

REQUERIMIENTO GESTIÓN DE ANTEPROYECTOS CORBA

MANUAL DE INSTALACIÓN



INTEGRANTES

Albeiro Silva Muñoz

Santiago Ramirez Chavez

PROFESOR

Ing. Daniel Eduardo Paz Perafán

UNIVERSIDAD DEL CAUCA

FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

PROGRAMA INGENIERÍA DE SISTEMAS

CURSO: SISTEMAS DISTRIBUIDOS

POPAYÁN, AGOSTO 2019

Tabla de contenido

1. INTRODUCCIÓN.....	3
2. REQUERIMIENTOS DEL SISTEMA	4
3. DESCRIPCIÓN DE LA DOCUMENTACIÓN ENTREGADA.....	5
4. DESCRIPCIÓN DE LA ESTRUCTURA DE DIRECTORIOS, DESCRIPCIÓN DE LOS ARCHIVOS DE SOPORTE.	6
4.1. DIRECTORIO SRC.....	6
5. DESCRIPCIÓN DE LOS ARCHIVOS FUENTE (INTERFACES, ARCHIVOS FUENTE CLIENTE, ARCHIVOS FUENTE SERVIDOR).....	10
5.1. INTERFACES.....	10
5.2. ARCHIVOS FUENTE CLIENTE	12
5.3. ARCHIVOS FUENTE SERVIDOR.....	13
6. DESCRIPCIÓN DEL PROCESO DE COMPILACIÓN.....	14
7. DESCRIPCIÓN DEL PROCESO DE EJECUCIÓN.....	15
7.1. EJECUCIÓN DEL SERVIDOR	16
7.2. EJECUCIÓN DEL CLIENTE	17
8. DESCRIPCIÓN DEL MANEJO DEL INSTALADOR	18

1. INTRODUCCIÓN

Este documento está dirigido a los usuarios que deseen instalar el programa para gestionar anteproyectos. A lo largo del documento se describirán los aspectos necesarios para el correcto funcionamiento del software. En la aplicación se podrá encontrar el código fuente listo para compilar y ejecutar. Ya que está diseñado para funcionar como una aplicación cliente/servidor podrá ejecutarlo en una misma computadora o en dos máquinas distintas.

2. REQUERIMIENTOS DEL SISTEMA

Sistema operativo

Para poder ejecutar la carpeta comprimida con códigos fuente que se le ha proporcionado se debe tener en su equipo instalado los siguientes elementos:

Para descargar e instalar java lo puedes hacer desde la página oficial de Oracle. Al instalar java se instala:

- **JRE** (Entorno de ejecución de Java): El JRE es la implementación de la Máquina virtual de Java que realmente ejecuta los programas de Java
- **JDK** (Java Development Kit): El JDK es un kit de desarrollo de software que se utiliza para escribir aplicaciones con el lenguaje de programación Java.

El desarrollo y ejecución de aplicaciones en Java exige que las herramientas para compilar (javac.exe) y ejecutar (java.exe).

- **Javac:** Es una herramienta para leer clases y definiciones de interface escritas en java, y compilarlas al lenguaje de la máquina virtual Java (archivos bytecode).
- **Java:** Es una herramienta para ejecutar los programas en lenguaje de máquina virtual java.

No existen requerimientos en cuanto al sistema operativo que se esté utilizando, aprovechando las ventajas que ofrece Java en cuanto a versatilidad y portabilidad de los códigos fuente.

IDE de desarrollo

- IDLE Java [Preferiblemente NetBeans versión 8.0 o superior]

Servidor local

- Servidor local Xampp para que nos permita instalar de forma sencilla Apache y además el servidor de bases de datos MySql.

Librerías

- Mysql-connector-java-5.1,46.jar: Utilizada para la conexión a la base de datos MySql.
- Jcalendar-1.4.jar: Utilizada para manejar las fechas al momento de realizar un determinado registro.

Estas dos librerías se adjuntan en el proyecto

Tecnología utilizada

- La tecnología utilizada fue java CORBA
- Motor de bases de datos MySql

Lenguaje de programación

El lenguaje de programación utilizado fue Java.

Cumpliendo con las características descritas anteriormente se asegura que la aplicación se ejecutará con total normalidad, seguridad, eficacia deseada.

3. DESCRIPCIÓN DE LA DOCUMENTACIÓN ENTREGADA

Este software se entrega junto con la siguiente documentación:

Manual de instalación (Actual): Documento que tiene como objetivo orientar a los usuarios en la instalación del software. Describirá los requisitos hardware y software para el funcionamiento de la aplicación para la gestión de anteproyectos.

Manual de usuario: Documento que tiene como objetivo presentar una descripción detallada de los servicios habilitados para el usuario.

Manual técnico: Documento que proporciona información técnica relacionada con el desarrollo del software.

4. DESCRIPCIÓN DE LA ESTRUCTURA DE DIRECTORIOS, DESCRIPCIÓN DE LOS ARCHIVOS DE SOPORTE.

Dentro de la estructura de directorio que se envió y que fue nombrada como `lsd_corba_archivos_fuente_silvaMuñozA_RamirezS`, se encuentran los siguientes directorios, ver ilustración 1.









	build	30/07/2019 20:57	Carpeta de archivos	
	MANUALES	01/08/2019 19:14	Carpeta de archivos	
	nbproject	30/07/2019 20:57	Carpeta de archivos	
	orb.db	01/08/2019 17:34	Carpeta de archivos	
	src	01/08/2019 19:08	Carpeta de archivos	
	test	21/07/2019 16:51	Carpeta de archivos	
	build.xml	27/07/2019 12:14	Documento XML	4 KB
	manifest.mf	21/07/2019 11:00	Archivo MF	1 KB

Ilustración 1. Estructura de directorios

En los siguientes puntos únicamente se explicará el contenido de la carpeta “src”, la cual es la que contiene los códigos fuente de la aplicación, todas las demás carpetas y archivos corresponden a ficheros generados por el entorno de desarrollo, que, aunque no son de importancia para el fin de este documento, si lo son para tener una correcta visualización del proyecto en el entorno de desarrollo, dentro de la carpeta “src” se tiene la siguiente estructura de directorios:

4.1. DIRECTORIO SRC

Dentro del directorio **src** se encuentran los siguientes directorios: Ver Ilustración 2.








	cliente	30/07/2019 20:57	Carpeta de archivos	
	orb.db	30/07/2019 20:57	Carpeta de archivos	
	servidor	01/08/2019 17:35	Carpeta de archivos	
	sop_corba	30/07/2019 20:58	Carpeta de archivos	
	gestionanteproyectos.sql	01/08/2019 18:49	Archivo SQL	16 KB
	interfaz.idl	30/07/2019 20:26	Interface Definitio...	2 KB
	lanzarNS.bat	21/07/2019 16:43	Archivo por lotes ...	1 KB

Ilustración 2. Directorios del directorio src

SRC/SERVIDOR

Dentro de **servidor** se encuentran los siguientes directorios: [Ver Ilustración 3.](#)

Cada uno de estos directorios tiene archivos necesarios para la ejecución del programa






 BD	30/07/2019 20:58	Carpeta de archivos	
 DAO	30/07/2019 20:58	Carpeta de archivos	
 DTO	01/08/2019 17:37	Carpeta de archivos	
 GestionAnteproyectosImpl.java	01/08/2019 11:41	Archivo JAVA	8 KB
 ServidorDeObjetos.java	21/07/2019 16:43	Archivo JAVA	2 KB

Ilustración 3. Directorios del directorio servidor

Src/servidor/BD

Dentro del directorio accesoDatos se encuentran los siguientes archivos: [Ver Ilustración 4.](#)

Estas clases se utilizan para gestionar el acceso a la base de datos.

 clsConexion.java	23/07/2019 22:50	Archivo JAVA	2 KB
 clsConsultas.java	01/08/2019 19:36	Archivo JAVA	11 KB

Ilustración 4. Archivos del directorio BD

Src/servidor/DAO

Dentro del directorio DAO se encuentran los siguientes archivos. [Ver Ilustración 5.](#)

Estos archivos se utilizan para desarrollar el CRUD a la base de datos









 ClsAnteproyectoDAOImpl.java	29/07/2019 22:35	Archivo JAVA	11 KB
 ClsEvaluadorDAOImpl.java	30/07/2019 11:45	Archivo JAVA	11 KB
 ClsInicioSesionDAOImpl.java	01/08/2019 1:05	Archivo JAVA	3 KB
 ClsUsuarioDAOImpl.java	01/08/2019 19:37	Archivo JAVA	6 KB
 IntAnteproyectoDAO.java	25/07/2019 19:07	Archivo JAVA	1 KB
 IntEvaluadorDAO.java	28/07/2019 10:56	Archivo JAVA	1 KB
 IntInicioDeSesionDAO.java	25/07/2019 22:24	Archivo JAVA	1 KB
 IntUsuarioDAO.java	25/07/2019 22:05	Archivo JAVA	1 KB

Ilustración 5. Archivos del directorio DAO

Src/servidor/DAO

Dentro del directorio DTO se encuentran los siguientes archivos: [Ver Ilustración 6.](#)

En este directorio se encuentran los archivos necesarios para intercambiar información cliente-servidor. Permite tener objetos por medio de los cuales se transporta datos entre procesos.





 anteproyectoDTO.java	01/08/2019 17:36	Archivo JAVA	4 KB
 evaluadoresDTO.java	01/08/2019 17:36	Archivo JAVA	3 KB
 usuarioDTO.java	01/08/2019 17:36	Archivo JAVA	2 KB
 usuarioIngresadoDTO.java	01/08/2019 17:37	Archivo JAVA	2 KB

Ilustración 6. Archivos del directorio DTO

Src/servidor/

Dentro del directorio servidor se encuentran los siguientes archivos: [Ver Ilustración 7.](#)

En este directorio se encuentra la clase que implementa la interfaz que tiene definido los métodos remotos, también se encuentra el servidor de objetos, el cual se utiliza para recibir las peticiones.

 GestionAnteproyectosImpl.java	01/08/2019 11:41	Archivo JAVA	8 KB
 ServidorDeObjetos.java	21/07/2019 16:43	Archivo JAVA	2 KB

Ilustración 7. Archivos del directorio sop_rmi

SRC/CLIENTE

Dentro del directorio cliente se encuentran los siguientes directorios: [Ver Ilustración 8.](#)

Dentro de cada directorio se encuentran archivos específicos para la buena ejecución del programa.




 img	30/07/2019 20:57	Carpeta de archivos	
 vistas	30/07/2019 21:04	Carpeta de archivos	
 ClienteDeObjetos.java	01/08/2019 17:37	Archivo JAVA	3 KB

Ilustración 8. Directorios del directorio cliente

Src/cliente/img

Dentro del directorio img se encuentran las imágenes utilizadas para el enriquecimiento visual de la interfaz gráfica. Ver Ilustración 9.

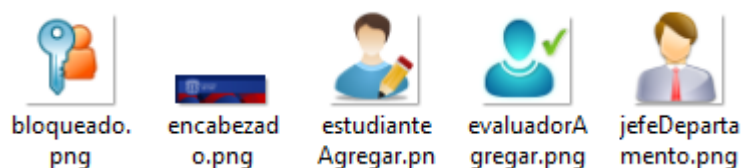


Ilustración 9. Imágenes utilizadas para la interfaz gráfica

Src/cliente/vistas

Dentro del directorio vistas se encuentran los archivos necesarios para generar la interfaz con la que el cliente va a interactuar, el menú del JefeDeDepartamento, menú del Evaluador, menú del EstudianteDirector...etc. Ver Ilustración 10.

	ClientImpl.java	31/07/2019 11:17	Archivo JAVA	1 KB
	ClsAsignarEvaluadoresJefeDD.form	01/08/2019 17:42	Archivo FORM	18 KB
	ClsAsignarEvaluadoresJefeDD.java	01/08/2019 17:42	Archivo JAVA	19 KB
	ClsBuscarAnteproyecto.form	29/07/2019 17:43	Archivo FORM	11 KB
	ClsBuscarAnteproyecto.java	29/07/2019 17:43	Archivo JAVA	14 KB
	ClsCambiarConceptoJefeDD.form	29/07/2019 22:36	Archivo FORM	8 KB
	ClsCambiarConceptoJefeDD.java	29/07/2019 22:36	Archivo JAVA	8 KB
	ClsIngresarConceptoEvaluador.form	29/07/2019 23:06	Archivo FORM	11 KB
	ClsIngresarConceptoEvaluador.java	29/07/2019 23:06	Archivo JAVA	11 KB
	ClsListarAnteproyectos.form	24/07/2019 23:30	Archivo FORM	9 KB
	ClsListarAnteproyectos.java	24/07/2019 23:33	Archivo JAVA	9 KB
	ClsLogin.form	01/08/2019 17:34	Archivo FORM	12 KB
	ClsLogin.java	01/08/2019 17:34	Archivo JAVA	16 KB
	ClsMenuDirector.form	28/07/2019 11:22	Archivo FORM	12 KB
	ClsMenuDirector.java	28/07/2019 11:22	Archivo JAVA	12 KB
	ClsMenuEstudiante.form	28/07/2019 11:17	Archivo FORM	10 KB
	ClsMenuEstudiante.java	28/07/2019 11:17	Archivo JAVA	10 KB
	ClsMenuEvaluador.form	01/08/2019 11:03	Archivo FORM	12 KB
	ClsMenuEvaluador.java	01/08/2019 11:03	Archivo JAVA	15 KB
	clsMenuJefeDeDepartamento.form	25/07/2019 19:37	Archivo FORM	15 KB
	clsMenuJefeDeDepartamento.java	25/07/2019 19:37	Archivo JAVA	17 KB
	ClsRegistrarAntepJefeDeDepartament...	29/07/2019 22:58	Archivo FORM	16 KB
	ClsRegistrarAntepJefeDeDepartament...	29/07/2019 22:58	Archivo JAVA	17 KB
	clsRegistrarUsuariosJefeDeDepartame...	29/07/2019 12:00	Archivo FORM	13 KB
	clsRegistrarUsuariosJefeDeDepartame...	29/07/2019 12:00	Archivo JAVA	13 KB

Ilustración 10. Archivos del directorio vistas.

Además dentro de este directorio se encuentra la clase ClientImpl la cual implementa los métodos de la interfaz ClientInt utilizados para realizar el proceso callback.

[Src/ sop_corba](#)

Dentro de este directorio se encuentran archivos importantes para la ejecución del programa como, las interfaces InterfaceOpretations las cuales tienen definidos los métodos remotos, también se encuentran los stub y los POA

5. DESCRIPCIÓN DE LOS ARCHIVOS FUENTE (INTERFACES, ARCHIVOS FUENTE CLIENTE, ARCHIVOS FUENTE SERVIDOR)

5.1. INTERFACES

La interfaz principal es la interfaz.idl la cual se ejecuta con el comando para generar los archivos base para la ejecución del programa.

Dentro de la interfaz idl se crearon 2 interfaces, GestionAnteproyectos y ClientInt.

En el patrón DAO se crearon 4 interfaces, cada una con la responsabilidad asociada a su nombre

En el patrón **DAO** se utilizaron las siguientes interfaces: Ver Ilustración 11

En cada interfaz están declarados los métodos relacionados a cada nombre de interfaz.





 IntAnteproyectoDAO.java	25/07/2019 19:07	Archivo JAVA	1 KB
 IntEvaluadorDAO.java	28/07/2019 10:56	Archivo JAVA	1 KB
 IntInicioDeSesionDAO.java	25/07/2019 22:24	Archivo JAVA	1 KB
 IntUsuarioDAO.java	25/07/2019 22:05	Archivo JAVA	1 KB

Ilustración 11. Interfaces utilizadas para el patrón DAO

La interfaz **intAnteproyectoDAO** tiene los siguientes métodos: Ver Ilustración 12. Todos los métodos están relacionados. Se utiliza esta interfaz para gestionar todo lo que tiene que ver con un anteproyecto.

```

public interface IntAnteproyectoDAO {
    public boolean registrarAnteproyecto(anteproyectoDTO objAnteproyecto) throws
    public anteproyectoDTO[] listarAnteproyectos()throws SQLException ;
    public anteproyectoDTO buscarAnteproyecto(int codigo)throws SQLException ;
    public long numeroFilas()throws SQLException;
    public boolean cambiarConceptoAnteproyecto(int codigo) throws SQLException;
}

```

Ilustración 12. Métodos de la interfaz intAnteproyectoDAO

La interfaz **intEvaluadorDAO** tiene los siguientes métodos. Ver Ilustración 13. Es ta interfaz se utiliza para gestionar todo lo que tiene que ver con un evaluador.

```

public interface IntEvaluadorDAO {
    public boolean asignarEvaluador(int codigo, evaluadoresDTO objEvaluador)throws
    public boolean registrarEvaluador(evaluadoresDTO objEvaluador) ;
    public evaluadoresDTO buscarEvaluador(int codigo) throws SQLException;
    public evaluadoresDTO[] listarEvaluadores()throws SQLException;
    public boolean aniadirConceptoEvaluador(int codigo, int concepto)throws SQLEx
}

```

Ilustración 13. Métodos de la interfaz intEvaluadorDAO

La interfaz **intInicioSesionDAO** solo tiene un método, el método que se utiliza para verificar los credenciales de un usuario. Ver Ilustración 14.

```

public interface IntInicioDeSesionDAO {
    public usuarioDTO verificarCredenciales(String usuario, String contrasenia)tr
}

```

Ilustración 14. Método de la interfaz intInicioSesion

La interfaz **intUsuarioDAO** se utiliza para gestionar todo lo que tiene que ver con un usuario, tiene los siguientes métodos. Ver Ilustración 15

```

public interface IntUsuarioDAO {
    public boolean registrarUsuario(usuarioDTO objUsuario)throws SQLException;
    public boolean guardarCredencialesUsuario(String usuario, String contrasenia
    public usuarioDTO[] listarUsuarios()throws SQLException ;
    public usuarioIngresadoDTO[] listarUsuariosIngresados();
    public usuarioIngresadoDTO recuperarUsuario()throws SQLException;
}

```

Ilustración 15. Métodos de la interfaz intUsuarioDAO

En el directorio **sop_corba** se utilizaron las siguientes interfaces:

GestionAnteproyectosOperations

La interfaz **GestionAnteproyectosOperations** la cual tiene definidos los métodos remotos que serán implementados por la clase **GestionAnteproyectosImpl** y a su vez serán invocados por el cliente mediante el objeto remoto. Ver Ilustración 16

```
public interface GestionAnteproyectosOperations
{
    sop_corba.GestionAnteproyectosPackage.usuarioDTO verificarCredenciales (String
    boolean registrarAnteproyecto (sop_corba.GestionAnteproyectosPackage.anteproyecto
    boolean registrarUsuario (sop_corba.GestionAnteproyectosPackage.usuarioDTO obj
    sop_corba.GestionAnteproyectosPackage.anteproyectoDTO[] listarAnteproyectos ();
    sop_corba.GestionAnteproyectosPackage.anteproyectoDTO buscarAnteproyecto (int c
    boolean asignarEvaluador (int codigo, sop_corba.GestionAnteproyectosPackage.eva
    int numeroFilas ();
    sop_corba.GestionAnteproyectosPackage.evaluadoresDTO buscarEvaluador (int codig
    boolean cambiarConceptoAnteproyecto (int codigo);
    boolean guardarCredencialesUsuario (String usuario, String contrasenia);
    boolean anadirConceptoEvaluador (int codigo, String concepto);
    sop_corba.GestionAnteproyectosPackage.usuarioIngresadoDTO recuperarUsuario ();
    sop_corba.GestionAnteproyectosPackage.usuarioDTO[] listarUsuarios ();
    sop_corba.GestionAnteproyectosPackage.evaluadoresDTO[] listarEvaluadores ();
    sop_corba.GestionAnteproyectosPackage.usuarioIngresadoDTO[] listarUsuariosIngre
    boolean registrarCliente (sop_corba.ClienteInt objcllback);
} // interface GestionAnteproyectosOperations
```

Ilustración 16. Métodos de la interfaz GestionAnteproyectosOperations

ClienteIntOperations

La interfaz **ClienteIntOperations** tiene definido el método que será utilizado para realizar el proceso callback Ver Ilustración 17.

```
public interface ClienteIntOperations
{
    void recibirMensaje (String mensaje);
} // interface ClienteIntOperations
```

Ilustración 17. Métodos de la interfaz ClienteIntOperations

5.2. ARCHIVOS FUENTE CLIENTE

CLIENTE DE OBJETOS

Hace referencia al cliente de objetos, el cual tiene la responsabilidad de establecer la referencia remota hacia el servidor y servir como puente entre las interfaces gráficas y el servidor remoto, invocando los procedimientos remotos y entregando los valores retornados de vuelta a la interfaz gráfica para ser procesados visualmente.

VISTAS

Los archivos llamados <nombre>.form Son los archivos que contienen la especificación de los formularios gráficos con los cuales interactúa el usuario y la aplicación, existe un archivo de este tipo por cada interfaz gráfica definida para cada tipo de usuario (una interfaz por cada funcionalidad del usuario).

Los archivos llamados <nombre>.java son archivos de contienen la lógica determinada para cada formulario, la lógica que despliegan los botones, las validaciones de los campos y el punto de entrada mediante el cual se inicializan los formularios (existe un archivo de este tipo por cada .form)

5.3. ARCHIVOS FUENTE SERVIDOR

UTILIDADES

El archivo llamado utilidadesRegistroS.java hacen referencia al archivo que contiene la lógica para efectuar el registro de servidores en el NS, esta clase es genérica y es usada por los servidores de objetos los cuales la usan de acuerdo a sus necesidades.

SERVIDOR DE OBJETOS

El archivo llamado ServidorDeObjetos se encarga de registrar la referencia remota en el NS para que esta pueda ser instanciada por los clientes, si se tiene más de un servidor esta clase será la encargada de registrarlos todos asignando la ip el puerto y el nombre característico para cada uno de ellos.

DAO

En este directorio se encuentran las interfaces y las clases que implementan las interfaces que componen el patrón DAO, en estos archivos se realiza la implementación de los métodos de acceso a los datos, para las necesidades de cada usuario, se tiene una interfaz para cada grupo de acciones relacionadas.

DTO

En este directorio se encuentran las clases que sirven para tener objetos por medio de los cuales se transporta datos entre procesos.

6. DESCRICIÓN DEL PROCESO DE COMPILACIÓN

1. Para realizar el proceso de compilación debe descargar el archivo `Isd_corba_archivos_fuente_silvaMuñozA_RamirezS.zip` y descomprimirlo.
2. Dentro de la carpeta `src` encontrará un archivo llamado `gestionanteproyectos.sql` el cual se utiliza para crear la base de datos.
3. Para la creación de la base de datos siga los siguientes pasos:
 - 3.1. Abra el servidor local xampp e inicie el Apache y MySQL como se muestra en la Ilustración 18.

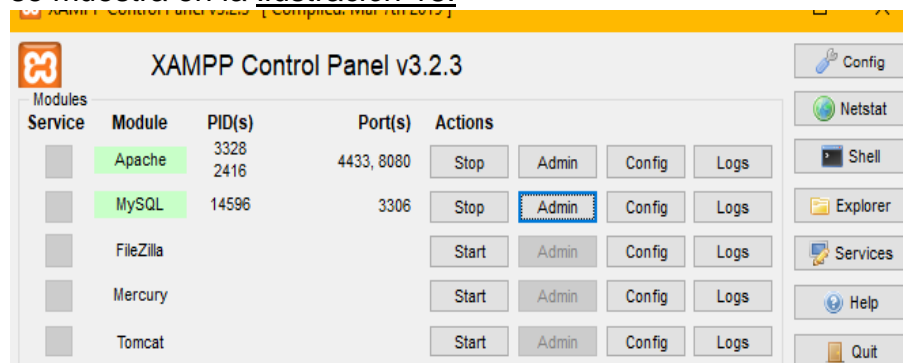


Ilustración 18. Ejecutar Apache y MySQL.

- 3.2. Acceda a phpMyAdmin y diríjase a la parte de importar, en esta ventana aparecerá la opción para subir el archivo.sql y presionar continuar para realizar la importación como se muestra en la Ilustración 19.



Trabajando en la tabla "usuarios"

Importar:

puede ser comprimido (gzip, zip) o descomprimido.
El archivo comprimido tiene que terminar en `.[formato].[compresión]`. Por ejemplo: `.sql.zip`
En su ordenador: Ningún archivo seleccionado (Máximo: 200MB)
Puede arrastrar un archivo en cualquier página.
Codificación de caracteres del archivo:

Ilustración 19. Importar archivo.sql

- 3.3. El nombre de la base de datos y el puerto que se está utilizando para acceder a MySQL deben configurarse en la clase clsConexion que se encuentra en el paquete servidor/BD. Como se muestra en la Ilustración 20.

```
public class clsConexion {  
  
    private static Connection conn;  
    private static final String driver = "com.mysql.jdbc.Driver";  
    private static final String user = "root";  
    private static final String password = "";  
    private static final String url = "jdbc:mysql://localhost:3306/gestionantepr  
    private String error = "";
```

Ilustración 20. Configuración de la base de datos.

- 3.4. Ahora procedemos a importar el proyecto en el IDE netbeans. Se realiza yendo al botón Open Project que se encuentra en la parte superior izquierda como se muestra en la Ilustración 21.

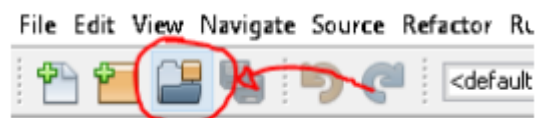


Ilustración 21. Importar el proyecto en el IDE

- 3.5. Después de haber importado el proyecto en el IDE procedemos a compilar. Para compilar estas clases entonces se debe hacer click en el botón de clean build ubicado en la parte superior central: como se muestra en la Ilustración 22.



Ilustración 22. Proceso de compilación.

Si todo salió bien procedemos al proceso de ejecución

7. DESCRIPCIÓN DEL PROCESO DE EJECUCIÓN

Antes de ejecutar el servidor, tenemos que lanzar el n_s el cual se hace utilizando el archivo lanzarNS.bat que se encuentra dentro del directorio src, hacemos doble click sobre el archivo para poder lanzarlo y se nos abrirá una consola como la que se muestra en la Ilustración 23.


```
C:\WINDOWS\system32\cmd.exe
C:\Users\USERPC\Desktop\corba\CORBA\SD_TERCER_CORTE_ENTREGAR>orbd -ORBInitialHost localhost -ORBInitialPort 2020
Picked up JAVA_TOOL_OPTIONS: -Dfile.encoding=UTF8
```

Ilustración 23. Lanzar N_S

7.1. EJECUCIÓN DEL SERVIDOR

Para la ejecución del servidor nos ubicamos en el directorio servidor y damos click derecho sobre la clase ServidorDeObjetos.java y seleccionamos la opción Run File, de esta manera el servidor de objetos se ejecutará, como se muestra en la Ilustración 24.

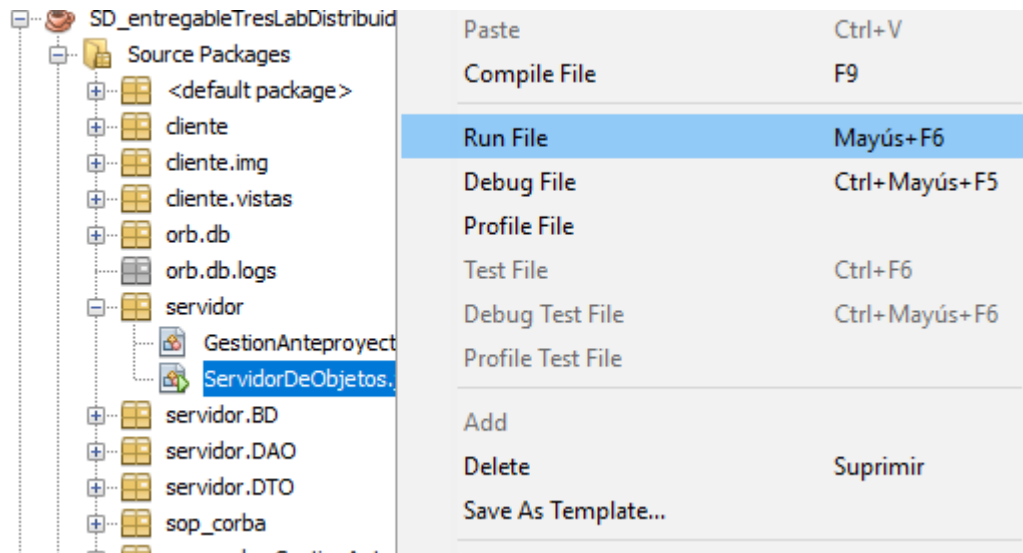


Ilustración 24. Ejecución del servidor

Si todo ha salido bien se mostrará un mensaje en la consola diciendo “servidor listo y esperando” como se muestra en la Ilustración 25.

```
El Servidor esta listo y esperando ...
|
```

Ilustración 25. Confirmación de que el servidor se ejecutó con éxito.

7.2. EJECUCIÓN DEL CLIENTE

Para la ejecución del cliente nos ubicamos en el directorio cliente, el cual contiene la clase ClienteDeObjetos.java, sobre la cual deberá dar click derecho y seleccionar la opción Run File como se muestra en la Ilustración 26.

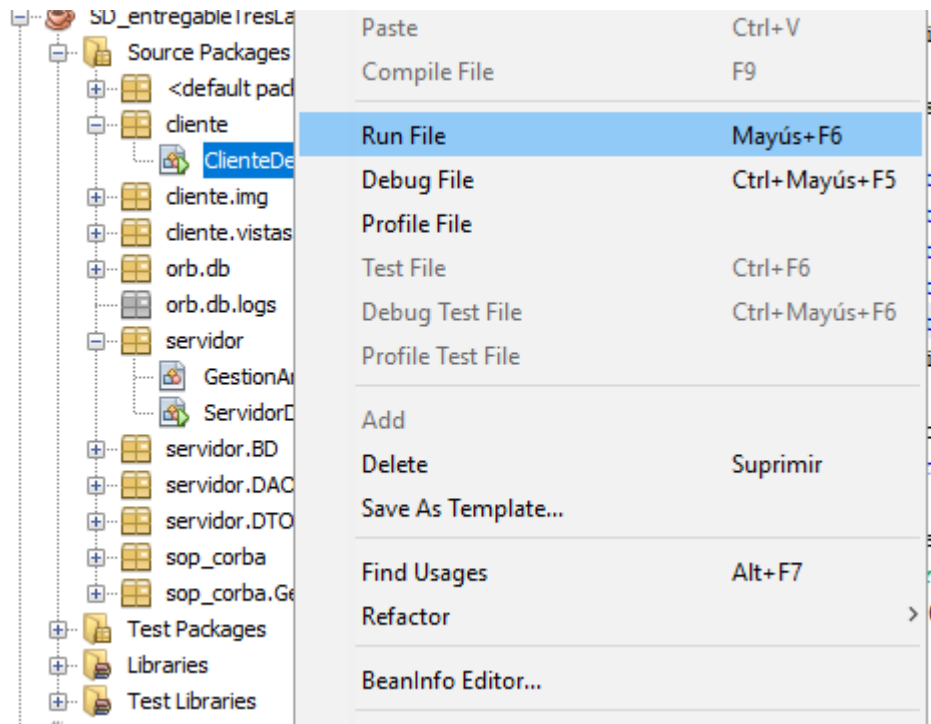


Ilustración 26. Ejecución del cliente

Si todo salió bien se le mostrará una ventana como la que se muestra en la Ilustración 27 en la cual usted podrá ingresar con un usuario y una contraseña para poder realizar la gestión de los anteproyectos.



Ilustración 27. Confirmación de que el cliente se ejecutó con éxito

8. DESCRIPCIÓN DEL MANEJO DEL INSTALADOR

En este caso, el software no posee un instalador. Viene con el código fuente y la instalación consiste en compilar el código y ejecutarlo, como se describió anteriormente. Queda a decisión del usuario donde guardar el código. En el caso de que quieran ejecutar el cliente y servidor en máquinas diferentes, cada una de estas deberá tener una copia del código para poder hacer el proceso de ejecución.