



Prueba 3

29/01/2021

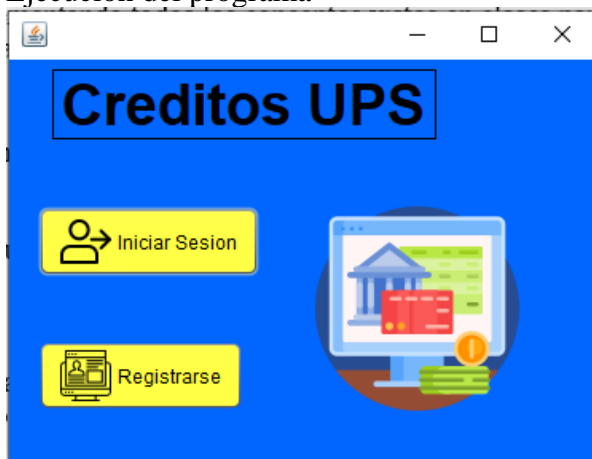
Objetivo:

- Consolidar los conocimientos adquiridos en clase sobre JPA.

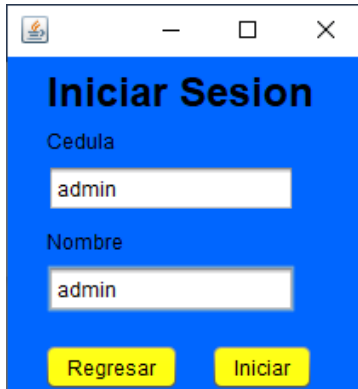
Enunciado:

Realizar un sistema implementando todos los conceptos vistos en clases para gestionar la hipoteca de las casas con las siguientes características:

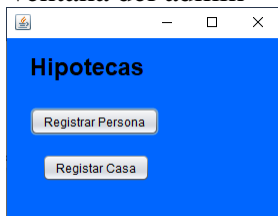
Ejecución del programa



Inicio de sesion del admin



Ventana del admin



Registro de una persona y de una casa
Persona



Prueba 3

29/01/2021

Registro

Cedula: 0106154370

Nombre: Andrea

Apellido: Calle

Fecha de Nacimiento: 01/09/2004

Direccion: Arsenio Ullauri y Jose Serrano

Salario: 300.00

Regresar Registrar

Casa

Registro de Casa

Direccion: 24 de Mayo y Calle sin Nombre

Precio: 50000.00

Registrar Regresar

Inicio de sesión de una persona

Iniciar Sesion

Cedula: 0106154362

Nombre: Andres

Regresar Iniciar

Ventana de la persona



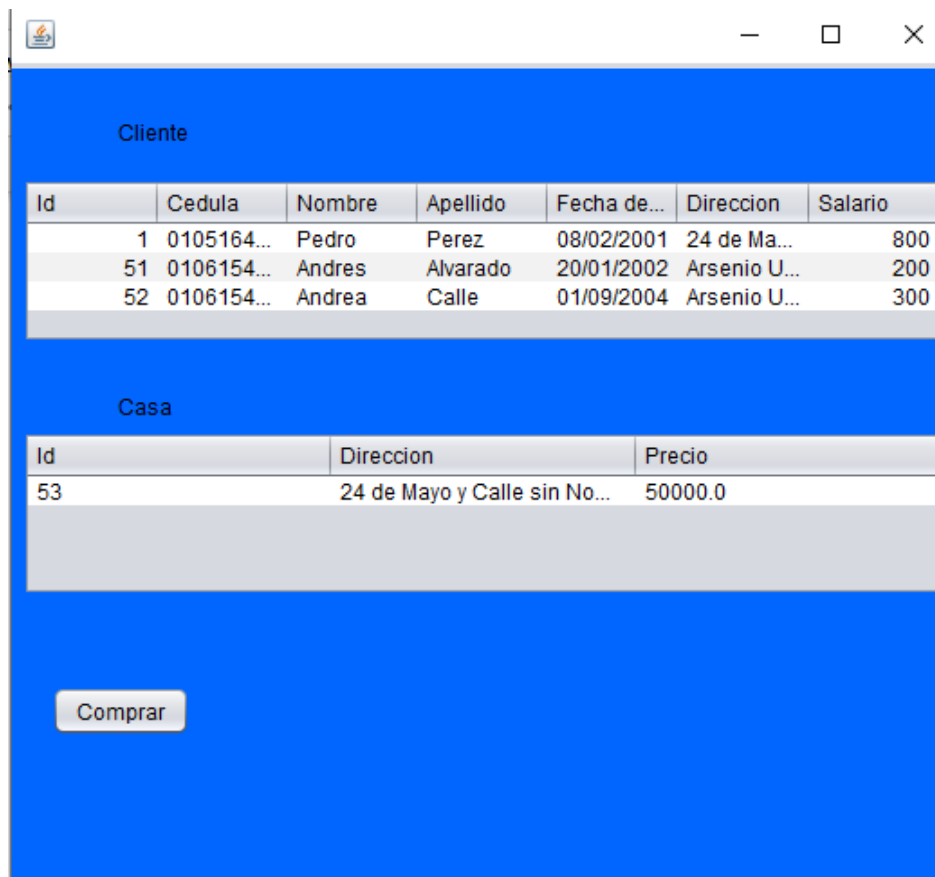
Prueba 3

29/01/2021



- Las personas compran casas y se convierten en propietarios.

Compra de la casa



Base de Datos

Antes de la Compra



Prueba 3

29/01/2021

SELECT * FROM "public".ca... X

Max. rows: 100 | Fetched Rows: 1 | Match

#	id	direccion	precio	fk_propietario
1	53	24 de Mayo y Calle sin Nombre	50.000	<NULL>

SELECT * FROM "public".pe... X

Max. rows: 100 | Fetched Rows: 3 | Matching Rows:

#	id	apellido	cedula	direccion	fechanacimiento
1		1 Perez	0105164362	24 de Mayo y calle sin nombre	08/02/2001
2		51 Alvarado	0106154362	Arsenio Ullauri y Jose Serrano	20/01/2002
3		52 Calle	0106154370	Arsenio Ullauri y Jose Serrano	01/09/2004

Después de la Compra

Cliente

Id	Cedula	Nombre	Apellido	Fecha de...	Direccion	Salario
1	0105164...	Pedro	Perez	08/02/2001	24 de Ma...	800
51	0106154...	Andres	Alvarado	20/01/2002	Arsenio U...	200
52	0106154...	Andrea	Calle	01/09/2004	Arsenio U...	300

Casa

Id	Direccion	Precio
53	24 de Mayo y Calle sin No...	50000.0

Comprar

SELECT * FROM "public".ca... X

Max. rows: 100 | Fetched Rows: 1 | Match

#	id	direccion	precio	fk_propietario
1	53	24 de Mayo y Calle sin Nombre	50.000	1



Prueba 3

29/01/2021

- Para pagarlas es habitual que el propietario formalice un préstamo hipotecario con una entidad bancaria.

The screenshot shows a Java application window with a blue background. At the top, there is a table with the following data:

Id	Cedula	Nombre	Apellido	Fecha de ...	Direccion	Salario
1	01051643...	Pedro	Perez	08/02/2001	24 de May...	800
51	01061543...	Andres	Alvarado	20/01/2002	Arsenio U...	200
52	01061543...	Andrea	Calle	01/09/2004	Arsenio U...	300

Below the table, there is a form with the following fields and controls:

- Cedula:** Input field with value "0105164362".
- Nombre:** Input field with value "Pedro".
- Apellido:** Input field with value "Perez".
- Fecha de nacimiento:** Input field with value "08/02/2001".
- Direccion:** Input field with value "y calle sin nombre".
- Salario:** Input field with value "800.0".
- importe:** Input field (empty).
- Plazo (años):** Input field (empty).
- Pagos:** Input field with value "800.0".
- Cuotas:** Input field (empty).
- Buttons:** "Calcular", "Guardar", and "Cargar informacion".

crédito

The screenshot shows the same Java application window as before, but with the following updated values:

- importe:** Input field with value "50000.00".
- Plazo (años):** Input field with value "7".
- Pagos:** Input field with value "800.0".
- Cuotas:** Input field with value "620,89".
- Buttons:** "Calcular", "Guardar", and "Cargar informacion".

- El banco toma la casa en forma de aval en caso de impago de las mensualidades.



Prueba 3

29/01/2021

SELECT * FROM "public"."ba..." X

Max. rows: 100 | Fetched Rows: 1 | Matching Rows:

#	id	avals	cuotas	hipotecas	importe	pagos
1		101	1	620,89	1	50.000

- 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).

Id Cedula Nombre Apellido Fecha de... Direccion Salario

1	0105164...	Pedro	Perez	08/02/2001	24 de Ma...	800
51	0106154...	Andres	Alvarado	20/01/2002	Arsenio U...	200
52	0106154...	Andrea	Calle	01/09/2004	Arsenio U...	300
102	0105164...	Pedro	Perez	08/02/2001	24 de Ma...	800

Cedula: 0106154362

Nombre: Andres

Apellido: Alvarado

Fecha de nacimiento: 20/01/2002

Direccion: iuri y Jose Serrano

Salario: 200.0

Importe: 5000.00

Cuotas: 167.75

Guardar

Cargar informacion

Mensaje: No puedes tener el prestamo tu salario es menor a las cuotas

Aceptar

- 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.

Cedula: 0105164362

Nombre: Pedro

Apellido: Perez

Fecha de nacimiento: 08/02/2001

Direccion: y calle sin nombre

Salario: 800.0

Cargar informacion



Prueba 3

29/01/2021

- El capital de la hipoteca se ajusta teniendo en cuenta el valor de tasación de la casa y los datos de dirección.

importe: 50000.00

Plazo (años): 7

Pagos: 800.0

Calcular

Cuotas: 620,89

Guardar

- JPA
El abstract controlador para usar el JPA

```
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;  
        clase = (Class) pt.getActualTypeArguments()[0];  
        em = JPAUtils.getEntityManager();  
    }  
    public AbstractControlador(EntityManager em) {  
        lista = new ArrayList();  
        Type t =getClass().getGenericSuperclass();  
        ParameterizedType pt=(ParameterizedType)t;  
        clase = (Class) pt.getActualTypeArguments()[0];  
        this.em = em;  
    }  
    public E crear(E objeto) throws Exception{  
        this.validar(objeto);  
        em.getTransaction().begin();  
        em.persist(objeto);  
        em.getTransaction().commit();  
        lista.add(objeto);  
        return objeto;  
    }  
}
```



Prueba 3

29/01/2021

```
public boolean eliminar(E objeto){
    em.getTransaction().begin();
    em.remove(em.merge(objeto));
    em.getTransaction().commit();
    lista.remove(objeto);
    return true;
}
public E actualizar(E objeto)throws Exception{
    this.validar(objeto);
    em.getTransaction().begin();
    objeto=em.merge(objeto);
    em.getTransaction().commit();
    lista= this.buscarTodo();
    return objeto;
}
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> 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;
    }
```




Prueba 3

29/01/2021

```
}  
Un ejemplo de los entity class aplicados  
@Entity  
@NamedQuery(name = "consultaCedula", query = "Select p from Persona p where p.cedula= :cedula")  
@NamedQuery(name = "consultaNombre", query = "Select p from Persona p where  
p.nombre= :nombre")  
public class Persona implements Serializable {  
  
    private static final long serialVersionUID = 1L;  
    @Id  
    @GeneratedValue(strategy = GenerationType.AUTO)  
    private Long id;  
    //partes de la tabla  
    @Column  
    @NotNull  
    private String cedula;  
    @Column  
    private String nombre;  
    @Column  
    private String apellido;  
    @Column  
    private String fechaNacimiento ;  
    @Column  
    private String direccion;  
    @Column  
    private Double salario;  
    //relacion  
    @OneToMany(mappedBy = "persona", cascade = CascadeType.ALL)  
  
    private List<Casa> lsitaCasa;  
  
    @ManyToOne  
    @JoinColumn(name = "fk_banco")  
    private Banco banco;  
    //getters y setters  
  
    public Persona() {  
    }  
  
    public Banco getBanco() {  
        return banco;  
    }  
  
    public void setBanco(Banco banco) {  
        this.banco = banco;  
    }  
}
```



Prueba 3

29/01/2021

```
public Persona(Long id, String cedula, String nombre, String apellido, String fechaNacimiento,
String direccion, Double salario, Banco banco) {
    this.id = id;
    this.cedula = cedula;
    this.nombre = nombre;
    this.apellido = apellido;
    this.fechaNacimiento = fechaNacimiento;
    this.direccion = direccion;
    this.salario = salario;
    this.banco = banco;
}
```

```
public Persona(Long id, String cedula, String nombre, String apellido, String fechaNacimiento,
String direccion, Double salario) {
    this.id = id;
    this.cedula = cedula;
    this.nombre = nombre;
    this.apellido = apellido;
    this.fechaNacimiento = fechaNacimiento;
    this.direccion = direccion;
    this.salario = salario;
}
```

```
public List<Casa> getLsitaCasa() {
    return lsitaCasa;
}
```

```
public void setLsitaCasa(List<Casa> lsitaCasa) {
    this.lsitaCasa = lsitaCasa;
}
```

```
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;
}
```



Prueba 3

29/01/2021

```
public String getFechaNacimiento() {  
    return fechaNacimiento;  
}  
  
public void setFechaNacimiento(String fechaNacimiento) {  
    this.fechaNacimiento = fechaNacimiento;  
}  
  
public String getApellido() {  
    return apellido;  
}  
  
public void setApellido(String apellido) {  
    this.apellido = apellido;  
}  
  
public String getDireccion() {  
    return direccion;  
}  
  
public void setDireccion(String direccion) {  
    this.direccion = direccion;  
}  
  
public Double getSalario() {  
    return salario;  
}  
  
public void setSalario(Double salario) {  
    this.salario = salario;  
}  
  
public Long getId() {  
    return id;  
}  
  
public void setId(Long id) {  
    this.id = id;  
}  
  
@Override  
public int hashCode() {  
    int hash = 0;
```



Prueba 3

29/01/2021

```
hash += (id != null ? id.hashCode() : 0);  
return hash;  
}
```

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

```
@Override  
public String toString() {  
    return "Persona{" + "id=" + id + ", cedula=" + cedula + ", nombre=" + nombre + ",  
fechaNacimiento=" + fechaNacimiento + ", apellido=" + apellido + ", direccion=" + direccion + ",  
salario=" + salario + '}';  
}
```

```
}
```

- Excepciones: 10%
Las excepciones con el método validar donde se aplico



Programación Aplicada

Tema: Base de Datos Java.

Prueba 3

29/01/2021

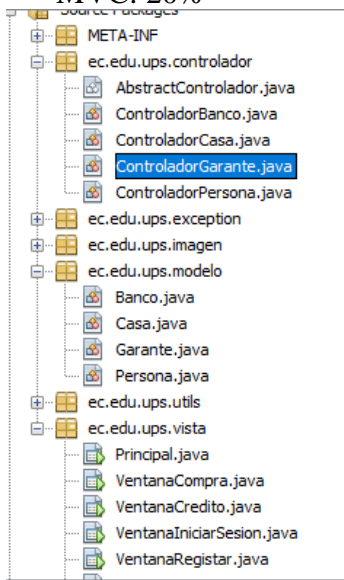
```
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30

import javax.persistence.Query;

/**
 * @author Andres
 */
public class ControladorPersona extends AbstractControlador<Persona> {

    public boolean validar(Persona objeto) throws MiExcepcion {
        String cedula = objeto.getCedula();
        int c;
        int suma = 0;
        int contador;
        int resta = 0;
        boolean bandera = false;
        if(cedula.length() != 10) {
            throw new MiExcepcion("La cedula debe tener 10 digitos ");
        }
        for(int i = 0; i < cedula.length(); i++) {
```

• MVC: 20%



Diagrama



Prueba 3

29/01/2021

