dineroCuenta()){
        int numeroApuesta = apostar();
        if(getNumeroApuesta()== numeroApuesta){
            cobrarDinero();
        }else{
            txtArea.append("El jugador "+persona.getNombre()+" no gano la
apuesta \n");
        }
    }else{
        txtArea.append("El jugador "+persona.getNombre()+" ya no puede seguir
apostando \n");
    }
}

private boolean dineroCuenta() {
    if(persona.getCuenta() > 10){
        return true;
    }else{
        return false;
    }
}

public int apostar(){
    int numeroA = (random.nextInt(36))+1;
    controladorBanco.incrementarApuesta(10);
    retirarDinero(numeroA);
    txtArea.append("El jugador "+persona.getNombre()+" apuesta al numero
"+numeroA +"\n");
    etiquetaNumero.setText(numeroA+"");
    return numeroA;
}

public void retirarDinero(int numeroApu){
    List<Registro> registros = persona.getRegistro();

```

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

```
registros.add( new Registro( numeroApu, 10, "Retirar", "Apuesta Concreto",
persona));
persona.setRegistro(registros);
double dinero = persona.getCuenta()-10;
persona.setCuenta(dinero);
try{
    System.out.println(persona);
    controladorPersona.update(persona);

}catch(NullPointerException e){

}

}

public int getNumeroApuesta() {
    return numeroApuesta;
}

public void setNumeroApuesta(int numeroApuesta) {
    this.numeroApuesta = numeroApuesta;
}

public void cobrarDinero(){
    int valor = controladorBanco.getGananciaConcreto();
    List<Registro> registros = persona.getRegistro();
    registros.add(new Registro(numeroApuesta, valor, "ingreso", "Gano Concreto",
persona));
    persona.setRegistro(registros);
    double dinero = persona.getCuenta()+valor;
    persona.setCuenta(dinero);
    controladorPersona.update(persona);
    txtArea.append("El jugador "+persona.getNombre()+" gana la apuesta \n");

}

}
```

Controlador Par Impar

```
public class ControladorParImpar implements Runnable{

    ControladorBanco controladorBanco;
    ControladorPersona controladorPersona;

    int numeroApuesta;

    Persona persona;
    JTextField numero;
    JTextArea txtArea;

    private Random random;

    public ControladorParImpar(Persona persona, JTextField numero, JTextArea
txtArea) {
```

```
controladorBanco = ControladorBanco.getInstance();
controladorPersona = ControladorPersona.getInstance();

random = new Random();

this.persona = persona;
this.numero = numero;
this.txtArea = txtArea;


}

@Override
public void run() {
    if(dineroCuenta()){
        if((numeroApuesta%2==0 && apostar()==0) || (numeroApuesta%2==1 &&
apostar()==1)){
            cobrarDinero();
        }else{
            txtArea.append("El jugador "+persona.getNombre()+" no gana la
apuesta \n");
        }
    }else{
        txtArea.append("El jugador "+persona.getNombre()+" ya no puede seguir
apostando \n");
    }
}

private boolean dineroCuenta() {
    if(persona.getCuenta() > 10){
        return true;
    }else{
        return false;
    }
}

public int apostar(){
    int apuesta = random.nextInt(2);
    controladorBanco.incrementarApuesta(10);
    retirarDinero(10);
    if(apuesta==0){
        txtArea.append("El jugador "+persona.getNombre()+" apuesta al numero Par
\n");
        numero.setText("Par");
    }else{
        txtArea.append("El jugador "+persona.getNombre()+" apuesta al numero
Impar \n");
        numero.setText("Impar");
    }
    return apuesta;
}

public void retirarDinero(int numeroApu){
```

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

```

List<Registro> registros = persona.getRegistro();
registros.add( new Registro( numeroApu, 10, "Retirar", "Apuesta Par Impar",
persona));
persona.setRegistro(registros);
double dinero = persona.getCuenta()-10;
persona.setCuenta(dinero);
try{
    System.out.println(persona);
    controladorPersona.update(persona);

}catch(NullPointerException e){

}

}

public void cobrarDinero(){
    int valor = controladorBanco.getGananciaPar();
    List<Registro> registros = persona.getRegistro();
    registros.add(new Registro(numeroApuesta, valor, "ingreso", "Gano Par
Impar", persona));
    persona.setRegistro(registros);
    double dinero = persona.getCuenta()+valor;
    persona.setCuenta(dinero);
    controladorPersona.update(persona);
    txtArea.append("El jugador "+persona.getNombre()+" gana la apuesta \n");

}

public int getNumeroApuesta() {
    return numeroApuesta;
}

public void setNumeroApuesta(int numeroApuesta) {
    this.numeroApuesta = numeroApuesta;
}

}

```

Controlador Martingala

```

public class ControladorMartingala implements Runnable{


    ControladorBanco controladorBanco;
    ControladorPersona controladorPersona;

    int numeroApuesta;
    int montoApostar;

    Persona persona;
    JTextField numero;
    JTextArea txtArea;

    Random random;
}

```

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

```

public ControladorMartingala(Persona persona, JTextField numero, JTextArea
txtArea) {

    controladorBanco = ControladorBanco.getInstance();
    controladorPersona = ControladorPersona.getInstance();

    random = new Random();

    this.persona = persona;
    this.numero = numero;
    this.txtArea = txtArea;

    montoApostar= 10;


}

@Override
public void run() {
    if(dineroCuenta()){
        int apuesta = numeroApostar();
        if(apuesta==getNumeroApuesta()){
            cobrarDinero();
            montoApostar=10;
        }else{
            montoApostar = montoApostar*2;
            txtArea.append("El jugador "+persona.getNombre()+" no gano la
apuesta \n");
        }
    }else{
        txtArea.append("El jugador "+persona.getNombre()+" ya no puede seguir
apostando \n");
    }
}

private boolean dineroCuenta() {
    if(persona.getCuenta() > montoApostar){
        return true;
    }else{
        return false;
    }
}

private int numeroApostar() {
    int numeroA = (random.nextInt(36))+1;
    controladorBanco.incrementarApuesta(montoApostar);
    retirarDinero(numeroA);
    txtArea.append("El jugador "+persona.getNombre()+" apuesta al numero
"+numeroA +
        "con el valor de "+montoApostar+"\n");
    numero.setText(numeroA+"");
    return numeroA;
}

```

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

```

public void retirarDinero(int numeroApu){
    List<Registro> registros = persona.getRegistro();
    registros.add( new Registro( numeroApu, montoApostar, "Retirar", "Apuesta
Martingala", persona));
    persona.setRegistro(registros);
    double dinero = persona.getCuenta()-montoApostar;
    persona.setCuenta(dinero);
    try{
        System.out.println(persona);
        controladorPersona.update(persona);

    }catch(NullPointerException e){

    }

}

public void cobrarDinero(){
    int valor = controladorBanco.getGananciaMartingala(montoApostar);
    List<Registro> registros = persona.getRegistro();
    registros.add(new Registro(numeroApuesta, valor, "ingreso", "Gano
Martingala", persona));
    persona.setRegistro(registros);
    double dinero = persona.getCuenta()+valor;
    persona.setCuenta(dinero);
    controladorPersona.update(persona);
    txtArea.append("El jugador "+persona.getNombre()+" gana la apuesta \n");


}

public int getNumeroApuesta() {
    return numeroApuesta;
}

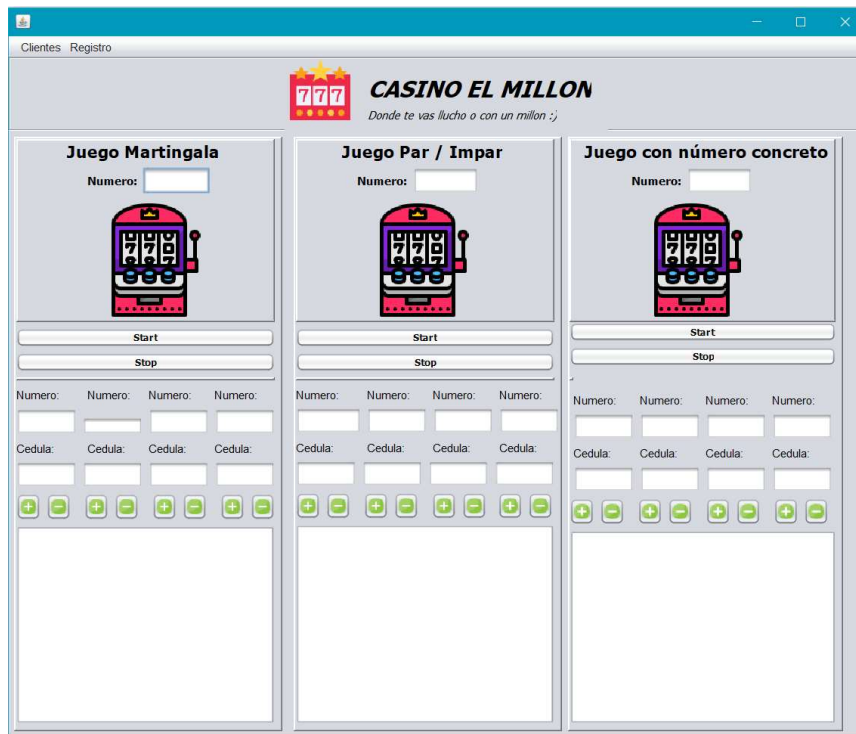
public void setNumeroApuesta(int numeroApuesta) {
    this.numeroApuesta = numeroApuesta;
}

}

```


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

Una vez completado todo el código se procede a realizar la interfaz y como resultado se obtuvo:



The screenshot shows a web application titled 'CASINO EL MILLON' with the tagline 'Donde te vas lluchó o con un millón :j'. The interface is divided into three main game sections:

- Juego Martingala:** Features a 'Numero:' input field, a slot machine icon, 'Start' and 'Stop' buttons, and a grid of four 'Numero:' and four 'Cedula:' input fields.
- Juego Par / Impar:** Features a 'Numero:' input field, a slot machine icon, 'Start' and 'Stop' buttons, and a grid of four 'Numero:' and four 'Cedula:' input fields.
- Juego con número concreto:** Features a 'Numero:' input field, a slot machine icon, 'Start' and 'Stop' buttons, and a grid of four 'Numero:' and four 'Cedula:' input fields.

Each game section also includes a large empty box at the bottom, likely for displaying game results or logs.

Para ingresar los clientes o gestionarlos se le hace click en el menú Clientes y se selecciona gestión de clientes:



The screenshot shows the 'Gestion Clientes' window of the 'CASINO EL MILLON' application. It includes a search bar and input fields for client information:

- Buscar:** Search bar with a 'Buscar' button.
- Nombres:** Input field.
- Apellidos:** Input field.
- Cedula:** Input field.
- Cuenta:** Input field with a value of '\$ 1000'.

Below the input fields are four buttons: 'Crear' (with a plus icon), 'Actualizar' (with a refresh icon), 'Cancelar' (with a red X icon), and 'Eliminar' (with a trash icon).

At the bottom, there is a table with the following data:

Id	Cedula	Nombre	Apellido	Cuenta
1	0000000000	Banco	Banco	50000.0

Below the table are two buttons: 'Actualizar Lista' and 'Atras'.


En este caso se ingresa los clientes:



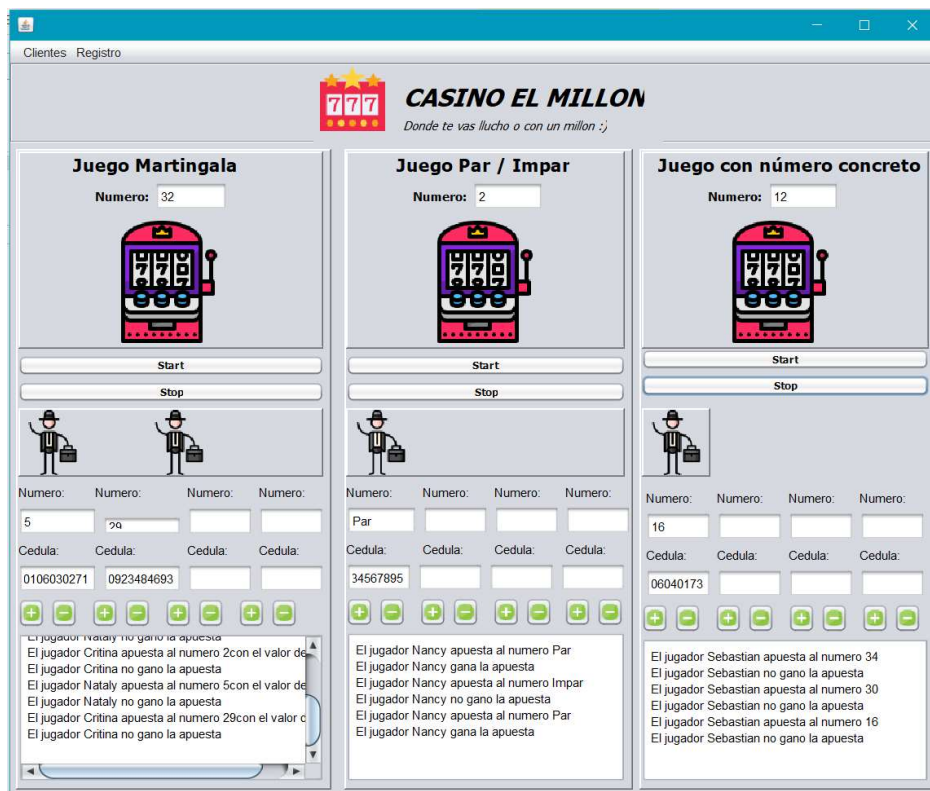
Id	Cedula	Nombre	Apellido	Cuenta
1	0000000000	Banco	Banco	50000.0
2	0106040173	Sebastian	Uyaguani	1000.0
3	0106030271	Nataly	Carmona	1000.0
4	0923484693	Cristina	Martinez	1000.0
5	1234567895	Nancy	Cordova	1000.0

Para ingresar a un juego se ingresa los clientes mediante la cedula:



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

Para los juegos el número de límite inferior es de una persona y el número mayor es de 4 personas además para comenzar a jugar se aplasta en el start y comienza los hilos.



CASINO EL MILLON
Donde te vas lluchu o con un millon :)

Juego Martingala
Numero: 32

Juego Par / Impar
Numero: 2

Juego con número concreto
Numero: 12

Start
Stop

Numero: Numero: Numero: Numero:
5 29

Cedula: Cedula: Cedula: Cedula:
0106030271 0923484693

El jugador Nancy apuesta al número 2 con el valor de 10
El jugador Nancy gana la apuesta
El jugador Nancy apuesta al número 4 con el valor de 20
El jugador Nancy gana la apuesta
El jugador Nancy apuesta al número 8 con el valor de 40
El jugador Nancy gana la apuesta
El jugador Nancy apuesta al número 16 con el valor de 80
El jugador Nancy gana la apuesta
El jugador Nancy apuesta al número 32 con el valor de 160
El jugador Nancy gana la apuesta

Numero: Numero: Numero: Numero:
Par

Cedula: Cedula: Cedula: Cedula:
34567895

El jugador Nancy apuesta al número Par
El jugador Nancy gana la apuesta
El jugador Nancy apuesta al número Impar
El jugador Nancy no gana la apuesta
El jugador Nancy apuesta al número Par
El jugador Nancy gana la apuesta

Numero: Numero: Numero: Numero:
16

Cedula: Cedula: Cedula: Cedula:
06040173

El jugador Sebastian apuesta al número 34
El jugador Sebastian no gana la apuesta
El jugador Sebastian apuesta al número 30
El jugador Sebastian no gana la apuesta
El jugador Sebastian apuesta al número 16
El jugador Sebastian no gana la apuesta


Para visualizar los registros se da click en el menú con el nombre de registro y se selecciona ver registros en el cual se puede visualizar el historial de transacciones:



CASINO EL MILLON
Donde te vas lluchu o con un millon :)

Id	Numero Apostado	Valor	Tipo	Modo
4028	22	10	Retirar	Apuesta Concreto
4029	23	10	Retirar	Apuesta Concreto
4030	12	10	Retirar	Apuesta Concreto
4031	9	10	Retirar	Apuesta Concreto
4032	35	10	Retirar	Apuesta Concreto
4033	21	10	Retirar	Apuesta Concreto
4034	24	10	Retirar	Apuesta Concreto
4035	33	10	Retirar	Apuesta Concreto
4036	31	10	Retirar	Apuesta Concreto
4037	12	10	Retirar	Apuesta Concreto
4038	9	10	Retirar	Apuesta Concreto
4039	34	10	Retirar	Apuesta Concreto
4040	8	10	Retirar	Apuesta Concreto
4041	16	10	Retirar	Apuesta Concreto
4042	9	360	ingreso	Gano Concreto
4043	11	10	Retirar	Apuesta Concreto
4044	8	360	ingreso	Gano Concreto
4045	13	10	Retirar	Apuesta Concreto
4046	23	10	Retirar	Apuesta Concreto
4047	15	10	Retirar	Apuesta Concreto
4048	34	10	Retirar	Apuesta Concreto
4049	30	10	Retirar	Apuesta Concreto

Listar Atras

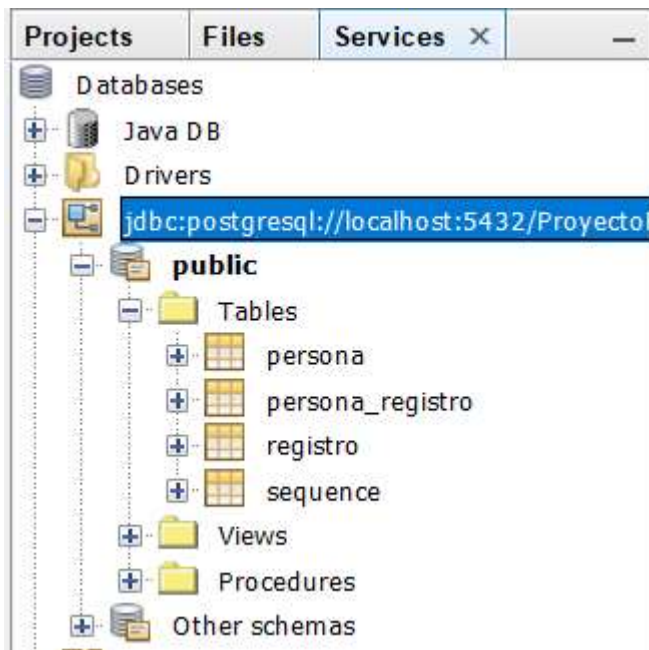
	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		


Por otra parte si se desea ver las transacciones de parte de cada cliente se selecciona ver registro de parte de el cliente:



Id	Numreo Apostado	Valor	Tipo	Modo
17181	14	10	Retirar	Apuesta Concreto
17182	22	10	Retirar	Apuesta Concreto
17183	7	10	Retirar	Apuesta Concreto
17184	9	10	Retirar	Apuesta Concreto
17185	15	10	Retirar	Apuesta Concreto
17189	10	10	Retirar	Apuesta Concreto
17187	28	360	ingreso	Gano Concreto
17188	19	10	Retirar	Apuesta Concreto
17186	6	10	Retirar	Apuesta Concreto
18657	31	20	ingreso	Gano Par Impar
18662	10	10	Retirar	Apuesta Par Impar
18658	10	10	Retirar	Apuesta Par Impar
18663	32	20	ingreso	Gano Par Impar
18659	10	10	Retirar	Apuesta Par Impar
18664	10	10	Retirar	Apuesta Par Impar
18660	9	20	ingreso	Gano Par Impar
18661	10	10	Retirar	Apuesta Par Impar
18665	32	20	ingreso	Gano Par Impar
19334	33	40	Retirar	Apuesta Martingala
19337	34	20	Retirar	Apuesta Martingala
19335	6	10	Retirar	Apuesta Martingala
19336	21	80	Retirar	Apuesta Martingala

Para comprobar que se guardó en la base de datos se dirige a la conexión de la base de datos:



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

Ahí le ponemos mostrar los datos de la tabla persona

SELECT * FROM "public".pe... X					
Max. rows: 100 Fetched Rows: 5					
#	persona_id	apellido	cedula	cuenta	nombre
1		5 Cordova	1234567895		910 Nancy
2		1 Banco	0000000000		50,970 Banco
3		2 Uyaguari	0106040173		1.380 Sebastian
4		3 Carmona	0106030271		370 Nataly
5		4 Martinez	0923484693		370 Critina

Y también los datos de la tabla registro


SELECT * FROM "public".re... X						
Max. rows: 100 Fetched Rows: 100						
#	id	modalidad	numero	tipo	valor	fk_persona
1		1 Apuesta		0 Ingreso	10	1
2		2 Apuesta		0 Ingreso	10	1
3		3 Apuesta		0 Ingreso	10	1
4		4 Apuesta Martingala		2 Retirar	10	3
5		5 Apuesta		0 Ingreso	10	1
6		6 Apuesta		0 Ingreso	10	1
7		7 Apuesta		0 Ingreso	10	1
8		8 Apuesta Martingala		10 Retirar	10	4
9		9 Apuesta Par Impar		10 Retirar	10	5
10		10 Apuesta		0 Ingreso	10	1
11		11 Apuesta		0 Ingreso	10	1
12		12 Apuesta		0 Ingreso	10	1
13		13 Perdida		0 Retirada	20	1
14		14 Gano Par Impar		16 ingreso	20	5
15		15 Apuesta Par Impar		10 Retirar	10	5
16		16 Apuesta		0 Ingreso	10	1
17		17 Apuesta		0 Ingreso	10	1
18		18 Perdida		0 Retirada	20	1
19		19 Apuesta		0 Ingreso	10	1
20		20 Apuesta		0 Ingreso	10	1
21		21 Apuesta Concreto		34 Retirar	10	2
22		22 Perdida		0 Retirada	20	1
23		23 Apuesta		0 Ingreso	10	1

RESULTADO(S) OBTENIDO(S):

- Un programa funcional con ingreso de datos y con la facilidad de agregar datos a la Base creado con JPA e hilos.

CONCLUSIONES:

- Con esta práctica sirve como introducción para la programación en base de datos la cual nos servirá para que en el futuro cercano pongamos en práctica los conocimientos adquiridos y reforzar los conocimientos acerca de hilos.

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

RECOMENDACIONES:

- Aplicar los conocimientos adquiridos con mayor frecuencia.

Nombre de estudiante: Sebastián Roberto Uyaguari Ramón

Firma de estudiante:

