

	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: PROGRAMACION APLICADA	
NRO. PRÁCTICA:	3	TÍTULO PRÁCTICA: PRUEBA DE BASE DE DATOS	
OBJETIVO ALCANZADO: <ul style="list-style-type: none"> Consolidar los conocimientos adquiridos en clase sobre JPA. 			
ACTIVIDADES DESARROLLADAS			
1. Problema: <p>Realizar un sistema implementando todos los conceptos vistos en clases para gestionar la hipoteca de las casas con las siguientes características:</p> <ul style="list-style-type: none"> Las personas compran casas y se convierten en propietarios. Para pagarlas es habitual que el propietario formalice un préstamo hipotecario con una entidad bancaria. El banco toma la casa en forma de aval en caso de impago de las mensualidades. En el caso de que el capital fiado supera el valor de tasación de la casa y el sueldo del propietario no es suficiente, el banco suele exigir la presencia de un avalista (garante). Para formalizar la hipoteca se necesitan los datos personales del propietario, además de su cédula, dirección de la casa, su dirección, nombres, apellidos y fecha de nacimiento y del garante de ser necesario. El capital de la hipoteca se ajusta teniendo en cuenta el valor de tasación de la casa y los datos de dirección. Toda hipoteca se formaliza detallando el capital, el interés (8,99 - 16,99%) y la duración (fecha de inicio y fecha de fin). A partir de estos datos se calcula el importe de cada mensualidad para el total del tiempo que pide el préstamo. No es necesario guardar los datos del banco, pero si un sistema de autenticación. Generar los datos con el sistema de amortización alemán [1]. 			
2. Uso de JPA:			

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

Persistence Units Add

PruebaJPAPU Remove

General:

Persistence Unit Name:

Persistence Library:

JDBC Connection:

☐ Use Java Transaction APIs

Table Generation Strategy: ☒ Create ☐ Drop and Create ☐ None

Validation Strategy: ☒ Auto ☐ Callback ☐ None

Shared Cache Mode: ☐ All ☐ None ☐ Enable Selective ☐ Disable Selective ☒ Unspecified

☐ Include All Entity Classes in "PruebaJPA" Module

Include Entity Classes:

- ec.edu.ups.modelo.Persona
- ec.edu.ups.modelo.Usuario
- ec.edu.ups.modelo.Propietario
- ec.edu.ups.modelo.Casa
- ec.edu.ups.modelo.Hipoteca

Add Class... Remove

Properties:

Controladores:

Abstract Controlador:

```
package ec.edu.ups.controlador;

import ec.edu.ups.utils.JPAUtils;
import java.lang.reflect.ParameterizedType;
import java.lang.reflect.Type;
import java.util.ArrayList;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.persistence.EntityManager;

/**
 *
 * @author user
 */
public abstract class AbstractControlador<E> {

    private List<E> lista;
    private Class<E> clase;
    private EntityManager em;

    /**
     *
     */
    public AbstractControlador() {

        lista= new ArrayList<>();
        Type t= getClass().getGenericSuperclass();
        ParameterizedType pt = (ParameterizedType) t;
        this.clase = (Class) pt.getActualTypeArguments()[0];
    }
}
```

```
this.em=JPAUtils.getEntityManager();
}

public AbstractControlador(EntityManager em) {
    lista= new ArrayList<>();
    Type t= getClass().getGenericSuperclass();
    ParameterizedType pt = (ParameterizedType) t;
    this.clase = (Class) pt.getActualTypeArguments()[0];
    this.em=em;
}


public boolean crear (E objeto){
    try {
        if(this.validar(objeto)){
            em.getTransaction().begin();
            em.persist(objeto);
            em.getTransaction().commit();
            lista.add(objeto);
            return true;
        } catch (Exception ex) {
            Logger.getLogger(AbstractControlador.class.getName()).log(Level.SEVERE,
null, ex);
        }

        return false;
    }

    public boolean eliminar (E objeto){
        em.getTransaction().begin();
        em.remove(em.merge(objeto));
        em.getTransaction().commit();
        lista.remove(objeto);
        return true;
    }

    public boolean actualizar (E objeto){
        try {
            if(this.validar(objeto)){
                em.getTransaction().begin();
                objeto=em.merge(objeto);
                em.getTransaction().commit();
                this.lista=buscarTodo();
                return true;
            }
        } catch (Exception ex) {
            Logger.getLogger(AbstractControlador.class.getName()).log(Level.SEVERE,
null, ex);
        }

        return false;
    }
}
```

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

```

public E buscar (Object id){
    return (E) em.find(clase, id);
}

public List<E> buscarTodo (){
    return em.createQuery("Select t from"+ clase.getSimpleName()+
"t").getResultList();
}

public abstract boolean validar(E objeto) throws Exception;

public List<E> getLista() {
    return lista;
}

public void setLista(List<E> lista) {
    this.lista = lista;
}

public Class<E> getClass() {
    return clase;
}

public void setClass(Class<E> clase) {
    this.clase = clase;
}

public EntityManager getEm() {
    return em;
}

public void setEm(EntityManager em) {
    this.em = em;
}
}

```

Controlador Casa:

```

package ec.edu.ups.controlador;

import ec.edu.ups.modelo.Casa;

/**
 *
 * @author user
 */
public class ControladorCasa extends AbstractControlador<Casa>{

    @Override
    public boolean validar(Casa objeto) throws Exception {
        return false;
    }
}

```

}

}

Controlador Hipoteca:

```
package ec.edu.ups.controlador;
```

```
import ec.edu.ups.modelo.Hipoteca;
```

```
/**
```

```
*
```

```
* @author user
```

```
*/
```

```
public class ControladorHipoteca extends AbstractControlador<Hipoteca>{
```

```
    @Override
```

```
    public boolean validar(Hipoteca objeto) throws Exception {  
        return true;
```

```
    }
```

}

Controlador Persona:

```
package ec.edu.ups.controlador;
```

```
import ec.edu.ups.Excepciones.ExcepcionCedula;
```

```
import ec.edu.ups.modelo.Persona;
```

```
/**
```

```
*
```

```
* @author user
```

```
*/
```

```
public class ControladorPersona extends AbstractControlador<Persona> {
```

```
    @Override
```

```
    public boolean validar(Persona objeto) throws ExcepcionCedula {
```

```
        int suma = 0;
```

```
        String x = objeto.getCedula();
```

```
        if (x.length() == 9) {
```

```
            return false;
```

```
        } else {
```

```
            int a[] = new int[x.length() / 2];
```

```
            int b[] = new int[(x.length() / 2)];
```

```
            int c = 0;
```

```
            int d = 1;
```

```
            for (int i = 0; i < x.length() / 2; i++) {
```

```
                a[i] = Integer.parseInt(String.valueOf(x.charAt(c)));
```

```
                c = c + 2;
```

```
                if (i < (x.length() / 2) - 1) {
```

```
                    b[i] = Integer.parseInt(String.valueOf(x.charAt(d)));
```

```
                    d = d + 2;
```

```
                }
```

```
            }
```

```
        }
```

```

        for (int i = 0; i < a.length; i++) {
            a[i] = a[i] * 2;
            if (a[i] > 9) {
                a[i] = a[i] - 9;
            }
            suma = suma + a[i] + b[i];
        }
        int aux = suma / 10;
        int dec = (aux + 1) * 10;
        if ((dec - suma) == Integer.parseInt(String.valueOf(x.charAt(x.length() - 1)))) {
            return true;
        } else if (suma % 10 == 0 && x.charAt(x.length() - 1) == '0') {
            return true;
        } else {
            throw new ExcepcionCedula();
        }
    }

}

}

Controlador Usuario:

package ec.edu.ups.controlador;

import ec.edu.ups.modelo.Usuario;

/**
 *
 * @author user
 */
public class ControladorUsuario extends AbstractControlador<Usuario> {

    private Usuario usuario;

    @Override
    public boolean validar(Usuario objeto) {

        return true;
    }

    public boolean iniciarSesion(String correo, String pass) {

```

```
for (Usuario usu : super.getList()) {
    Usuario u = (Usuario) usu;
    if (u.getUsername().equals(correo) && u.getPassword().equals(pass)) {
        this.usuario = u;
        return true;
    }
}
return false;

}

public Usuario getUsuario() {
    return usuario;
}

}
```

Modelo:

Casa:

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

import java.io.Serializable;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.OneToOne;


/**
 *
 * @author user
 */
@Entity
public class Casa implements Serializable {

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

    @Column (name="direccion")
    private String direccion;

    @OneToOne(mappedBy = "casa")
    private Propietario propietario;

    public Long getId() {
        return id;
    }
}
```

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

```

}

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

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

public Propietario getPropietario() {
    return propietario;
}

public void setPropietario(Propietario propietario) {
    this.propietario = propietario;
}

public double getValor() {
    return valor;
}

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

public String getDireccion() {
    return direccion;
}

public void setDireccion(String direccion) {
    this.direccion = direccion;
}

@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 Casa)) {
        return false;
    }
    Casa other = (Casa) 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 "ec.edu.ups.modelo.Casa[ id=" + id + " ]";  
    }  
}
```

Hipoteca:

```
package ec.edu.ups.modelo;  
  
import java.io.Serializable;  
import java.util.Date;  
import javax.persistence.Column;  
import javax.persistence.Entity;  
import javax.persistence.GeneratedValue;  
import javax.persistence.GenerationType;  
import javax.persistence.Id;  
import javax.persistence.JoinColumn;  
import javax.persistence.ManyToOne;  
import javax.persistence.Temporal;  
import javax.persistence.TemporalType;  
  
/**  
 *  
 * @author user  
 */  
@Entity  
public class Hipoteca implements Serializable {  
  
    private static final long serialVersionUID = 1L;  
    @Id  
    @GeneratedValue(strategy = GenerationType.AUTO)  
    private Long id;  
    @Column (name="capital")  
    private double capital;  
    @Column (name="interes")  
    private double interes;  
  
    @Column(name = "fechaInicio")  
    @Temporal(TemporalType.DATE)  
    private Date fechaInicio;  
  
    @Column(name = "fechaFin")  
    @Temporal(TemporalType.DATE)  
    private Date fechaFin;  
  
    @ManyToOne  
    @JoinColumn(name="fk_propietario")  
    private Propietario propietario;  
    public Long getId() {  
        return id;  
    }  
  
    public void setId(Long id) {  
        this.id = id;  
    }  
}
```

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

```

public double getCapital() {
    return capital;
}

public void setCapital(double capital) {
    this.capital = capital;
}

public double getInteres() {
    return interes;
}

public void setInteres(double interes) {
    this.interes = interes;
}

public Date getFechaInicio() {
    return fechaInicio;
}

public void setFechaInicio(Date fechaInicio) {
    this.fechaInicio = fechaInicio;
}

public Date getFechaFin() {
    return fechaFin;
}

public void setFechaFin(Date fechaFin) {
    this.fechaFin = fechaFin;
}


public Propietario getPropietario() {
    return propietario;
}

public void setPropietario(Propietario propietario) {
    this.propietario = propietario;
}

@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 Hipoteca)) {
        return false;
    }

```

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

```

    }
    Hipoteca other = (Hipoteca) object;
    return !((this.id == null && other.id != null) || (this.id != null &&
!this.id.equals(other.id)));
}

@Override
public String toString() {
    return "ec.edu.ups.modelo.hipoteca[ id=" + id + " ]";
}

}
Persona:

package ec.edu.ups.modelo;

import java.io.Serializable;
import java.util.Date;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;


/**
 *
 * @author user
 */
@Entity
public class Persona implements Serializable {

    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;
    @Column(name = "cedula")
    private String cedula;
    @Column(name = "nombre")
    private String nombre;
    @Column(name = "apellido")
    private String apellido;
    @Column(name = "fechaNacimiento")
    @Temporal(TemporalType.DATE)
    private Date fechaNacimiento;
    @Column(name = "direccion")
    private String direccion;

    public Long getId() {
        return id;
    }

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

```

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

```

}

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;
}

public Date getFechaNacimiento() {
    return fechaNacimiento;
}

public void setFechaNacimiento(Date fechaNacimiento) {
    this.fechaNacimiento = fechaNacimiento;
}

public String getDireccion() {
    return direccion;
}

public void setDireccion(String direccion) {
    this.direccion = direccion;
}

@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() {
        StringBuilder sb = new StringBuilder();
        sb.append("Persona{id=").append(id);
        sb.append(", cedula=").append(cedula);
        sb.append(", nombre=").append(nombre);
        sb.append(", apellido=").append(apellido);
        sb.append(", fechaNacimiento=").append(fechaNacimiento);
        sb.append(", direccion=").append(direccion);
        sb.append('}');
        return sb.toString();
    }
}
```

Propietario:

```
package ec.edu.upse.modelo;

import java.io.Serializable;
import java.util.Set;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.OneToMany;
import javax.persistence.OneToOne;

/**
 *
 * @author user
 */
@Entity
public class Propietario extends Persona implements Serializable {

    @Column(name="direccionCasa")
    @OneToOne
    private Casa casa;

    @Column(name="garante")
    private String garante;

    @OneToMany(mappedBy="propietario",targetEntity=Hipoteca.class)
    private Set<Hipoteca> hipotecas;

    public Casa getCasa() {
```

```
        return casa;
    }

    public void setCasa(Casa casa) {
        this.casa = casa;
    }

    public Set<Hipoteca> getHipotecas() {
        return hipotecas;
    }

    public void setHipotecas(Set<Hipoteca> hipotecas) {
        this.hipotecas = hipotecas;
    }

    public String getGarante() {
        return garante;
    }

    public void setGarante(String garante) {
        this.garante = garante;
    }

    @Override
    public String toString() {
        StringBuilder sb = new StringBuilder();
        sb.append("Propietario{casa=").append(casa);
        sb.append(", garante=").append(garante);
        sb.append('}');
        return sb.toString();
    }
}
```


Usuario:

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

import java.io.Serializable;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.OneToOne;

/**
 *
 * @author user
 */
@Entity
public class Usuario implements Serializable {

    private static final long serialVersionUID = 1L;
```

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

```

@Id
@GeneratedValue(strategy = GenerationType.AUTO)
private Long id;
@Column(name = "username")
private String username;
@Column(name = "password")
private String password;
@OneToOne
@JoinColumn (name="idPersona")
private Persona persona;
public Long getId() {
    return id;
}

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

public static long getSerialVersionUID() {
    return serialVersionUID;
}

public String getUsername() {
    return username;
}

public String getPassword() {
    return password;
}

public Persona getPersona() {
    return 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 Usuario)) {
        return false;
    }
    Usuario other = (Usuario) object;
    if ((this.id == null && other.id != null) || (this.id != null &&
!this.id.equals(other.id))) {
        return false;
    }
    return true;
}

```

```
}  
  
@Override  
public String toString() {  
    StringBuilder sb = new StringBuilder();  
    sb.append("Usuario{id=").append(id);  
    sb.append(", username=").append(username);  
    sb.append(", password=").append(password);  
    sb.append(", persona=").append(persona);  
    sb.append('}');  
    return sb.toString();  
}
```

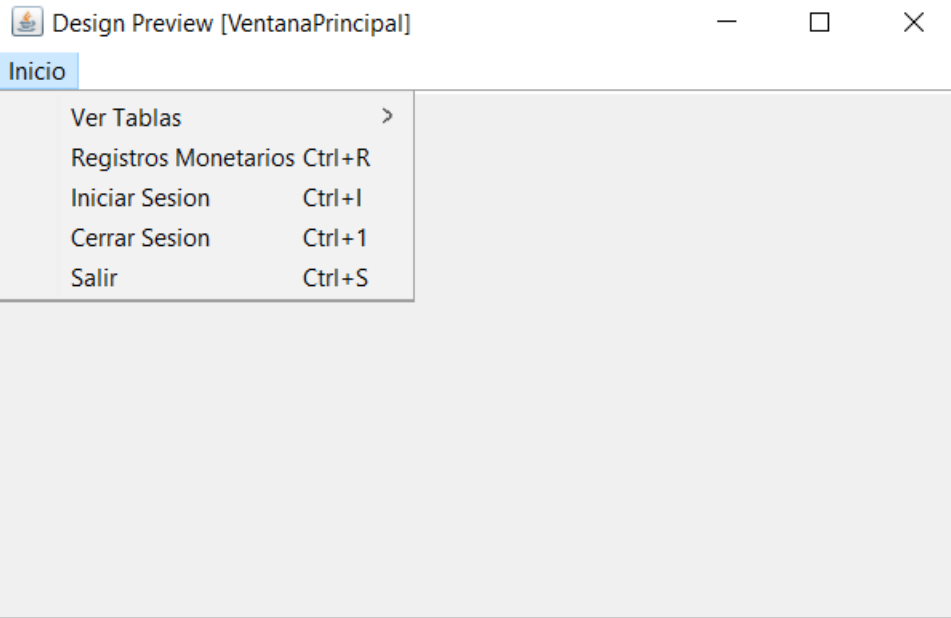
}
Utils:

UTILS:

```
package ec.edu.ups.utils;  
  
import javax.persistence.EntityManager;  
import javax.persistence.EntityManagerFactory;  
import javax.persistence.Persistence;  
  
/**  
 *  
 * @author user  
 */  
public class JPAUtils {  
  
    private static final EntityManagerFactory emf =  
Persistence.createEntityManagerFactory("PruebaJPAPU");  
  
    public static EntityManager getEntityManager() {  
        return emf.createEntityManager();  
    }  
}
```

Vista:

Ventana Principal:



```
package ec.edu.ups.vista;
```


```
import ec.edu.ups.controlador.ControladorAutoridadCivil;
import ec.edu.ups.controlador.ControladorMatrimonio;
import ec.edu.ups.controlador.ControladorPersona;
import javax.swing.JDesktopPane;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
```

```
/**
 *
 * @author paul_
 */
```

```
public class VentanaPrincipal extends javax.swing.JFrame {
```

```
    private VentanaIniciarSesion ventanaIniciarSesion;
    private VentanaGestionPersona registrarPersona;
    private VentanaRegistroCasa registrarCasa;
    private VentanaRegistroUsuario registrarUsuario;
    private ControladorPersona controladorPersona;
    private ControladorHipoteca controladorHipoteca;
```

```
    public VentanaPrincipal() {
        initComponents();
        GestionMenu.setVisible(false);
        btnCerrarSesionMenu.setVisible(false);
        controladorPersona = new ControladorPersona("datos/Persona.obj");
        controladorUsuario = new ControladorUsuario("datos/Usuario.obj");
        controladorCasa = new ControladorCasa("datos/Casa.obj");
```

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

```

        ventanaIniciarSesion = new VentanaIniciarSesion(this, controladorUsuario);
        registrarPersona = new VentanaGestionPersona(controladorPersona);
        registrarCasa = new
VentanaRegistroCasa(controladorPersona, controladorUsuario, controladorCasa);
        registrarUsuario = new VentanaRegistroUsuario(controladorUsuario);

    }
    private void btnSalirMenuActionPerformed(java.awt.event.ActionEvent evt) {
        System.exit(0);
    }

    private void btnRegistrarUsuarioMenuActionPerformed(java.awt.event.ActionEvent
evt) {
        desktopPane.add(registrarUsuario);
        registrarUsuario.setVisible(true);
    }

    private void btnMenuIniciarSesionActionPerformed(java.awt.event.ActionEvent evt)
{
        desktopPane.add(ventanaIniciarSesion);
        ventanaIniciarSesion.setVisible(true);
    }

    private void btnCerrarSesionMenuActionPerformed(java.awt.event.ActionEvent evt)
{
        btnMenuIniciarSesion.setVisible(true);
        btnRegistrarUsuarioMenu.setVisible(true);
        GestionMenu.setVisible(false);
        btnSalirMenu.setVisible(true);
    }

    private void btnRegistroCasaActionPerformed(java.awt.event.ActionEvent evt) {
        desktopPane.add(registrarCasa);
        registrarCasa.setVisible(true);
    }


    private void btnGestionPersonaActionPerformed(java.awt.event.ActionEvent evt) {
        desktopPane.add(registrarPersona);
        registrarPersona.setVisible(true);
    }
    public JMenu getGestionMenu() {
        return GestionMenu;
    }

    public void setGestionMenu(JMenu GestionMenu) {
        this.GestionMenu = GestionMenu;
    }

    public JMenu getMenuInicio() {
        return MenuInicio;
    }

    public void setMenuInicio(JMenu MenuInicio) {
        this.MenuInicio = MenuInicio;
    }

```

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

```

}

public JMenuItem getBtnCerrarSesionMenu() {
    return btnCerrarSesionMenu;
}

public void setBtnCerrarSesionMenu(JMenuItem btnCerrarSesionMenu) {
    this.btnCerrarSesionMenu = btnCerrarSesionMenu;
}

public JMenuItem getBtnMenuIniciarSesion() {
    return btnMenuIniciarSesion;
}

public void setBtnMenuIniciarSesion(JMenuItem btnMenuIniciarSesion) {
    this.btnMenuIniciarSesion = btnMenuIniciarSesion;
}

public JMenuItem getBtnRegistrarPersonaMenu() {
    return btnRegistrarUsuarioMenu;
}

public void setBtnRegistrarPersonaMenu(JMenuItem btnRegistrarPersonaMenu) {
    this.btnRegistrarUsuarioMenu = btnRegistrarPersonaMenu;
}

public JMenuItem getBtnSalirMenu() {
    return btnSalirMenu;
}

public void setBtnSalirMenu(JMenuItem btnSalirMenu) {
    this.btnSalirMenu = btnSalirMenu;
}

public JDesktopPane getDesktopPane() {
    return desktopPane;
}


public void setDesktopPane(JDesktopPane desktopPane) {
    this.desktopPane = desktopPane;
}

public JMenuItem getjMenuItem1() {
    return btnRegistroMatrimonio;
}

public void setjMenuItem1(JMenuItem jMenuItem1) {
    this.btnRegistroMatrimonio = jMenuItem1;
}

public void setMenuBar(JMenuBar menuBar) {
    this.menuBar = menuBar;
}

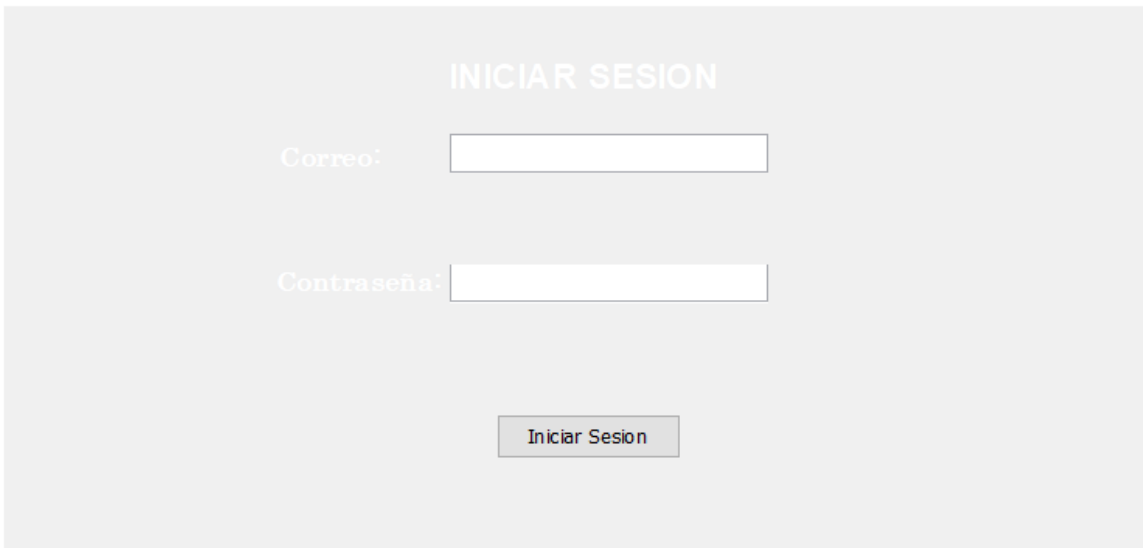
```

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

```
// Variables declaration - do not modify
private javax.swing.JMenu GestionMenu;
private javax.swing.JMenu MenuInicio;
private javax.swing.JMenuItem btnCerrarSesionMenu;
private javax.swing.JMenuItem btnGestionPersona;
private javax.swing.JMenuItem btnMenuIniciarSesion;
private javax.swing.JMenuItem btnRegistrarUsuarioMenu;
private javax.swing.JMenuItem btnRegistroMatrimonio;
private javax.swing.JMenuItem btnSalirMenu;
private javax.swing.JDesktopPane desktopPane;
private javax.swing.JMenuBar menuBar;
// End of variables declaration
```

```
}
```

Ventana Iniciar Sesión:




```
package ec.edu.ups.vista;

import ec.edu.ups.controlador.ControladorAutoridadCivil;
import javax.swing.JOptionPane;

/**
 *
 * @author paul_
 */
public class VentanaIniciarSesion extends javax.swing.JInternalFrame {

    private VentanaPrincipal ventanaPrincipal;
    private ControladorAutoridadCivil controladorUsuario;

    /**
     * Creates new form VentanaIniciarSesion
     */
}
```

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

```

* @param ventanaPrincipal
* @param controladorUsuario
*/
public VentanaIniciarSesion(VentanaPrincipal ventanaPrincipal,
ControladorAutoridadCivil controladorUsuario) {
    initComponents();
    this.ventanaPrincipal = ventanaPrincipal;
    this.controladorUsuario = controladorUsuario;
}
private void btnIniciarSesionActionPerformed(java.awt.event.ActionEvent evt) {
    controladorUsuario.cargarDatos();
    String usuario = txtCorreo.getText();
    String pass = "";
    char[] pass1 = txtPass.getPassword();
    for (int i = 0; i < pass1.length; i++) {
        pass = pass + pass1[i];
    }

    if (controladorUsuario.iniciarSesion(usuario, pass)) {
        ventanaPrincipal.getBtnMenuIniciarSesion().setVisible(false);
        ventanaPrincipal.getBtnCerrarSesionMenu().setVisible(true);
        ventanaPrincipal.getGestionMenu().setVisible(true);


        ventanaPrincipal.getBtnSalirMenu().setVisible(false);
        ventanaPrincipal.getBtnRegistrarPersonaMenu().setVisible(false);
        ventanaPrincipal.getGestionMenu().setVisible(true);

        this.dispose();
        JOptionPane.showMessageDialog(this, "Inicio de sesion exitoso");
    } else {

        JOptionPane.showMessageDialog(this, "Usuario o contrasena incorrecta ");
        Limpiar();
    }

}
public void Limpiar() {
    txtCorreo.setText("");
    txtPass.setText("");
}
// Variables declaration - do not modify
private javax.swing.JButton btnIniciarSesion;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel lblContra;
private javax.swing.JLabel lblCorreo;
private javax.swing.JTextField txtCorreo;
private javax.swing.JPasswordField txtPass;
// End of variables declaration
}

```

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

Registro Propietario:

Propietario

Nombre

Cedula

Fecha Nacimiento

febrero

2021

lun.	mar.	mié.	jue.	vie.	sáb.	dom.
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28

Apellido

Direccion

Direccion Casa

Nombre Garante

GUARDAR

ELIMINAR

ACTUALIZAR

```

package ec.edu.ups.vista;

import ec.edu.ups.controlador.ControladorAutoridadCivil;
import ec.edu.ups.modelo.AutoridadCivil;
import java.io.IOException;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;


/**
 *
 * @author paul_
 */
public class VentanaRegistroAutoridad extends javax.swing.JInternalFrame {

    private ControladorAutoridadCivil controlador;

    public VentanaRegistroAutoridad(ControladorAutoridadCivil controlador) {
        initComponents();
        this.controlador = controlador;
        controlador.cargarDatos();
    }

    public void Limpiar() {
        txtGcedula.setText("");
        txtGnombre.setText("");
        txtGapellido.setText("");
        txtCargo.setText("");
    }

```

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

```

txtEstadoCivil.setText("");
txtDireccion.setText("");
txtCorreo.setText("");
jGpr.setText("");

}
private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
String pass = "";
char[] pass1 = jGpr.getPassword();
for (int i = 0; i < pass1.length; i++) {
    pass = pass + pass1[i];
}
SimpleDateFormat formato = new SimpleDateFormat("dd/MM/yyyy");
Date fechaN = new Date();
try {
    fechaN = formato.parse(txtFechaN.getText().trim());
} catch (ParseException ex) {
    System.out.println(ex);
}
AutoridadCivil usuario = new AutoridadCivil(txtGcedula.getText().trim(),
txtGnombre.getText().trim(), txtGapellido.getText().trim(),
txtDireccion.getText().trim(),
cbxGenero.getSelectedItem().toString().trim(),
fechaN, txtEstadoCivil.getText().trim(), "Autoridad",
txtCargo.getText().trim(),
txtCorreo.getText().trim(), pass.trim());


System.out.println(usuario);
if (controlador.validar(usuario)) {
    controlador.crear(usuario);
    JOptionPane.showMessageDialog(this, "Usuario creado correctamente");
    Limpiar();
    this.dispose();
} else {
    JOptionPane.showMessageDialog(this, "La cedula ingresada no es valida");
    Limpiar();
}

try {
    controlador.guardarDatos("datos/Autoridad.obj");
} catch (IOException ex) {

}

Logger.getLogger(VentanaRegistroAutoridad.class.getName()).log(Level.SEVERE, null,
ex);
}

```

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

Crédito Hipotecario:

Formalizar Hipoteca

Capital

Interés

Mensualidad

Fecha Inicio

febrero 2021

lun.	mar.	mié.	jue.	vie.	sáb.	dom.
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28

Fecha Fin

febrero 2021

lun.	mar.	mié.	jue.	vie.	sáb.	dom.
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28

GUARDAR

ELIMINAR

ACTUALIZAR

```


package ec.edu.ups.vista;

import ec.edu.ups.controlador.ControladorMatrimonio;
import ec.edu.ups.modelo.Matrimonio;
import ec.edu.ups.modelo.Persona;
import java.util.Iterator;
import javax.swing.table.DefaultTableModel;

/**
 *
 * @author paul_
 */
public class VentanaListaRegistrosDeMatrimonio extends javax.swing.JFrame {

    private ControladorMatrimonio controladorMatrimonio;
    public VentanaListaRegistrosDeMatrimonio(ControladorMatrimonio controladorMatrimonio) {
        initComponents();
        this.controladorMatrimonio= controladorMatrimonio;
    }

```


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

```
private void formInternalFrameActivated(javax.swing.event.InternalFrameEvent evt) {
    controladorMatrimonio.cargarDatos();
    DefaultTableModel modelo = (DefaultTableModel) tblRegistros.getModel();
    modelo.setRowCount(0);
}
```

3. Excepciones:

Excepción de la cedula:

```
public boolean validar(Persona objeto) throws ExcepcionCedula {
    int suma = 0;
    String x = objeto.getCedula();
    if (x.length() == 9) {
        return false;
    } else {
        int a[] = new int[x.length() / 2];
        int b[] = new int[(x.length() / 2)];
        int c = 0;
        int d = 1;
        for (int i = 0; i < x.length() / 2; i++) {
            a[i] = Integer.parseInt(String.valueOf(x.charAt(c)));
            c = c + 2;
            if (i < (x.length() / 2) - 1) {
                b[i] = Integer.parseInt(String.valueOf(x.charAt(d)));
                d = d + 2;
            }
        }


        for (int i = 0; i < a.length; i++) {
            a[i] = a[i] * 2;
            if (a[i] > 9) {
                a[i] = a[i] - 9;
            }
            suma = suma + a[i] + b[i];
        }
        int aux = suma / 10;
        int dec = (aux + 1) * 10;
        if ((dec - suma) ==
Integer.parseInt(String.valueOf(x.charAt(x.length() - 1)))) {
            return true;
        } else if (suma % 10 == 0 && x.charAt(x.length() - 1) == '0') {
            return true;
        } else {
            throw new ExcepcionCedula();
        }
    }
}

}
```

Excepción de iniciar sesión:

```
public boolean validar(Usuario objeto) {

    return true;
}
```

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

```

}

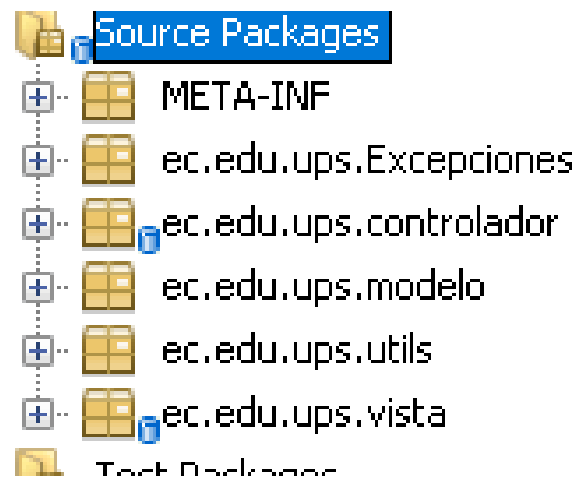
public boolean iniciarSesion(String correo, String pass) {

    for (Usuario usu : super.getLista()) {
        Usuario u = (Usuario) usu;
        if (u.getUsername().equals(correo) && u.getPassword().equals(pass)) {
            this.usuario = u;
            return true;
        }
    }
    return false;
}

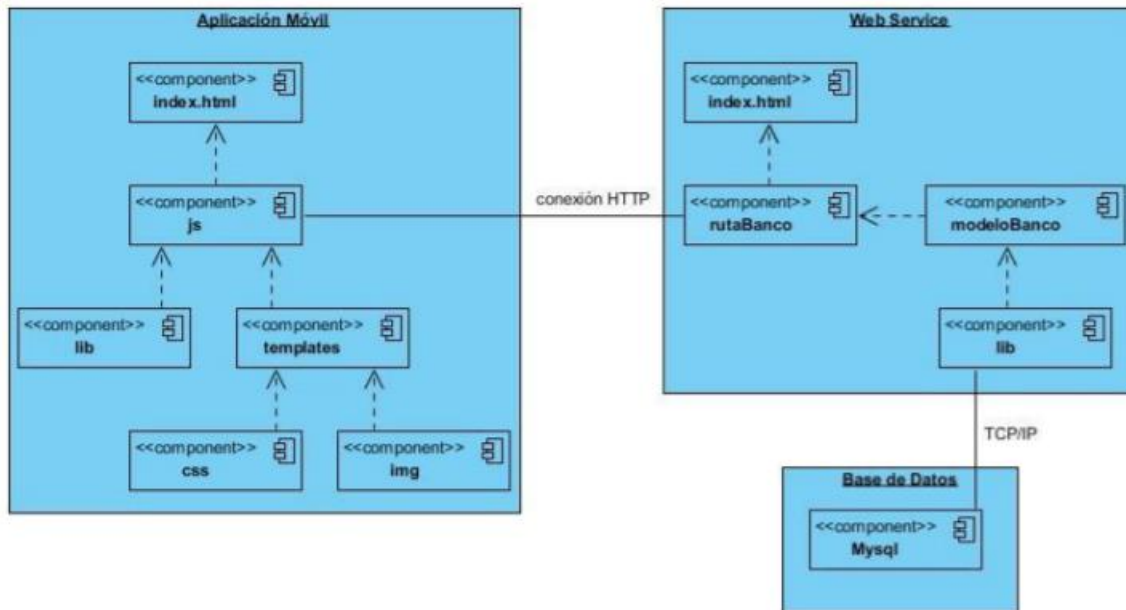
public Usuario getUsuario() {
    return usuario;
}

```

4. MVC:



5. Diagrama de Clases:



RESULTADO(S) OBTENIDO(S):

- Enlazar la Base de Datos (Postgress) con NetBeans y crear un proyecto.

CONCLUSIONES:

- Hacer este tipo de modelo es muy bueno a la hora de emplear una base de datos con un proyecto en el cual nos pida almacenar distintos parámetros.

RECOMENDACIONES:

- Usar la base de datos (Postgress) debido a que es muy fiable y fácil de manejar al momento de usar un almacenamiento de datos en el cual luego se pueda manejar las cosas de manera fácil.

Nombre de estudiante:

PAUL ALEXANDER GUAPUCAL CARDENAS

Firma de estudiante: