

FORMATO DE INFORME DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA ESTUDIANTES

CARRERA: COMPUTACIÓN/INGENIERÍA DE
SISTEMAS

ASIGNATURA: PROGRAMACIÓN APLICADA

NRO. PROYECTO: 1.1

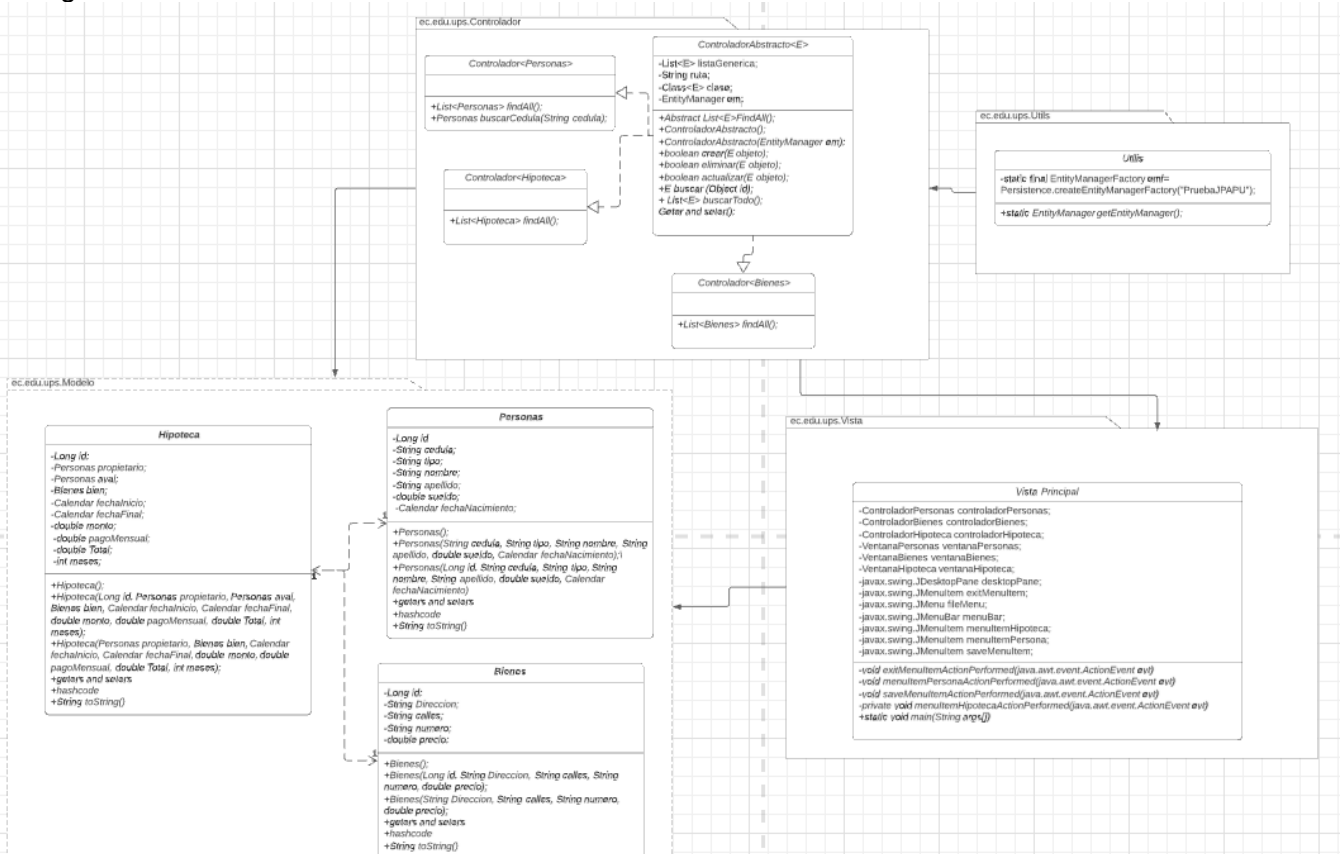
TÍTULO PROYECTO: Prueba Practica

OBJETIVO:

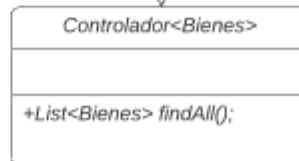
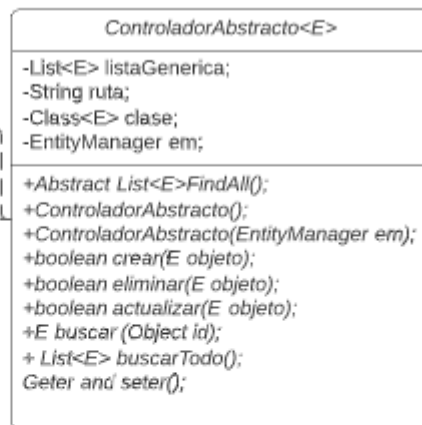
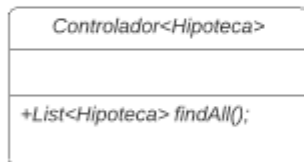
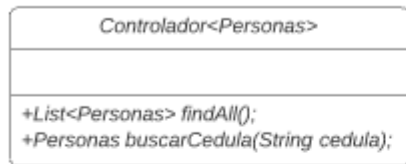
- Consolidar los conocimientos adquiridos en clase sobre JPA.

ACTIVIDADES DESARROLLADAS

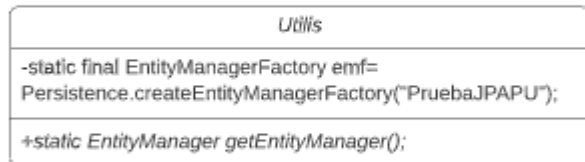
1. Diagrama de Clase



ec.edu.ups.Controlador



ec.edu.ups.Utils



ec.edu.ups.Vista

Vista Principal

```
-ControladorPersonas controladorPersonas;
-ControladorBienes controladorBienes;
-ControladorHipoteca controladorHipoteca;
-VentanaPersonas ventanaPersonas;
-VentanaBienes ventanaBienes;
-VentanaHipoteca ventanaHipoteca;
-javax.swing.JDesktopPane desktopPane;
-javax.swing.JMenuItem exitMenuItem;
-javax.swing.JMenu fileMenu;
-javax.swing.JMenuBar menuBar;
-javax.swing.JMenuItem menuItemHipoteca;
-javax.swing.JMenuItem menuItemPersona;
-javax.swing.JMenuItem menuItemSave;

-void exitMenuItemActionPerformed(java.awt.event.ActionEvent evt)
-void menuItemPersonaActionPerformed(java.awt.event.ActionEvent evt)
-void menuItemSaveActionPerformed(java.awt.event.ActionEvent evt)
-private void menuItemHipotecaActionPerformed(java.awt.event.ActionEvent evt)
+static void main(String args[])
```

ec.edu.ups.Modelo

Hipoteca

```
-Long id;
-Personas propietario;
-Personas aval;
-Bienes bien;
-Calendar fechaInicio;
-Calendar fechaFinal;
-double monto;
-double pagoMensual;
-double Total;
-int meses;

+Hipoteca();
+Hipoteca(Long id, Personas propietario, Personas aval,
Bienes bien, Calendar fechaInicio, Calendar fechaFinal,
double monto, double pagoMensual, double Total, int
meses);
+Hipoteca(Personas propietario, Bienes bien, Calendar
fechaInicio, Calendar fechaFinal, double monto, double
pagoMensual, double Total, int meses);
+getters and setters
+hashCode
+String toString()
```

Personas

```
-Long id;
-String cedula;
-String tipo;
-String nombre;
-String apellido;
-double sueldo;
-Calendar fechaNacimiento;

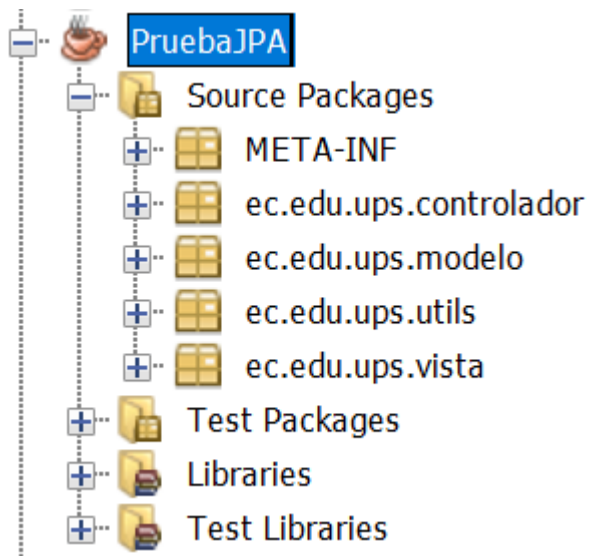
+Personas();
+Personas(String cedula, String tipo, String nombre, String
apellido, double sueldo, Calendar fechaNacimiento);
+Personas(Long id, String cedula, String tipo, String
nombre, String apellido, double sueldo, Calendar
fechaNacimiento)
+getters and setters
+hashCode
+String toString()
```

Bienes

```
-Long id;
-String Direccion;
-String calles;
-String numero;
-double precio;

+Bienes();
+Bienes(Long id, String Direccion, String calles, String
numero, double precio);
+Bienes(String Direccion, String calles, String numero,
double precio);
+getters and setters
+hashCode
+String toString()
```

2.Modelo Vista Controlador



Paquete Controlador

Controlador Abstracto

```
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 javax.persistence.EntityManager;

/**
 *
 * @author NANCY
 */
public abstract class ControladorAbstracto <E>{

    private List <E> listaGenerica;

    private Class<E> clase;

    private EntityManager em;

    public abstract List<E> findAll();

    public ControladorAbstracto() {
```

```

listaGenerica= new ArrayList();

//java.lang.reflect.Type t= getClass().getGenericSuperclass();

Type t = getClass().getGenericSuperclass();

ParameterizedType pt = (ParameterizedType) t;

clase= (Class) pt.getActualTypeArguments()[0];

em=JPAUtils.getEntityManager();

}

public ControladorAbstracto(EntityManager em) {

    listaGenerica= new ArrayList();

    //java.lang.reflect.Type t= getClass().getGenericSuperclass();

    Type t =getClass().getGenericSuperclass();

    ParameterizedType pt = (ParameterizedType) t;

    clase= (Class) pt.getActualTypeArguments()[0];

    this.em=em;

}

public boolean crear(E objeto){

    em.getTransaction().begin();

    em.persist(objeto);

    em.getTransaction().commit();

    listaGenerica.add(objeto);

    return true;

}

public boolean eliminar(E objeto){

    em.getTransaction().begin();

    em.remove(em.merge(objeto));

    em.getTransaction().commit();

    listaGenerica.remove(objeto);

    return true;

}

public boolean actualizar(E objeto){

```

```
    em.getTransaction().begin();

    em.merge(objeto);

    em.getTransaction().commit();

    return true;
}

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 List<E> getListaGenerica() {
    return listaGenerica;
}

public void setListaGenerica(List<E> listaGenerica) {
    this.listaGenerica = listaGenerica;
}

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

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

public EntityManager getEm() {
    return em;
}
```

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

Esta clase es abstracta por lo que se heredará a los demás controladores entonces es así como utilizamos programación genérica con la sentencia <E> para pasar el objeto requerido, a su vez esta clase también utiliza reflexión en la parte del constructor para poder obtener el tipo de clase que ingresa, en este controlador podemos ver los métodos del CRUD para los objetos que van a ser guardados en la base de datos es por eso que utilizamos el entity manager con el cual pasaremos objetos a la base de datos utilizando JPA.

Controlador Personas

```
package ec.edu.ups.controlador;  
  
import ec.edu.ups.modelo.Personas;  
  
import java.util.List;  
  
import javax.persistence.Query;  
  
/**  
 *  
 * @author NANCY  
 */  
  
public class ControladorPersonas extends ControladorAbstracto<Personas> {  
    @Override  
    public List<Personas> findAll() {  
        Query consulta = getEm().createNamedQuery("Personas.findAll");  
        return consulta.getResultList();  
    }  
  
    public Personas buscarCedula(String cedula){  
        Query consulta = getEm().createNamedQuery("ConsultaCedula");  
        consulta.setParameter("cedula", cedula);  
        return (Personas) consulta.getSingleResult();  
    }  
}
```

El controlador personas hereda del controlador abstracto por lo que tenemos que poner todos los métodos que incluye esta clase en este caso es en find All el cual nos ayuda a encontrar todos los datos

de la base de datos esto lo obtenemos a través de consultas JPQL de la misma manera realizamos un método buscar por la cedula con consultas JPQL.

Controlador Bienes

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

import ec.edu.ups.modelo.Bienes;

import java.util.List;

import javax.persistence.Query;

/**
 *
 * @author NANCY
 */
public class ControladorBienes extends ControladorAbstracto<Bienes>{

    @Override
    public List<Bienes> findAll() {

        Query consulta = getEm().createNamedQuery("Bienes.findAll");

        return consulta.getResultList();

    }

}
```

El controlador Bienes hereda del controlador abstracto por lo que tenemos que poner todos los métodos que incluye esta clase en este caso es en find All el cual nos ayuda a encontrar todos los datos de la base de datos esto lo obtenemos a través de consultas JPQL

Controlador Hipoteca

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

import ec.edu.ups.modelo.Hipoteca;

import java.util.List;

import javax.persistence.Query;

/**
 *
 * @author NANCY
 */
public class ControladorHipoteca extends ControladorAbstracto<Hipoteca>{
```


@Override

```
public List<Hipoteca> findAll() {  
    Query consulta = getEm().createNamedQuery("hipoteca.findAll");  
    return consulta.getResultList();  
}  
}
```

El controlador Hipoteca hereda del controlador abstracto por lo que tenemos que poner todos los métodos que incluye esta clase en este caso es en find All el cual nos ayuda a encontrar todos los datos de la base de datos esto lo obtenemos a través de consultas JPQL

Paquete Utils

Clase JPAUtils

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

Esta clase nos ayuda controlar la persistencia de los objetos que van a una base de datos

Paquete modelo

Entity Class Personas

```
package ec.edu.ups.modelo;  
  
import java.io.Serializable;  
import java.util.Calendar;  
import javax.persistence.Column;  
import javax.persistence.Entity;  
import javax.persistence.GeneratedValue;
```

```

import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.NamedQuery;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;

/**
 *
 * @author NANCY
 */
@Entity
@NamedQuery(name = "Personas.findAll", query = "SELECT p FROM Personas p")
@NamedQuery(name = "ConsultaCedula", query = "SELECT p FROM Personas p where p.cedula = :cedula")
public class Personas implements Serializable {

    private static final long serialVersionUID = 1L;

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

    @Column
    private String cedula;

    @Column
    private String tipo;

    @Column
    private String nombre;

    @Column
    private String apellido;

    @Column
    private double sueldo;

    @Column
    @Temporal(TemporalType.DATE)

```

```
private Calendar fechaNacimiento;

public Personas() {
}

public Personas(String cedula, String tipo, String nombre, String apellido, double sueldo, Calendar
fechaNacimiento) {
    this.cedula = cedula;
    this.tipo = tipo;
    this.nombre = nombre;
    this.apellido = apellido;
    this.sueldo = sueldo;
    this.fechaNacimiento = fechaNacimiento;
}

public Personas(Long id, String cedula, String tipo, String nombre, String apellido, double sueldo, Calendar
fechaNacimiento) {
    this.id = id;
    this.cedula = cedula;
    this.tipo = tipo;
    this.nombre = nombre;
    this.apellido = apellido;
    this.sueldo = sueldo;
    this.fechaNacimiento = fechaNacimiento;
}

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 getTipo() {
    return tipo;
}

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

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 double getSueldo() {
    return sueldo;
}

public void setSueldo(double sueldo) {
    this.sueldo = sueldo;
}

public Calendar getFechaNacimiento() {
    return fechaNacimiento;
}

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

@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 Personas)) {
        return false;
    }
}
```

```

    Personas other = (Personas) 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 "Personas{" + "id=" + id + ", cedula=" + cedula + ", tipo=" + tipo + ", nombre=" + nombre + ", apellido="
+ apellido + ", sueldo=" + sueldo + ", fechaNacimiento=" + fechaNacimiento + '}';
}

}

```

Entity Class que nos va a ayudar con la creación de los objetos tipo persona en una tabla de base de datos

Entity Class Bienes

```

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.NamedQuery;

/**
 *
 * @author NANCY
 */
@Entity

```

```
@NamedQuery(name = "Bienes.findAll", query = "SELECT b FROM Bienes b")
```

```
public class Bienes implements Serializable {
```

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

```
    @Id
```

```
    @GeneratedValue(strategy = GenerationType.AUTO)
```

```
    private Long id;
```

```
    @Column
```

```
    private String Direccion;
```

```
    @Column
```

```
    private String calles;
```

```
    @Column
```

```
    private String numero;
```

```
    @Column
```

```
    private double precio;
```

```
    public Bienes() {
```

```
    }
```

```
    public Bienes(Long id, String Direccion, String calles, String numero, double precio) {
```

```
        this.id = id;
```

```
        this.Direccion = Direccion;
```

```
        this.calles = calles;
```

```
        this.numero = numero;
```

```
        this.precio = precio;
```

```
    }
```

```
    public Bienes(String Direccion, String calles, String numero, double precio) {
```

```
        this.Direccion = Direccion;
```

```
        this.calles = calles;
```

```
        this.numero = numero;
```

```
        this.precio = precio;
    }

    public Long getId() {
        return id;
    }

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

    public String getDireccion() {
        return Direccion;
    }

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

    public String getNumero() {
        return numero;
    }

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

    public double getPrecio() {
        return precio;
    }
}
```



```
public void setPrecio(double precio) {  
    this.precio = precio;  
}  
  
public String getCalles() {  
    return calles;  
}  
  
public void setCalles(String calles) {  
    this.calles = calles;  
}  
  
@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 Bienes)) {  
        return false;  
    }  
    Bienes other = (Bienes) 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 "Bienes{" + "id=" + id + ", Direccion=" + Direccion + ", calles=" + calles + ", numero=" + numero + ",
precio=" + precio + '}';

}

}

```

Entity Class que nos va a ayudar con la creación de los objetos tipo Bienes en una tabla de base de datos

Entity Class Hipoteca

```

package ec.edu.ups.modelo;

import java.io.Serializable;
import java.util.Calendar;
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.NamedQuery;
import javax.persistence.OneToOne;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;

/**
 *
 * @author NANCY
 */
@Entity
@NamedQuery(name = "hipoteca.findAll", query = "SELECT h FROM Hipoteca h")
public class Hipoteca implements Serializable {

```

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

```
@Id
```

```
@GeneratedValue(strategy = GenerationType.AUTO)
```

```
private Long id;
```

```
@OneToOne
```

```
@JoinColumn(name="Propietariofk")
```

```
private Personas propietario;
```

```
@OneToOne
```

```
@JoinColumn(name="Avalfk")
```

```
private Personas aval;
```

```
@OneToOne
```

```
@JoinColumn(name="Bienfk")
```

```
private Bienes bien;
```

```
@Column
```

```
@Temporal(TemporalType.DATE)
```

```
private Calendar fechaInicio;
```

```
@Column
```

```
@Temporal(TemporalType.DATE)
```

```
private Calendar fechaFinal;
```

```
@Column
```

```
private double monto;
```

```
@Column
```

```
private double pagoMensual;
```

```
@Column
```

```
private double Total;
```

```
@Column
```

```
private int meses;
```

```
public Hipoteca() {
```

```
}
```

```
public Hipoteca(Long id, Personas propietario, Personas aval, Bienes bien, Calendar fechaInicio, Calendar  
fechaFinal, double monto, double pagoMensual, double Total, int meses) {
```

```
this.id = id;

this.propietario = propietario;

this.aval = aval;

this.bien = bien;

this.fechaInicio = fechaInicio;

this.fechaFinal = fechaFinal;

this.monto = monto;

this.pagoMensual = pagoMensual;

this.Total = Total;

this.meses = meses;

}
```

```
public Hipoteca(Personas propietario, Personas aval, Bienes bien, Calendar fechaInicio, Calendar fechaFinal,
double monto, double pagoMensual, double Total, int meses) {
```

```
    this.propietario = propietario;

    this.aval = aval;

    this.bien = bien;

    this.fechaInicio = fechaInicio;

    this.fechaFinal = fechaFinal;

    this.monto = monto;

    this.pagoMensual = pagoMensual;

    this.Total = Total;

    this.meses = meses;

}
```

```
public Hipoteca(Personas propietario, Bienes bien, Calendar fechaInicio, Calendar fechaFinal, double monto,
double pagoMensual, double Total, int meses) {
```

```
    this.propietario = propietario;

    this.bien = bien;

    this.fechaInicio = fechaInicio;

    this.fechaFinal = fechaFinal;

    this.monto = monto;

    this.pagoMensual = pagoMensual;
```

```
this.Total = Total;

this.meses = meses;
}

public Long getId() {
    return id;
}

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

public Calendar getFechaInicio() {
    return fechaInicio;
}

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

public Calendar getFechaFinal() {
    return fechaFinal;
}

public void setFechaFinal(Calendar fechaFinal) {
    this.fechaFinal = fechaFinal;
}

public int getMeses() {
    return meses;
}
```

```
public void setMeses(int meses) {  
    this.meses = meses;  
}  
  
public double getMonto() {  
    return monto;  
}  
  
public void setMonto(double monto) {  
    this.monto = monto;  
}  
  
public double getPagoMensual() {  
    return pagoMensual;  
}  
  
public void setPagoMensual(double pagoMensual) {  
    this.pagoMensual = pagoMensual;  
}  
  
public double getTotal() {  
    return Total;  
}  
  
public void setTotal(double Total) {  
    this.Total = Total;  
}  
  
public Personas getPropietario() {  
    return propietario;  
}
```

```
public void setPropietario(Personas propietario) {  
    this.propietario = propietario;  
}  
  
public Personas getAval() {  
    return aval;  
}  
  
public void setAval(Personas aval) {  
    this.aval = aval;  
}  
  
public Bienes getBien() {  
    return bien;  
}  
  
public void setBien(Bienes bien) {  
    this.bien = bien;  
}  
  
@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;  
    }  
}
```

```

    }

    Hipoteca other = (Hipoteca) 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.Hipoteca[ id=" + id + " ]";

}

}

```

Entity Class Hipoteca esta clase nos ayuda con la creación de objetos tipo Hipoteca en la base de datos a su vez contiene relaciones one to one con las clases personas y bienes.

Paquete VISTA

Ventana Principal

```

package ec.edu.ups.vista;

import ec.edu.ups.controlador.ControladorBienes;
import ec.edu.ups.controlador.ControladorHipoteca;
import ec.edu.ups.controlador.ControladorPersonas;

/**
 *
 * @author NANCY
 */
public class VtnPrincipal extends javax.swing.JFrame {

    private ControladorPersonas controladorPersonas;

    private ControladorBienes controladorBienes;

    private ControladorHipoteca controladorHipoteca;

```



```
private VentanaPersonas ventanaPersonas;
private VentanaBienes ventanaBienes;
private VentanaHipoteca ventanaHipoteca;
/**
 * Creates new form VtnPrincipal
 */
public VtnPrincipal() {
    initComponents();

    controladorPersonas = new ControladorPersonas();
    controladorBienes = new ControladorBienes();
    controladorHipoteca = new ControladorHipoteca();

    ventanaPersonas = new VentanaPersonas(controladorPersonas);
    ventanaBienes = new VentanaBienes(controladorBienes);
    ventanaHipoteca = new VentanaHipoteca(controladorHipoteca, controladorPersonas, controladorBienes);
}
private void exitMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(0);
}

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

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

private void menuItemHipotecaActionPerformed(java.awt.event.ActionEvent evt) {
```

```

desktopPane.add(ventanaHipoteca);

ventanaHipoteca.setVisible(true);

}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
     * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
     */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(VtnPrincipal.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(VtnPrincipal.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(VtnPrincipal.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(VtnPrincipal.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
    }
    //</editor-fold>

```

```
/* Create and display the form */
java.awt.EventQueue.invokeLater(new Runnable() {
    public void run() {
        new VtnPrincipal().setVisible(true);
    }
});
}

// Variables declaration - do not modify
private javax.swing.JDesktopPane desktopPane;
private javax.swing.JMenuItem exitMenuItem;
private javax.swing.JMenu fileMenu;
private javax.swing.JMenuBar menuBar;
private javax.swing.JMenuItem menuItemHipoteca;
private javax.swing.JMenuItem menuItemPersona;
private javax.swing.JMenuItem menuItemSave;

// End of variables declaration
}

Ventana Personas
package ec.edu.ups.vista;

import ec.edu.ups.controlador.ControladorPersonas;
import ec.edu.ups.modelo.Personas;
import java.util.Calendar;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

/**
 *
 * @author NANCY
 */
public class VentanaPersonas extends javax.swing.JInternalFrame {

    private ControladorPersonas controladorPersonas;

    /**
     * Creates new form VentanaPersonas
     */
}
```

```

* @param controladorPersonas
*/

public VentanaPersonas(ControladorPersonas controladorPersonas) {

    initComponents();

    this.controladorPersonas = controladorPersonas;
}

public void limpiar(){
    txtCodigo.setText("0");
    txtCedula.setText("");
    cbxTipo.setSelectedIndex(0);
    txtNombre.setText("");
    txtApellido.setText("");
    txtSueldo.setText("");
}

public void cargarDatos() {
    DefaultTableModel modeloTabla = (DefaultTableModel) tblDatos.getModel();
    modeloTabla.setRowCount(0);

    for (Personas personas : controladorPersonas.findAll()) {
        Object[] rowData = {personas.getId(),personas.getCedula(),personas.getTipo(),
personas.getNombre(),personas.getApellido(),personas.getSueldo(),personas.getFechaNacimiento()};
        modeloTabla.addRow(rowData);
    }

    tblDatos.setModel(modeloTabla);
}

private void formInternalFrameActivated(javax.swing.event.InternalFrameEvent evt) {
    cargarDatos();
}

private void tblDatosMouseClicked(java.awt.event.MouseEvent evt) {
    int fila = tblDatos.getSelectedRow();
    long cod = (long) tblDatos.getValueAt(fila, 0);
    String cedula = (String)tblDatos.getValueAt(fila, 1);

```

```
String tipo = (String) tblDatos.getValueAt(fila, 2);
String nombre = (String) tblDatos.getValueAt(fila, 3);
String Apellido = (String) tblDatos.getValueAt(fila, 4);
double sueldo = (double) tblDatos.getValueAt(fila, 5);
Calendar fecha = (Calendar) tblDatos.getValueAt(fila, 6);

txtCodigo.setText(String.valueOf(cod));
txtCedula.setText(cedula);
cbxTipo.setSelectedItem(tipo);
txtNombre.setText(nombre);
txtApellido.setText(Apellido);
txtSueldo.setText(String.valueOf(sueldo));
JDate.setCalendar(fecha);
}

private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
    boolean resultado = false;
    long codigo=Long.parseLong(txtCodigo.getText());
    String cedula= txtCedula.getText();
    String tipo = (String) cbxTipo.getSelectedItem();
    String nombre= txtNombre.getText();
    String Apellido=txtApellido.getText();
    double sueldo = Double.parseDouble(txtSueldo.getText());
    Calendar fecha = JDate.getCalendar();

    if (cedula.isEmpty() || tipo.isEmpty() || nombre.isEmpty() || Apellido.isEmpty()) {
        JOptionPane.showMessageDialog(this, "Llene todos los campos solicitados");
    }else{
        if(controladorPersonas.buscar(codigo)!= null){
            System.out.println("Hola");
            Personas personas = new Personas(cedula,tipo,nombre,Apellido,sueldo, fecha);
            resultado =controladorPersonas.actualizar(personas);
        }
    }
}
```

```

    }else{
        Personas personas = new Personas(cedula, tipo, nombre, Apellido, sueldo, fecha);
        resultado = controladorPersonas.crear(personas);
    }
    limpiar();
    JOptionPane.showMessageDialog(this, "Operación : " + resultado);
    cargarDatos();
    dispose();
}
}

private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
    int eliminar = JOptionPane.showConfirmDialog(this, "¿Seguro quieres eliminar esta persona?");
    long codigo = Long.parseLong(txtCodigo.getText());

    Personas personas = controladorPersonas.buscar(codigo);
    if(eliminar == JOptionPane.YES_OPTION){
        boolean resultado = controladorPersonas.eliminar(personas);
        JOptionPane.showMessageDialog(this, "Operación : " + resultado);
        limpiar();
        cargarDatos();
        dispose();
    }
}

// Variables declaration - do not modify
private com.toedter.calendar.JDateChooser JDate;
private javax.swing.JButton btnEliminar;
private javax.swing.JButton btnGuardar;
private javax.swing.JComboBox<String> cbxTipo;
private javax.swing.JLabel jLabel1;

```

```
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JLabel jLabel6;
private javax.swing.JLabel jLabel7;
private javax.swing.JLabel jLabel8;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JTable tblDatos;
private javax.swing.JTextField txtApellido;
private javax.swing.JTextField txtCedula;
private javax.swing.JTextField txtCodigo;
private javax.swing.JTextField txtNombre;
private javax.swing.JTextField txtSueldo;

// End of variables declaration
}
```

Ventana Bienes

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

import ec.edu.ups.controlador.ControladorBienes;
import ec.edu.ups.modelo.Bienes;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

/**
 *
 * @author NANCY
 */
public class VentanaBienes extends javax.swing.JFrame {

    private ControladorBienes controladorBienes;

    /**
     * Creates new form VentanaBienes
     * @param controladorBienes
     */
    public VentanaBienes(ControladorBienes controladorBienes) {
```

```

initComponents();

this.controladorBienes= controladorBienes;
}

public void limpiar(){
    txtCodigo.setText("0");
    txtDireccion.setText("");
    txtCalles.setText("");
    txtNumero.setText("");
    txtPrecio.setText("");
}

public void cargarDatos() {
    DefaultTableModel modeloTabla = (DefaultTableModel) tblDatos.getModel();
    modeloTabla.setRowCount(0);
    for (Bienes bienes : controladorBienes.findAll()) {
        Object[] rowData = {bienes.getId(),bienes.getDireccion(),bienes.getCalles(),
bienes.getNumero(),bienes.getPrecio()};
        modeloTabla.addRow(rowData);
    }
    tblDatos.setModel(modeloTabla);
}

private void formInternalFrameActivated(javax.swing.event.InternalFrameEvent evt) {
    cargarDatos();
}

private void tblDatosMouseClicked(java.awt.event.MouseEvent evt) {
    int fila = tblDatos.getSelectedRow();
    long cod = (long) tblDatos.getValueAt(fila, 0);
    String direccion = (String)tblDatos.getValueAt(fila, 1);
    String calles = (String) tblDatos.getValueAt(fila, 2);
    String numero = (String) tblDatos.getValueAt(fila, 3);
    double precio = (double) tblDatos.getValueAt(fila, 4);
    txtCodigo.setText(String.valueOf(cod));
    txtDireccion.setText(direccion);
    txtCalles.setText(calles);
}

```



```
txtNumero.setText(numero);

txtPrecio.setText(String.valueOf(precio));
}

private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
    boolean resultado = false;

    long codigo=Long.parseLong(txtCodigo.getText());

    String direccion= txtDireccion.getText();

    String calles= txtCalles.getText();

    String numero=txtNumero.getText();

    double presio = Double.parseDouble(txtPrecio.getText());

    if (direccion.isEmpty() ||calles.isEmpty()|| numero.isEmpty()) {
        JOptionPane.showMessageDialog(this, "Llene todos los campos solicitados");
    }else{
        if(controladorBienes.buscar(codigo)!= null){
            System.out.println("Hola");

            Bienes bienes = new Bienes(direccion, calles, numero, presio);

            resultado =controladorBienes.actualizar(bienes);
        }else{
            Bienes bienes = new Bienes(direccion, calles, numero, presio);

            resultado =controladorBienes.crear(bienes);
        }
        limpiar();

        JOptionPane.showMessageDialog(this, "Operación : " + resultado);

        cargarDatos();

        dispose();
    }
}

private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
    int eliminar = JOptionPane.showConfirmDialog(this, "¿Seguro quieres eliminar este Bien?");

    long codigo=Long.parseLong(txtCodigo.getText());
```

```

        Bienes bienes= controladorBienes.buscar(codigo);
        if(eliminar == JOptionPane.YES_OPTION){
            boolean resultado = controladorBienes.eliminar(bienes);
            JOptionPane.showMessageDialog(this, "Operación : " + resultado);
            limpiar();
            cargarDatos();
            dispose();
        }
    }

    // Variables declaration - do not modify
    private javax.swing.JButton btnEliminar;
    private javax.swing.JButton btnGuardar;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTable tblDatos;
    private javax.swing.JTextField txtCalles;
    private javax.swing.JTextField txtCodigo;
    private javax.swing.JTextField txtDireccion;
    private javax.swing.JTextField txtNumero;
    private javax.swing.JTextField txtPrecio;

    // End of variables declaration
}

```

Ventana Hipotecas

```

package ec.edu.ups.vista;

import ec.edu.ups.controlador.ControladorBienes;
import ec.edu.ups.controlador.ControladorHipoteca;

```

```
import ec.edu.ups.controlador.ControladorPersonas;
import ec.edu.ups.modelo.Bienes;
import ec.edu.ups.modelo.Hipoteca;
import ec.edu.ups.modelo.Personas;
import java.util.concurrent.TimeUnit;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
/**
 *
 * @author NANCY
 */
public class VentanaHipoteca extends javax.swing.JInternalFrame {
    private ControladorPersonas controladorPersonas;
    private ControladorBienes controladorBienes;
    private ControladorHipoteca controladorHipoteca;
    private double pagoTotal;
    /**
     * Creates new form VentanaHipoteca
     * @param controladorHipoteca
     * @param controladorPersonas
     * @param controladorBienes
     */
    public VentanaHipoteca(ControladorHipoteca controladorHipoteca, ControladorPersonas controladorPersonas,
        ControladorBienes controladorBienes) {
        initComponents();
        this.controladorHipoteca = controladorHipoteca;
        this.controladorPersonas= controladorPersonas;
        this.controladorBienes = controladorBienes;
    }
    public void limpiar(){
        txtCedula.setText("");
        txtTipo.setText("");
        txtNombre.setText("");
    }
}
```

```

txtApellido.setText("");
txtSueldo.setText("");
txtCedula2.setText("");
txtTipo2.setText("");
txtNombre2.setText("");
txtApellido1.setText("");
txtSueldo1.setText("");
txtDireccion.setText("");
txtCalles.setText("");
txtCodigo.setText("");
txtNumero.setText("");
txtPrecio.setText("");
txtMonto.setText("");
txtMeses.setText("");
txtPagoM.setText("");
}

public void cargarDatos(int n, double monto) {
    DefaultTableModel modeloTabla = (DefaultTableModel) tblDatos.getModel();

    double pago = 0;
    double interes = 0;
    double amortiguacion = monto/n;
    double saldo = monto;
    modeloTabla.setRowCount(0);
    for (int i = 0; i < n; i++) {
        saldo=saldo-amortiguacion;
        interes= saldo*0.0075;
        pago=amortiguacion+interes;
        pagoTotal=pagoTotal+pago;
        Object[] rowData = {i, pago, interes, amortiguacion, saldo};
        modeloTabla.addRow(rowData);
    }

    tblDatos.setModel(modeloTabla);
}

```

```
private void btnBuscarPropietarioActionPerformed(java.awt.event.ActionEvent evt) {

    String cedula= txtCedula.getText();

    Personas personas = controladorPersonas.buscarCedula(cedula);

    if(personas.getTipo().equals("Propietario")){

        txtTipo.setText(personas.getTipo());

        txtNombre.setText(personas.getNombre());

        txtApellido.setText(personas.getApellido());

        txtSueldo.setText(String.valueOf(personas.getSueldo()));

        JDate.setCalendar(personas.getFechaNacimiento());

    }

}

private void btnBuscarAvalActionPerformed(java.awt.event.ActionEvent evt) {

    String cedula= txtCedula2.getText();

    Personas personas = controladorPersonas.buscarCedula(cedula);

    if(personas.getTipo().equals("Aval")){

        txtTipo2.setText(personas.getTipo());

        txtNombre2.setText(personas.getNombre());

        txtApellido1.setText(personas.getApellido());

        txtSueldo1.setText(String.valueOf(personas.getSueldo()));

        JDate1.setCalendar(personas.getFechaNacimiento());

    }

}

private void btnBuscarBienActionPerformed(java.awt.event.ActionEvent evt) {

    long codigo=Long.parseLong(txtCodigo.getText());

    Bienes bienes= controladorBienes.buscar(codigo);

    txtDireccion.setText(bienes.getDireccion());

    txtCalles.setText(bienes.getCalles());

    txtNumero.setText(bienes.getNumero());

    txtPrecio.setText(String.valueOf(bienes.getPrecio()));

}

private void btnCalculoActionPerformed(java.awt.event.ActionEvent evt) {
```

```

double monto = Double.parseDouble(txtMonto.getText());

long          DiferenciaMiliSegundos=          Math.abs(jDateFinal.getCalendar().getTimeInMillis()-
jDateInicio.getCalendar().getTimeInMillis());

long dias=TimeUnit.MILLISECONDS.toDays(DiferenciaMiliSegundos);

int numeroMeses= (int) (dias/30);

txtMeses.setText(String.valueOf(numeroMeses));

System.out.println(dias);

cargarDatos(numeroMeses, monto);

txtPagoM.setText(String.valueOf(pagoTotal));
}

private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {

    int numeroMeses= Integer.valueOf(txtMeses.getText());

    double monto = Double.parseDouble(txtMonto.getText());

    boolean resultado = false;

    Personas propietario = controladorPersonas.buscarCedula(txtCedula.getText());

    Bienes bien = controladorBienes.buscar(Long.parseLong(txtCodigo.getText()));

    double suma= bien.getPrecio()+ propietario.getSueldo();

    System.out.println(suma);

    System.out.println(pagoTotal);

    if(pagoTotal<=suma){

        Hipoteca hipoteca = new Hipoteca(propietario, bien, jDateInicio.getCalendar(), jDateFinal.getCalendar(),
monto, pagoTotal, pagoTotal, numeroMeses);

        resultado =controladorHipoteca.crear(hipoteca);

    }else{

        if(txtNombre2.getText().equals("")){

            JOptionPane.showMessageDialog(this, "Ingrese Aval" );

        }else{

            Personas aval = controladorPersonas.buscarCedula(txtCedula2.getText());

            Hipoteca hipoteca= new Hipoteca(propietario, aval, bien, jDateInicio.getCalendar(),
jDateFinal.getCalendar(), monto, pagoTotal,pagoTotal, numeroMeses);

            resultado =controladorHipoteca.crear(hipoteca);

        }

    }

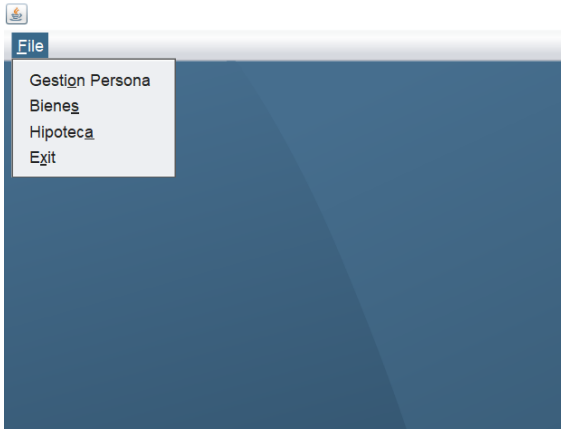
    JOptionPane.showMessageDialog(this, "Operación : " + resultado);
}

```

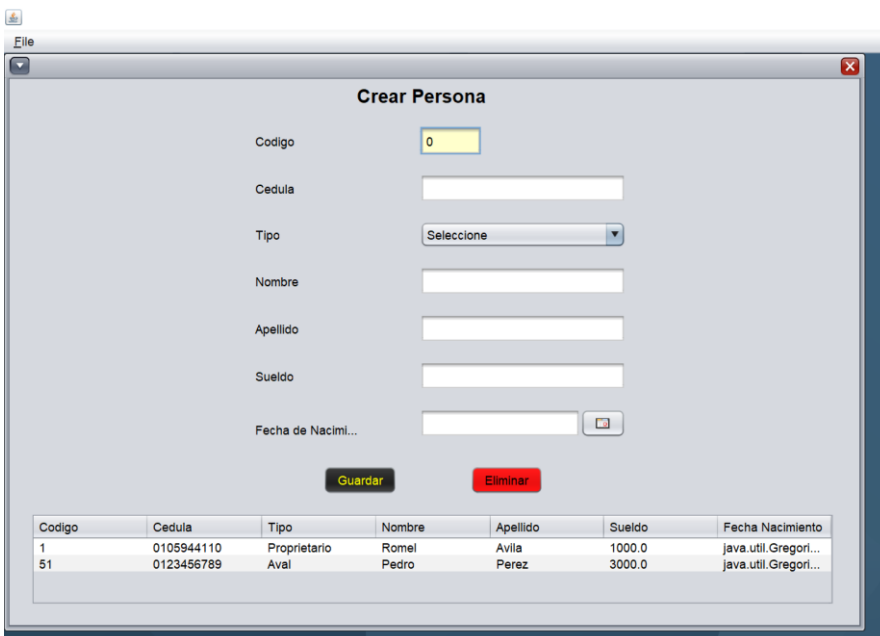
```
limpiar();
```

```
}
```

Ejecucion



Ventana al ejecutar el programa



Codigo	Cedula	Tipo	Nombre	Apellido	Sueldo	Fecha Nacimiento
1	0105944110	Proprietario	Romel	Avila	1000.0	java.util.Gregori...
51	0123456789	Aval	Pedro	Perez	3000.0	java.util.Gregori...

Ventana al presionar gestión personas en donde podemos crear eliminar o actualizar a las personas y después se cargarán en la tabla

Bibiendas

Codigo: 0

Direccion: Monay

Calles: calle Alvaro Ptricio

Numero: 123

Precio: 100000

Guardar **Eliminar**

Codigo	Direccion	Calles	Numero	Precio
101	Ricaurte	calle.Cordillera ...	000	90000.0

Ventana Bibienda en donde se gestiona toda la parte de la vivienda para poder crear el objeto en la base de datos.

File

Cedula del propietario

Buscar

Tipo

Nombre

Apellido

Sueldo

Fecha de Nacimi...

Bienes

Codigo

Buscar

Direccion

Calles

Numero

Precio

Cedula del Aval

Buscar

Tipo

Nombre

Apellido

Sueldo

Fecha de Nacimi...

Monto

Fecha Inicio

Fecha Final

Meses

Pago total

Hipoteca

Mes	Cuota	Interes	Amortizacion	Saldo

Calculo

Guardar

Ventana Hipoteca

File

Cedula del propietario: 0105944110 **Buscar**

Tipo: Proprietario

Nombre: Romel

Apellido: Avila

Sueldo: 1000.0

Fecha de Nacimi...: 21 feb. 2020

Bienes

Codigo: 351 **Buscar**

Direccion: Monay

Calles: calle Alvaro Ptricio

Numero: 123

Precio: 100000.0

Cedula del Aval: **Buscar**

Tipo:

Nombre:

Apellido:

Sueldo:

Fecha de Nacimi...:

Monto: 100000

Fecha Inicio: 1 ene. 2021

Fecha Final: 1 dic. 2022

Meses: 23

Pago total: 429250.0

Hipoteca

Mes	Cuota	Interes	Amortizacion	Saldo
0	5065.21739...	717.391304...	4347.82608...	95652.1739...
1	5032.60869...	684.782608...	4347.82608...	91304.3478...
2	5000.0	652.173913...	4347.82608...	86956.5217...
3	4967.39130...	619.565217...	4347.82608...	82608.6956...

Calculo **Guardar**

Ventana al llenar el campo de la cedula de propietario y buscar también al poner el código del bien y buscar y finalmente al poner el monto a pedir la fecha de inicio y fin y así calcula el total a pagar más la tabla de amortización que podemos observar con el valor a cancelar cada mes. La tabla de amortización esta calculada con el método alemán

File

Cedula del propietario: 0105944110 **Buscar**

Tipo: Proprietario

Nombre: Romel

Apellido: Avila

Sueldo: 1000.0

Fecha de Nacimi...: 21 feb. 2020

Bienes

Codigo: 351 **Buscar**

Direccion: Monay

Calles: calle Alvaro Ptricio

Numero: 123

Precio: 100000.0

Cedula del Aval: **Buscar**

Tipo:

Nombre:

Apellido:

Sueldo:

Fecha de Nacimi...:

Monto: 100000

Fecha Inicio: 1 ene. 2021

Fecha Final: 1 dic. 2022

Meses: 23

Pago total: 429250.0

Hipoteca

Mes	Cuota	Interes	Amortizacion	Saldo
0	5065.21739...	717.391304...	4347.82608...	95652.1739...
1	5032.60869...	684.782608...	4347.82608...	91304.3478...
2	5000.0	652.173913...	4347.82608...	86956.5217...
3	4967.39130...	619.565217...	4347.82608...	82608.6956...

Calculo **Guardar**

Message

i Ingrese Aval

OK

Si ponemos guardar y como el pago total es mayor al de la propiedad y al sueldo del propietario el sistema nos solicita un aval o no se podrá guardar

File

Cedula del propietario: 0105944110 **Buscar**

Tipo: Proprietario

Nombre: Romel

Apellido: Avila

Sueldo: 1000.0

Fecha de Naci...: 21 feb. 2020

Bienes

Codigo: 351 **Buscar**

Direccion: Money

Calles: calle Alvaro Ptricio

Numero: 123

Precio: 100000.0

Cedula del Aval: 0123456789 **Buscar**

Tipo: Aval

Nombre: Pedro

Apellido:

Sueldo:

Fecha de Naci...:

Monto: 100000

Fecha Inicio: 1 ene. 2021

Fecha Final: 1 dic. 2022

Meses: 23 **Calculo**

Pago total: 537500.0 **Guardar**

Hipoteca

Mes	Cuota	Interes	Amortizacion	Saldo
0	5065.21739...	717.391304...	4347.82608...	95652.1739...
1	5032.60869...	684.782608...	4347.82608...	91304.3478...
2	5000.0	652.173913...	4347.82608...	86956.5217...
3	4967.39130...	619.565217...	4347.82608...	82608.6956...

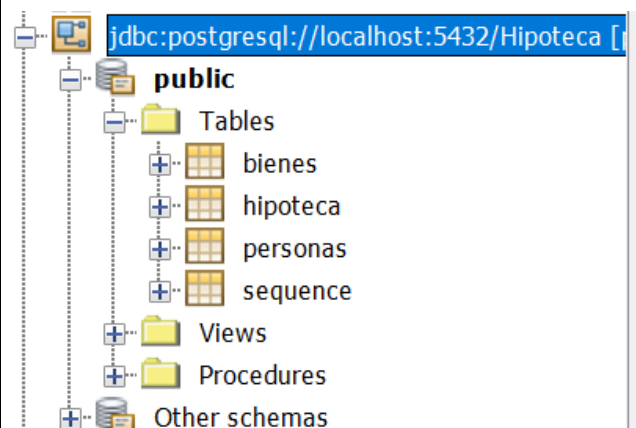
Message

Operación : true

OK

Windows taskbar: Escribe aquí para buscar, 15:13 31/1/2021

Una vez ingresado el aval ya nos deja guardar los datos




Podemos comprobar que se encuentra creada la base de datos del programa.

SELECT * FROM "public".hi... x

Max. rows: 100 | Fetched Rows: 5 |

#	id	total	fechafinal	fechainicio	meses	monto	pagomensual	avalfk	bienfk	propietariofk
1	151	10.000	2022-07-08	2021-01-01	18	90.000	10.000	<NULL>	101	1
2	201	10.000	2022-07-01	2021-01-01	18	90.000	10.000	<NULL>	101	1
3	251	10.000	2022-07-01	2021-01-01	18	90.000	10.000	<NULL>	101	1
4	301	95.737,5	2022-07-01	2021-01-01	18	90.000	95.737,5	51	101	1
5	352	537.500	2022-12-01	2021-01-01	23	100.000	537.500	51	351	1

Tabal Hipoteca

	Computación	Docente: Diego Quisi Peralta
	Programacion Aplicada	Período Lectivo: Septiembre 2020 – Febero 2021

RESULTADO(S) OBTENIDO(S):

Generar un programa utilizando una programación con JPA para guardar datos en una base de datos haciendo una optimización del código

CONCLUSIONES:

En conclusión, esta prueba nos ayudo a manejar de una mejor manera las sentencias utilizadas en el JPA y en JPQL para hacer búsquedas y guardar datos en una base de datos conectada a un proyecto de NetBeans

RECOMENDACIONES:

No hay Recomendaciones

Estudiantes: Romel Ávila

Firma:

