



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

Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación

Universidad Politécnica Salesiana

Vicerrectorado Docente

| | |
|----------------------------|--|
| Código del Formato: | GUIA-PRL-001 |
| Versión: | VF1.0 |
| Elaborado por: | Directores de Área del Conocimiento Integrantes Consejo Académico |
| Fecha de elaboración: | 2016/04/01 |
| Revisado por: | Consejo Académico |
| Fecha de revisión: | 2016/04/06 |
| Aprobado por: | Lauro Fernando Pesántez Avilés Vicerrector Docente |
| Fecha de aprobación: | 2016/14/06 |
| Nivel de confidencialidad: | Interno |

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

Descripción General

Propósito


El propósito del presente documento es definir un estándar para elaborar documentación de guías de práctica de laboratorio, talleres o centros de simulación de las Carreras de la Universidad Politécnica Salesiana, con la finalidad de lograr una homogenización en la presentación de la información por parte del personal académico y técnico docente.


Alcance

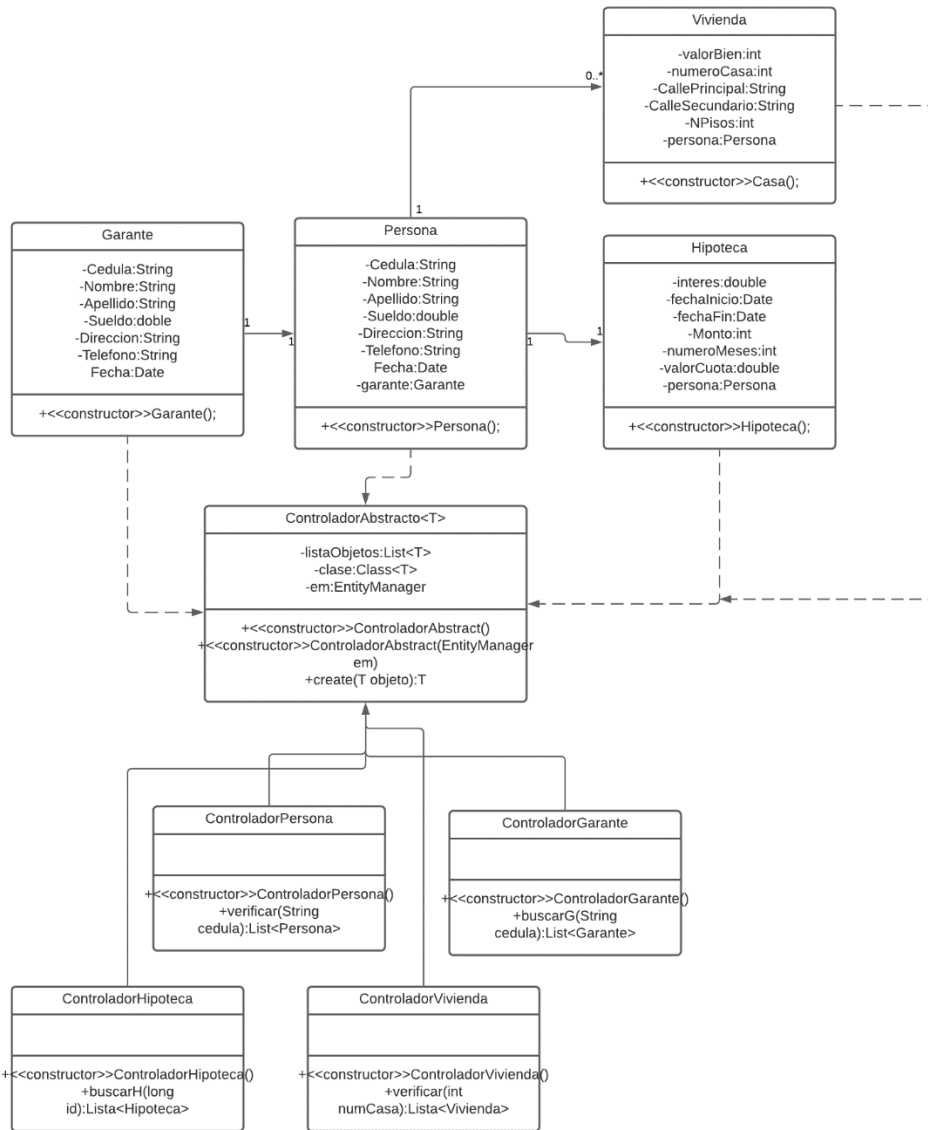
El presente estándar será aplicado a toda la documentación referente a informes de prácticas de laboratorio, talleres o centros de simulación de las Carreras de la Universidad Politécnica Salesiana.

Formatos

- Formato de Guía de Práctica de Laboratorio / Talleres / Centros de Simulación – para Docentes
- Formato de Informe de Práctica de Laboratorio / Talleres / Centros de Simulación – para Estudiantes

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

| | | | | | |
|---|---|---|--|--|--|
|  | | | FORMATO DE INFORME DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA ESTUDIANTES | | |
| CARRERA: Computación | | | ASIGNATURA: Programacion Aplicada | | |
| NRO. PRÁCTICA: | 3 | TÍTULO PRÁCTICA: Prueba Practica JPA | | | |
| OBJETIVO ALCANZADO: <ul style="list-style-type: none"> • Ampliar conocimientos sobre base de datos usando JPA • Entender el funcionamiento de base de datos • Comprender y analizar los enlaces entre tablas, base de datos | | | | | |
| ACTIVIDADES DESARROLLADAS | | | | | |
| 1. Creamos el Diagrama UML para tener una idea clara de que atributos crear y métodos a usar | | | | | |



2. Creamos el Proyecto con el nombre "Prueba_Practica_JPA"


Prueba_Practica_JPA

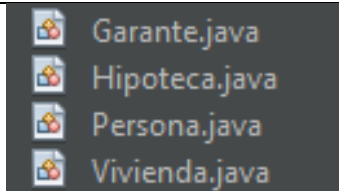
- Creamos los paquetes

```

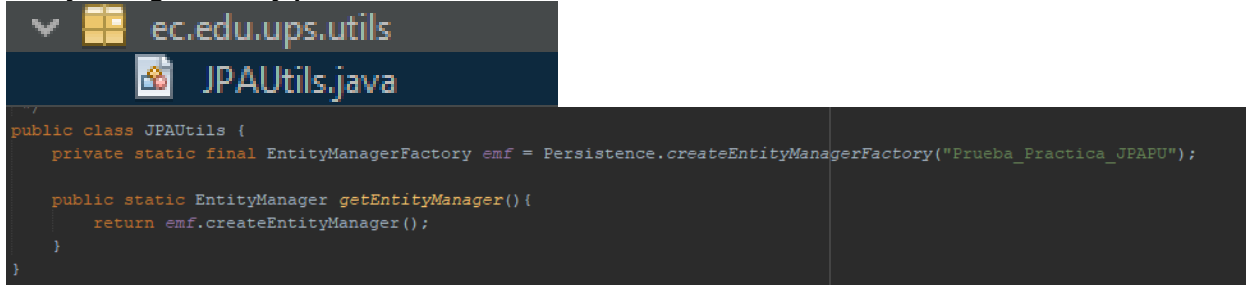
> META-INF
> ec.edu.ups.controlador
> ec.edu.ups.modelo
> ec.edu.ups.utils
> ec.edu.ups.vista
  
```

- En lugar de crear java class para los modelos creamos entity class, que crea las tablas, con su id y primary key por defecto

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



3. En el paquete utils creamos una clase llamada JPAUtils, en esa instanciamos el EntityManagerFactory para llamar la conexión con la base de datos



4. En cada clase creada en el paquete los atributos se crearan y hay que asignarles que son columnas, con su respectivo nombre.

- Tabla Persona


```

@Entity
@NamedQuery(name = "buscarCedula", query = "Select p from Persona p where p.cedula = :cedula")
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 = "Sueldo")
    private double sueldo;
    @Column(name = "Direccion")
    private String direccion;
    @Column(name = "Telefono")
    private String telefono;
    @Column(name = "Fecha_Nacimiento")
    @Temporal(javax.persistence.TemporalType.DATE)
    private Date fechaNac;
    @OneToMany(mappedBy = "persona", cascade = CascadeType.ALL)
    private List<Vivienda> listaViviendas;
    @OneToOne
    @JoinColumn(name = "Garante")
    private Garante garante;

    public Persona() {
        listaViviendas = new ArrayList<>();
    }
}

```

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

```

}

public Persona(String cedula, String nombre, String Apellido, double sueldo,
String direccion, String telefono, Date fechaN) {
    this.cedula = cedula;
    this.nombre = nombre;
    this.Apellido = Apellido;
    this.sueldo = sueldo;
    this.direccion = direccion;
    this.telefono = telefono;
    this.fechaNac = fechaN;
    listaViviendas = new ArrayList<>();
}

public Persona(String cedula, String nombre, String Apellido, double sueldo,
String direccion, String telefono, Date fechaN, List<Vivienda> listaViviendas) {
    this.cedula = cedula;
    this.nombre = nombre;
    this.Apellido = Apellido;
    this.sueldo = sueldo;
    this.direccion = direccion;
    this.telefono = telefono;
    this.fechaNac = fechaN;
    listaViviendas = new ArrayList<>();
}


// public Persona(String cedula, String nombre, String Apellido, double sueldo,
String direccion, String telefono, Date fechaNac, List<Vivienda> listaViviendas,
Garante garante) {
//     this.cedula = cedula;
//     this.nombre = nombre;
//     this.Apellido = Apellido;
//     this.sueldo = sueldo;
//     this.direccion = direccion;
//     this.telefono = telefono;
//     this.fechaNac = fechaNac;
//     this.listaViviendas = listaViviendas;
//     this.garante = garante;
//     listaViviendas = new ArrayList<>();
// }

public Long getId() {
    return id;
}

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

public String getCedula() {
    return cedula;
}

```

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

```

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 double getSueldo() {
    return sueldo;
}

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

public String getDireccion() {
    return direccion;
}

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

public String getTelefono() {
    return telefono;
}


public void setTelefono(String telefono) {
    this.telefono = telefono;
}

public Date getFechaNac() {
    return fechaNac;
}

public void setFechaNac(Date fechaNac) {
    this.fechaNac = fechaNac;
}

public Garante getGarante() {
    return garante;
}

```

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

```

}

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

public List<Vivienda> getListaViviendas() {
    return listaViviendas;
}

public void setListaViviendas(List<Vivienda> listaViviendas) {
    this.listaViviendas = listaViviendas;
}


@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;
}

    • Tabla Vivienda
@Entity
@NamedQuery(name = "buscarNumCasa", query = "Select v from Vivienda v where
v.numCasa= :numCasa")
public class Vivienda implements Serializable {

    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;
    @Column(name = "ValorBien")
    private int valorBien;
    @Column(name = "NumeroCasa")
    private int numCasa;
    @Column(name = "Calle_Principal")
    private String calleP;
    @Column(name = "Calle_Secundaria")
    private String calleS;

```


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

```

@Column(name = "Numero_de_Pisos")
private int numPisos;
@ManyToOne
@JoinColumn(name = "fk_persona")
private Persona persona;
//////////
// @OneToOne(mappedBy = "Vivienda", cascade = CascadeType.ALL)
// @PrimaryKeyJoinColumn
// private Hipoteca hipoteca;
//////////
public Vivienda() {
}

public Vivienda(int valorBien, int numCasa, String calleP, String calleS, int
numPisos, Persona persona) {
    this.valorBien = valorBien;
    this.numCasa = numCasa;
    this.calleP = calleP;
    this.calleS = calleS;
    this.numPisos = numPisos;
    this.persona = persona;
}

public Long getId() {
    return id;
}

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

public int getValorBien() {
    return valorBien;
}

public void setValorBien(int valorBien) {
    this.valorBien = valorBien;
}


public int getNumCasa() {
    return numCasa;
}

public void setNumCasa(int numCasa) {
    this.numCasa = numCasa;
}

public String getCalleP() {
    return calleP;
}

public void setCalleP(String calleP) {
    this.calleP = calleP;
}

```

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

```

public String getCalleS() {
    return calleS;
}

public void setCalleS(String calleS) {
    this.calleS = calleS;
}

public int getNumPisos() {
    return numPisos;
}

public void setNumPisos(int numPisos) {
    this.numPisos = numPisos;
}


public Persona getPersona() {
    return persona;
}

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

////////////////////////////////////
//      public Hipoteca getHipoteca() {
//          return hipoteca;
//      }
//
//      public void setHipoteca(Hipoteca hipoteca) {
//          this.hipoteca = hipoteca;
//      }
////////////////////////////////////
@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 Vivienda)) {
        return false;
    }
    Vivienda other = (Vivienda) object;
    if ((this.id == null && other.id != null) || (this.id != null &&
!this.id.equals(other.id))) {
        return false;
    }
    return true;
}

```

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

```

}


//      @Override
//      public String toString() {
//          return "ec.edu.ups.modelo.Vivienda[ id=" + id + " ]";
//      }
//
//      @Override
//      public String toString() {
//          return "Vivienda{" + "id=" + id + ", valorBien=" + valorBien + ", numCasa="
+ numCasa + ", calleP=" + calleP + ", calleS=" + calleS + ", numPisos=" + numPisos +
", persona=" + persona + '}';
//      }
    }
    • Tabla Hipoteca
@Entity
@NamedQuery(name = "buscarIdHipo", query = "Select h from Hipoteca h where h.id=
:id")
public class Hipoteca implements Serializable {

    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;
    @Column(name = "Interes")
    private double interes;
    @Column(name = "Fecha_Inicio")
    private Date fechaIn;
    @Column(name = "Fecha_Fin")
    private Date fechaFin;
    @Column(name = "Monto")
    private int monto;
    @Column(name = "NumeroMeses")
    private int numMeses;
    @Column(name = "valorCuota")
    private double valorCu;
    //////////////////////////////////////
    @OneToOne
    @JoinColumn(name = "id_Persona")
    private Persona persona;
    //////////////////////////////////////

    public Hipoteca() {
    }

    public Hipoteca(double interes, Date fechaIn, Date fechaFin, int monto, int
numMeses, double valorCuota, Persona persona) {
        this.interes = interes;
        this.fechaIn = fechaIn;
        this.fechaFin = fechaFin;
        this.monto = monto;
        this.numMeses = numMeses;
        this.valorCu = valorCuota;
        this.persona = persona;
    }

```

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

```

}

public Long getId() {
    return id;
}

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

public double getInteres() {
    return interes;
}

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

public Date getFechaIn() {
    return fechaIn;
}

public void setFechaIn(Date fechaIn) {
    this.fechaIn = fechaIn;
}

public Date getFechaFin() {
    return fechaFin;
}

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

public int getMonto() {
    return monto;
}


public void setMonto(int monto) {
    this.monto = monto;
}

public int getNumMeses() {
    return numMeses;
}

public void setNumMeses(int numMeses) {
    this.numMeses = numMeses;
}

public double getValorCu() {
    return valorCu;
}

```

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

```

public void setValorCu(double valorCu) {
    this.valorCu = valorCu;
}

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 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 + " ]";
// }

@Override
public String toString() {
    return "Hipoteca{" + "id=" + id + ", interes=" + interes + ", fechaIn=" +
fechaIn + ", fechaFin=" + fechaFin + ", monto=" + monto + ", numMeses=" + numMeses +
", persona=" + persona + '}';
}


    }

    • Tabla Garante

@Entity
@NamedQuery(name = "buscarCedulaG", query = "Select g from Garante g where g.cedula =
:cedula")
public class Garante 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 = "Cedula")
private String cedula;
@Column(name = "Nombre")
private String nombre;
@Column(name = "Apellido")
private String Apellido;
@Column(name = "Sueldo")
private double sueldo;
@Column(name = "Direccion")
private String direccion;
@Column(name = "Telefono")
private String telefono;
@Column(name = "Fecha_Nacimiento")
@Temporal(javax.persistence.TemporalType.DATE)
private Date fechaNac;

public Garante() {
}

public Garante(String cedula, String nombre, String Apellido, double sueldo,
String direccion, String telefono, Date fechaNac) {
    this.cedula = cedula;
    this.nombre = nombre;
    this.Apellido = Apellido;
    this.sueldo = sueldo;
    this.direccion = direccion;
    this.telefono = telefono;
    this.fechaNac = fechaNac;
}

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;
}

```

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

```

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 String getDireccion() {
    return direccion;
}

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

public String getTelefono() {
    return telefono;
}

public void setTelefono(String telefono) {
    this.telefono = telefono;
}


public Date getFechaNac() {
    return fechaNac;
}

public void setFechaNac(Date fechaNac) {
    this.fechaNac = fechaNac;
}

@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

```

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

```

set
    if (!(object instanceof Garante)) {
        return false;
    }
    Garante other = (Garante) 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.Garante[ id=" + id + " ]";
// }

@Override
public String toString() {
    return "Garante{" + "id=" + id + ", cedula=" + cedula + ", nombre=" + nombre
+ ", Apellido=" + Apellido + ", sueldo=" + sueldo + ", direccion=" + direccion + ",
telefono=" + telefono + ", fechaNac=" + fechaNac + '}';
}

}

```

5. En el paquete controlador estarán todas las clases que se llaman para crear y agregar datos a sus respectivas tablas y llamar métodos que fueron necesarios para realizar consultas, la clase ControladorAbstract será la clase padre y las son las subclases que heredaran los métodos, eso es programación genérica.

- ControladorAbstract(Clase Padre)

```

public abstract class ControladorAbstract<T> {
    private List<T> listaObjetos;
    private Class<T> clase;
    private EntityManager em;

    public ControladorAbstract() {
        listaObjetos = new ArrayList<>();
        Type t = getClass().getGenericSuperclass();
        ParameterizedType pt = (ParameterizedType) t;
        clase = (Class) pt.getActualTypeArguments()[0];
        em = JPAUtils.getEntityManager();
    }

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

    public T create(T objeto){
        em.getTransaction().begin();
        em.persist(objeto);
    }
}

```



```
        em.getTransaction().commit();
        listaObjetos.add(objeto);
        return objeto;
    }

    // public boolean delete(T objeto){
    //     em.getTransaction().begin();
    //     em.remove(em.merge(objeto));
    //     em.getTransaction().commit();
    //     listaObjetos.remove(objeto);
    //     return true;
    // }
    //
    // public T update(T objeto){
    //     em.getTransaction().begin();
    //     objeto = em.merge(objeto);
    //     em.getTransaction().commit();
    //     this.findAll();
    //     return objeto;
    // }
    //
    // public T read(Object id){
    //     return (T) em.find(clase, id);
    // }
    //
    // public List<T> findAll(){
    //     return em.createQuery("Select t from " + clase.getSimpleName() +
    "t").getResultList();
    // }

    public List<T> getListaObjetos() {
        return listaObjetos;
    }


    public void setListaObjetos(List<T> listaObjetos) {
        this.listaObjetos = listaObjetos;
    }

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

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

    public EntityManager getEm() {
        return em;
    }

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

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

```

}

    • ControladorPersona
public class ControladorPersona extends ControladorAbstract<Persona> {
    public Persona obtenerPersona;

    public ControladorPersona() {
        super();
    }

    public List<Persona> verificar(String cedula) {
        Query buscar = getEm().createNamedQuery("buscarCedula");
        buscar.setParameter("cedula", cedula);
        var p = buscar.getResultList();
        if (p.isEmpty()) {
            System.out.println("null");
            return null;
        }else{
            System.out.println("lleno");
            return p;
        }
    }

    public Persona buscar(String cedula) {
        return (Persona) getEm().createNamedQuery("buscarCedula")
            .setParameter("cedula", cedula)
            .getSingleResult();
    }
}

    • ControladorVivienda
public class ControladorVivienda extends ControladorAbstract<Vivienda>{


    public ControladorVivienda() {
        super();
    }

    public List<Vivienda> verificar(int numCasa) {
        Query buscar = getEm().createNamedQuery("buscarNumCasa");
        buscar.setParameter("numCasa", numCasa);
        var p = buscar.getResultList();
        if (p.isEmpty()) {
            System.out.println("null");
            return null;
        }else{
            System.out.println("lleno");
            return p;
        }
    }
}

    • ControladorHipoteca
public class ControladorHipoteca extends ControladorAbstract<Hipoteca>{

    public ControladorHipoteca() {
        super();
    }
}

```

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

```

}

public List<Hipoteca> buscarH(long id){
    Query buscar = getEm().createNamedQuery("buscarIdHipo");
    buscar.setParameter("id", id);
    var h = buscar.getResultList();
    if (h.isEmpty()) {
        System.out.println("null");
        return null;
    }else{
        System.out.println("lleno");
        return h;
    }
}

    }
    • ControladorGarante
public class ControladorGarante extends ControladorAbstact<Garante>{

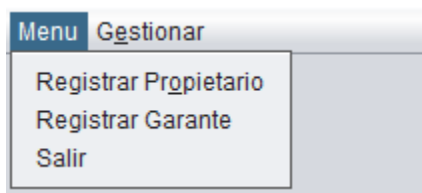
    public ControladorGarante() {
        super();
    }

    public List<Garante> buscarG(String cedula){
        Query buscar = getEm().createNamedQuery("buscarCedulaG");
        buscar.setParameter("cedula", cedula);
        var p = buscar.getResultList();
        if (p.isEmpty()) {
            System.out.println("null");
            return null;
        }else{
            System.out.println("lleno");
            return p;
        }
    }
}


```

6. Vista

- Se tendrá un menú para registrar a un propietario, un garante, si aun no se ha registrado un propietario el garante a un no podrá ser creado.



- VntRegistrar: esta venta nos permitirá crear al propietario, al presionar el botón Registrar los datos de cada caja de texto ingresado se guardara en 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 | | |



- **VntVivienda:** Aquí el propietario ingresará sus casas que tiene, ese tiene una relación de uno a muchos, un propietario puede tener varias casas, primero debe ingresar su cedula y verificar si se encuentra en la base de datos, después de eso se tendrá la posibilidad de crear o registrar todas sus propiedades

Registrar Vivienda

Cedula del Propietario

Nombre del Propietario

Apellido del Propietario

Valor del Bien

Numero de casa

Calle principal

Calle Secundaria

Numero de pisos

- **VntPrestamo:** Igual que la ventana registrar vivienda, se tendrá que validar que ese propietario existe y podrá registrar sus hipoteca, en el caso de que el valor de pago cada mes sea mas alto que su sueldo pedirá que se registre un garante, después repetirá el mismo proceso pero en lugar de registrar a otro garante buscara a su garante ingresado

Propietario

Cedula del Propietario

Nombre del Propietario

Apellido del Propietario

Hipoteca

ID Hipoteca

Interes Anual

Fecha de Inicio

Fecha de Fin

Monto

Numero de fechas

Cuota Mensual

Seleccion que propiedad a continuacion desea Hipotecar

Numero de Casa

Valor del Bien

| Numero Casa | Calle Principal | Calle Secundaria | Numero de Pisos | Valor del Bien |
|-------------|-----------------|------------------|-----------------|----------------|
| 1 | via al arenal | via a nulti | 1 | 70000 |
| 2 | monay | monay | 2 | 10000 |
| 3 | totems | totems | 3 | 180000 |

Garante

Cedula del Garante

Nombre del Propietario

Apellido del Propietario

Registrar Garante

Cedula Propietario

0107543027

Cedula

0107543019

Nombre

Julian

Apellido

Zambrano

Sueldo

900

Direccion

Nulti

Telefono

4044381

Fecha Nacimiento

27

06

2000

REGISTRAR

CANCELAR

Garante

Cedula del Garante

0107543019

Nombre del Propietario

Julian

Apellido del Propietario

Zambrano

BUSCAR GARANTE

REGISTRAR GARANTE

- **VntTabla:** Cuando ya tenga creado su hipoteca nos dará el id, ese id hay que ingresar para visualizar la tabla de amortización

Tabla Amortizacion

Verificar Hipoteca

Ingrese Id para buscar Hipoteca:

Cedula Propietario:

Propietario:

| N° Cuota | Saldo Inicial | Cuota | Intereses | Amortizacion Capital | Saldo final |
|----------|---------------|---------|-----------|----------------------|-------------|
| 1 | 20.000 | 476,376 | 143,376 | 333 | 19.667 |
| 2 | 19.667 | 476,376 | 140,989 | 333 | 19.334 |
| 3 | 19.334 | 473,989 | 138,602 | 333 | 19.001 |
| 4 | 19.001 | 471,602 | 136,215 | 333 | 18.668 |
| 5 | 18.668 | 469,215 | 133,827 | 333 | 18.335 |
| 6 | 18.335 | 466,827 | 131,44 | 333 | 18.002 |
| 7 | 18.002 | 464,44 | 129,053 | 333 | 17.669 |
| 8 | 17.669 | 462,053 | 126,666 | 333 | 17.336 |
| 9 | 17.336 | 459,666 | 124,279 | 333 | 17.003 |
| 10 | 17.003 | 457,279 | 121,891 | 333 | 16.670 |
| 11 | 16.670 | 454,891 | 119,504 | 333 | 16.337 |
| 12 | 16.337 | 452,504 | 117,117 | 333 | 16.004 |
| 13 | 16.004 | 450,117 | 114,73 | 333 | 15.671 |
| 14 | 15.671 | 447,73 | 112,342 | 333 | 15.338 |
| 15 | 15.338 | 445,342 | 109,955 | 333 | 15.005 |
| 16 | 15.005 | 442,955 | 107,568 | 333 | 14.672 |

7.

N.

RESULTADO(S) OBTENIDO(S):

- Se aprendió sobre base de datos usando JPA
- Se obtuvo el necesario conocimiento sobre hipotecas y tablas de amortización

CONCLUSIONES:


- Entender el funcionamiento de enlazar tablas
- Comprender el funcionamiento de base de datos usando JPA

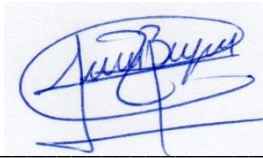
RECOMENDACIONES:

- Tener los debidos conocimientos para realizar la practica
- Consultar al profesor si se tienen dudas sobre JPA o relaciones entre tablas

Nombre de estudiante: Bryam Parra

Resolución CS N° 076-04-2016-04-20

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



Firma de estudiante: _____