

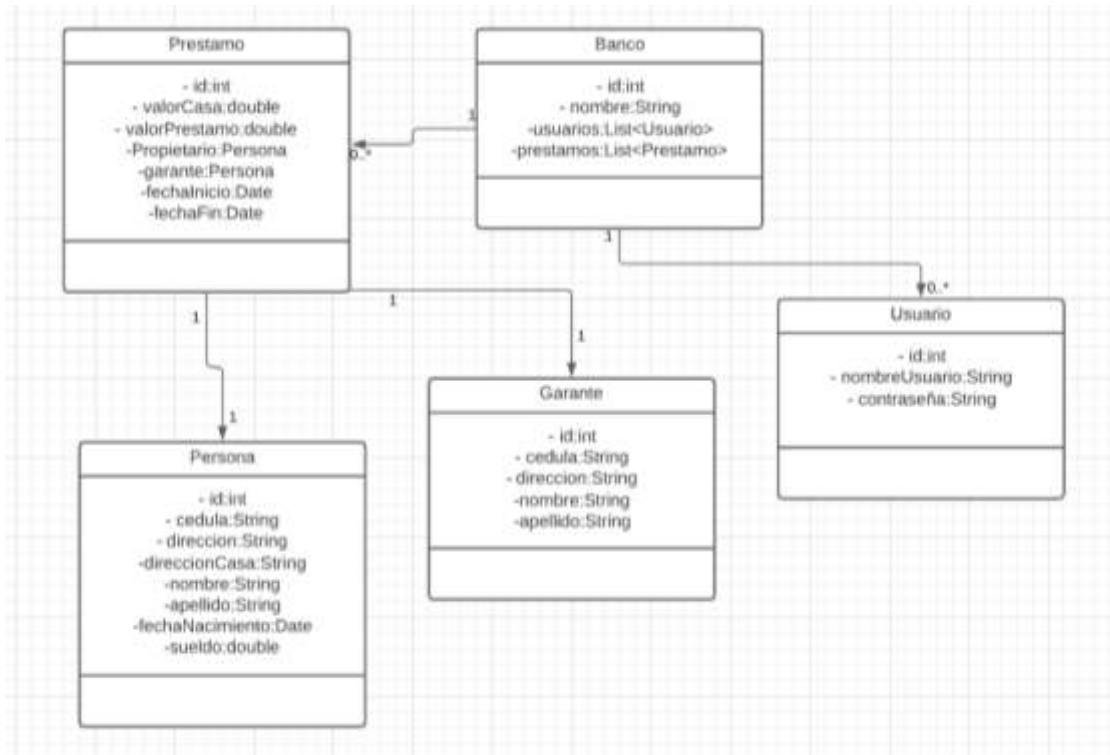
	<b>VICERRECTORADO DOCENTE</b>	<b>Código:</b> GUIA-PRL-001
	CONSEJO ACADÉMICO	<b>Aprobación:</b> 2016/04/06
<b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

		<b>PRÁCTICA DE LABORATORIO</b>	
<b>CARRERA:</b> COMPUTACIÓN		<b>ASIGNATURA:</b> PROGRAMACIÓN APLICADA	
<b>NRO. PRÁCTICA:</b>	3	<b>TÍTULO PRÁCTICA:</b> Prueba Practica 3	
<b>OBJETIVO ALCANZADO:</b> <ul style="list-style-type: none"> <li>Consolidar los conocimientos adquiridos en clase sobre JPA.</li> </ul>			
<b>ACTIVIDADES DESARROLLADAS</b>			
<p>1. <b>Realizar un sistema implementando todos los conceptos vistos en clases para gestionar la hipoteca de las casas con las siguientes características:</b></p> <ul style="list-style-type: none"> <li>Las personas compran casas y se convierten en propietarios.</li> <li>Para pagarlas es habitual que el propietario formalice un préstamo hipotecario con una entidad bancaria.</li> <li>El banco toma la casa en forma de aval en caso de impago de las mensualidades.</li> <li>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).</li> <li>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.</li> <li>El capital de la hipoteca se ajusta teniendo en cuenta el valor de tasación de la casa y los datos de dirección.</li> <li>Toda hipoteca se formaliza detallando el capital, el interés (8,99 - 16,99%) y la duración (fecha de inicio y fecha de fin).</li> <li>A partir de estos datos se calcula el importe de cada mensualidad para el total del tiempo que pide el préstamo.</li> <li>No es necesario guardar los datos del banco, pero si un sistema de autenticación.</li> <li>Generar los datos con el sistema de amortización alemán [1].</li> </ul>			

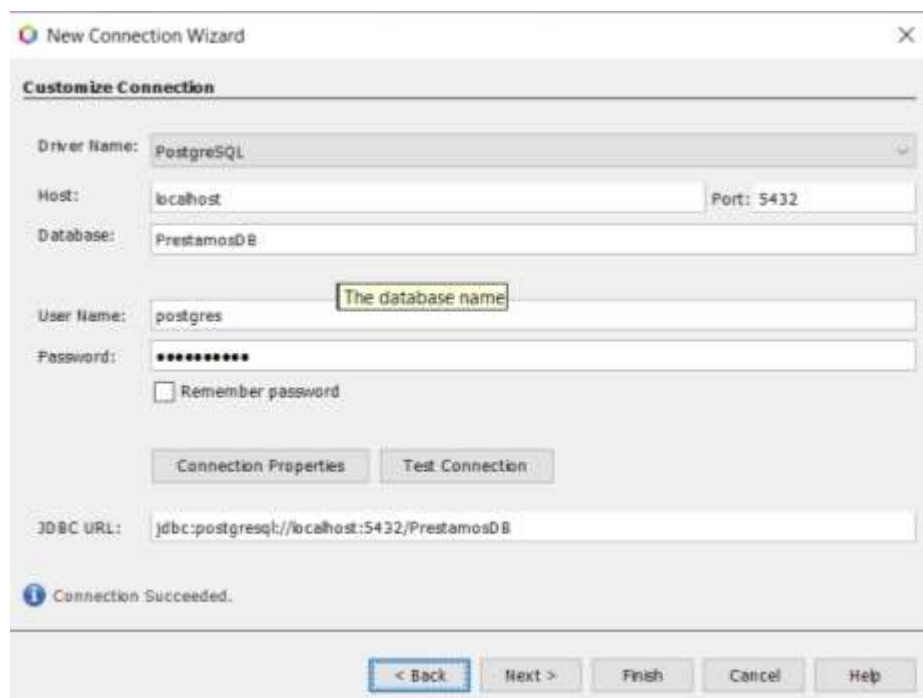
	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

## • Desarrollo de la Aplicación

1. Se desarrollo un diagrama de clases:




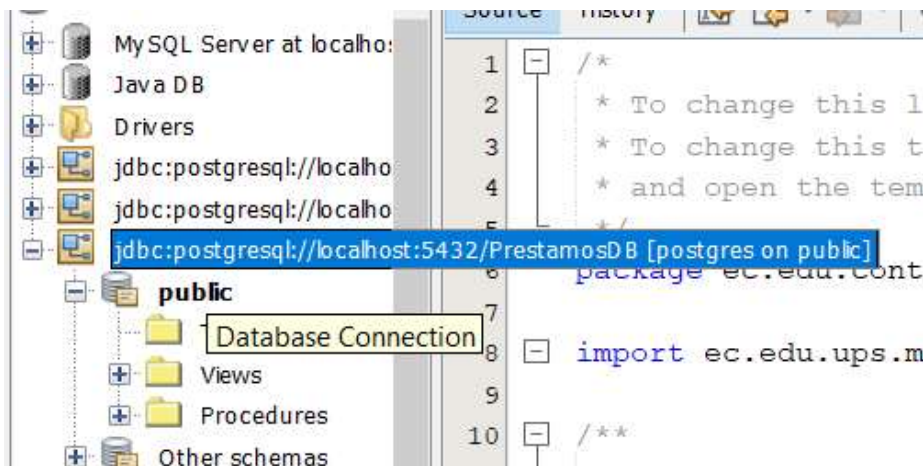
2. Se creo una nueva base de datos con el nombre de PrestamosDB



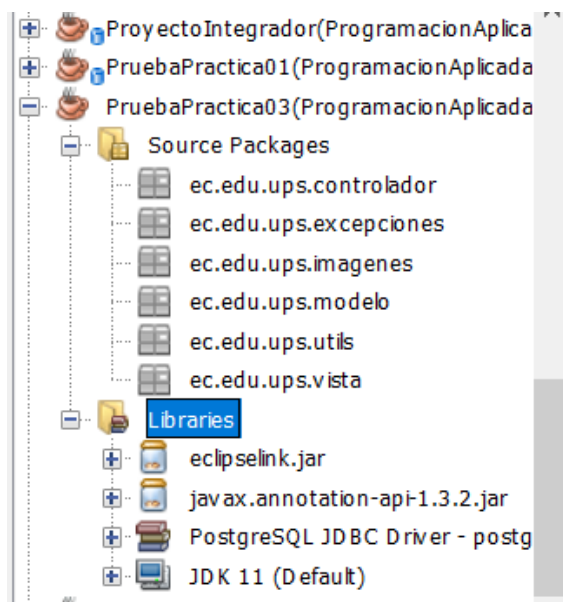
The screenshot shows the 'New Connection Wizard' dialog box for PostgreSQL. The 'Customize Connection' tab is active. The fields are filled as follows:

- Driver Name: PostgreSQL
- Host: localhost
- Port: 5432
- Database: PrestamosDB
- User Name: postgres
- Password: (masked with dots)
- Remember password: ☐
- Connection Properties: (button)
- Test Connection: (button)
- JDBC URL: jdbc:postgresql://localhost:5432/PrestamosDB
- Connection Succeeded: (message)
- Navigation buttons: < Back, Next >, Finish, Cancel, Help

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



- Se creó un nuevo proyecto con el nombre PruebaPractica3(ProgramacionAplicada) en el cual se importaron las siguientes librerías:



- Dentro del paquete ec.edu.ups.utils se creó la clase JPAUtils:

```


public class JPAUtils {

    private static final EntityManagerFactory emf = Persistence.createEntityManagerFactory("PrestamosJPAU");

    public static EntityManager getEntityManager() {
        return emf.createEntityManager();
    }

}

```

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

5. Dentro del paquete `ec.edu.ups.modelo` se crearon las entidades definidas en el diagrama de clases

- Banco:

```
@Entity
public class Banco implements Serializable {

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

    @Column
    private String nombre;

    @OneToMany(mappedBy = "banco", cascade = CascadeType.ALL)
    private List<Usuario> usuarios;

    @OneToMany(mappedBy = "banco", cascade = CascadeType.ALL)
    private List<Prestamo> prestamos;
```

- Garante:

```
@Entity
public class Garante implements Serializable {

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

    @Column
    private String cedula;

    @Column
    private String nombre;

    @Column
    private String apellido;

    @Column
    private String direccion;
```

- Persona:

```
@Entity
public class Persona implements Serializable {

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

    @Column
    private String cedula;

    @Column
    private String direccion;

    @Column
    private String direccionCasa;

    @Column
    private String nombre;

    @Column
    private String apellido;

    @Column
    @Temporal(TemporalType.DATE)
    private Date fechaNacimiento;

    @Column
    private double sueldo;
```

- Usuario:

```
@Entity
public class Usuario implements Serializable {

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

    @Column
    private String nombreUsuario;

    @Column
    private String contraseña;

    @ManyToOne
    @JoinColumn(name = "fk_banco")
    private Banco banco;
```

- Préstamo:

```
@Entity
public class Prestamo implements Serializable {

    private static final long serialVersionUID = 1L;

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

    @Column
    private double valorCasa;

    @Column
    private double valorPrestamo;

    @Column
    @Temporal(TemporalType.DATE)
    private Date fechaInicio;


    @Column
    @Temporal(TemporalType.DATE)
    private Date fechaFin;

    @OneToOne
    @JoinColumn(name = "id")
    private Persona persona;

    @OneToOne
    @JoinColumn(name = "id")
    private Garante garante;

    @ManyToOne
    @JoinColumn(name = "fk_banco")
    private Banco banco;
```

6. Dentro del paquete ec.edu.ups.controlador se desarrolló un controlador genérico para lo cual se usó los conocimientos adquiridos sobre programación genérica y reflexión :

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

```

public abstract class ControladorGenerico<E> {

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

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


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

```

7. Finalmente se diseño la interfaz para el usuario:

- Ventana de logeo:




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

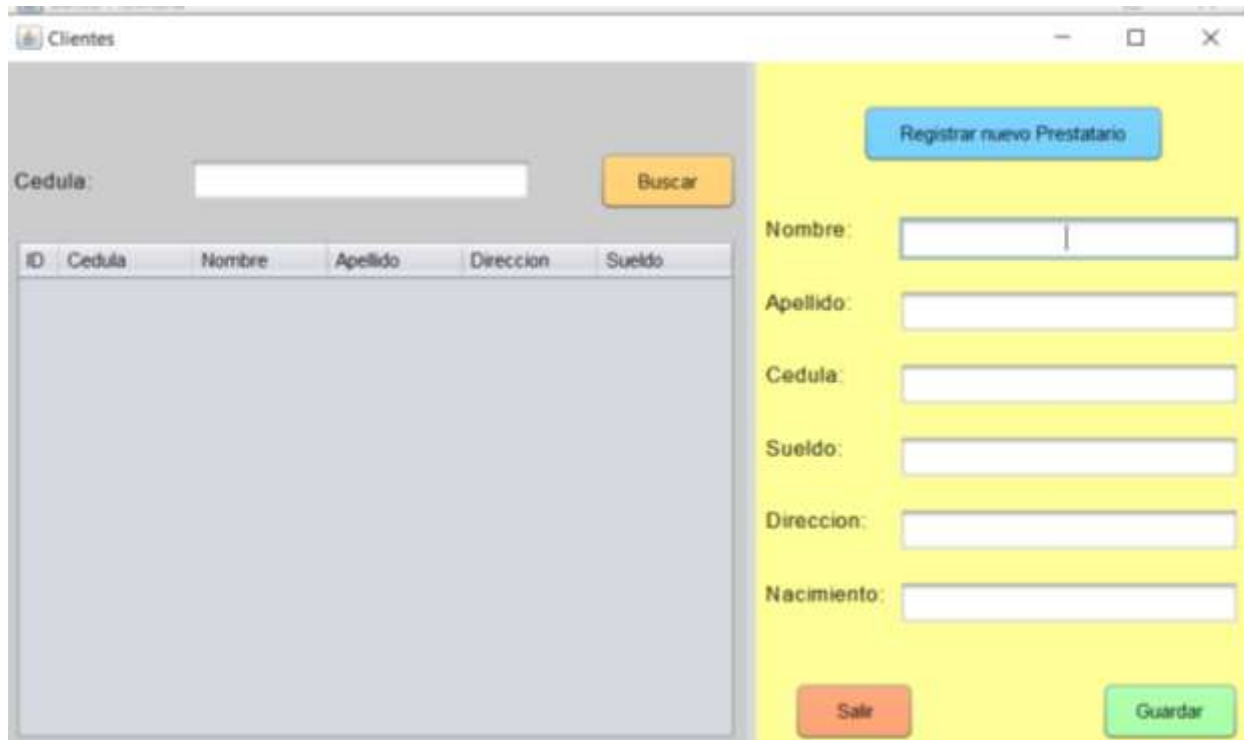
- Ventana para registrar un nuevo Usuario:

- Ventana Principal

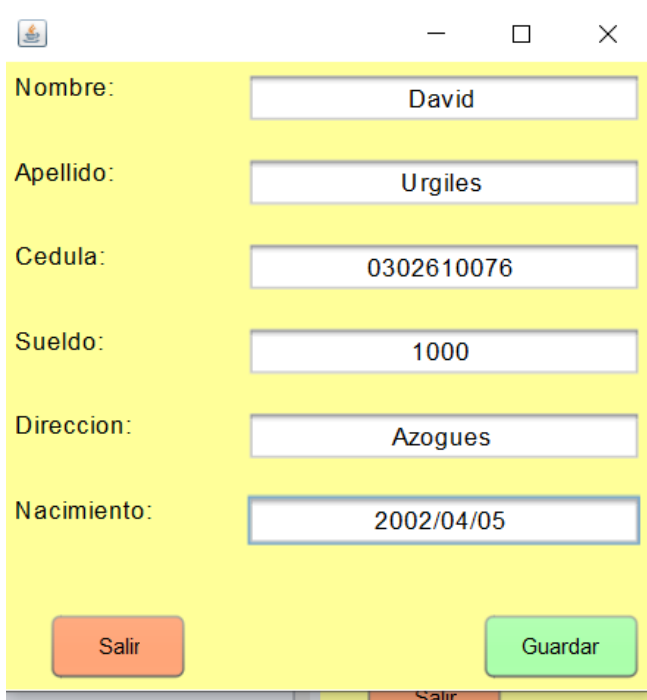



	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

- Ventana Prestatarios:



- Ventana Nuevo Prestatario




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

- Ventana Garantes:



- Ventana Nuevo Garante



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

- Ventana Nuevo Prestamo:

stamo

**solicitante:**


**Datos de la Vivienda:**  
Direccion:   
Valor:   
**Datos del Prestamo:**  
Valor Solicitado:   
Fecha de Inicio:   
Fecha de Fin:   
Interes:

**garante:**

- Ventana Prestamos

Cedula:

Prestatario	Garante	Monto	FechaInicio	FechaFin
David Urgiles	Maria Quintuña	85000	2021/01/31	2022/01/31

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

- Link Git:

<https://github.com/VazquezAriel/PruebaPractica03-ProgramacionAplicada/tree/master>

#### RESULTADO(S) OBTENIDO(S):

- Conocimos el funcionamiento de JPA y su implementación en JAVA.

#### CONCLUSIONES:

- Pudimos reforzar el contenido aprendido en clase por medio de la práctica realizada

#### RECOMENDACIONES:

- Revisar la información proporcionada por el docente previo a la práctica
- Haber asistido a las sesiones de clase.
- Consultar con el docente las dudas que puedan surgir al momento de realizar la práctica

**Nombre de estudiante:** Ariel Vazquez

**Firma de estudiante:**

