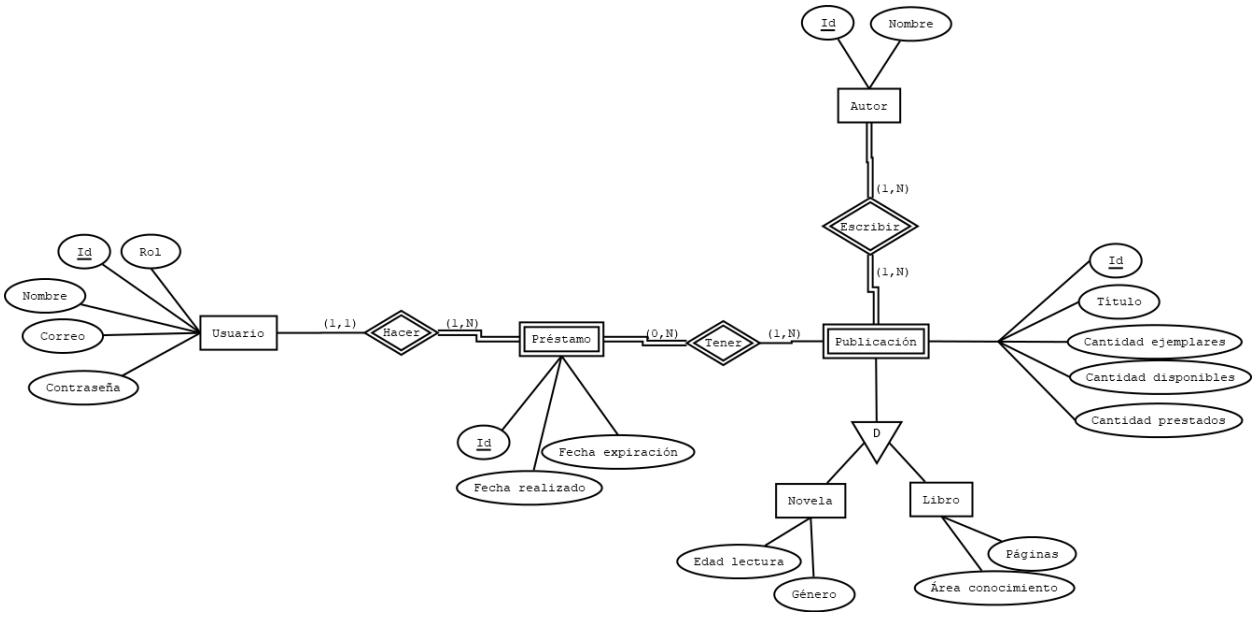
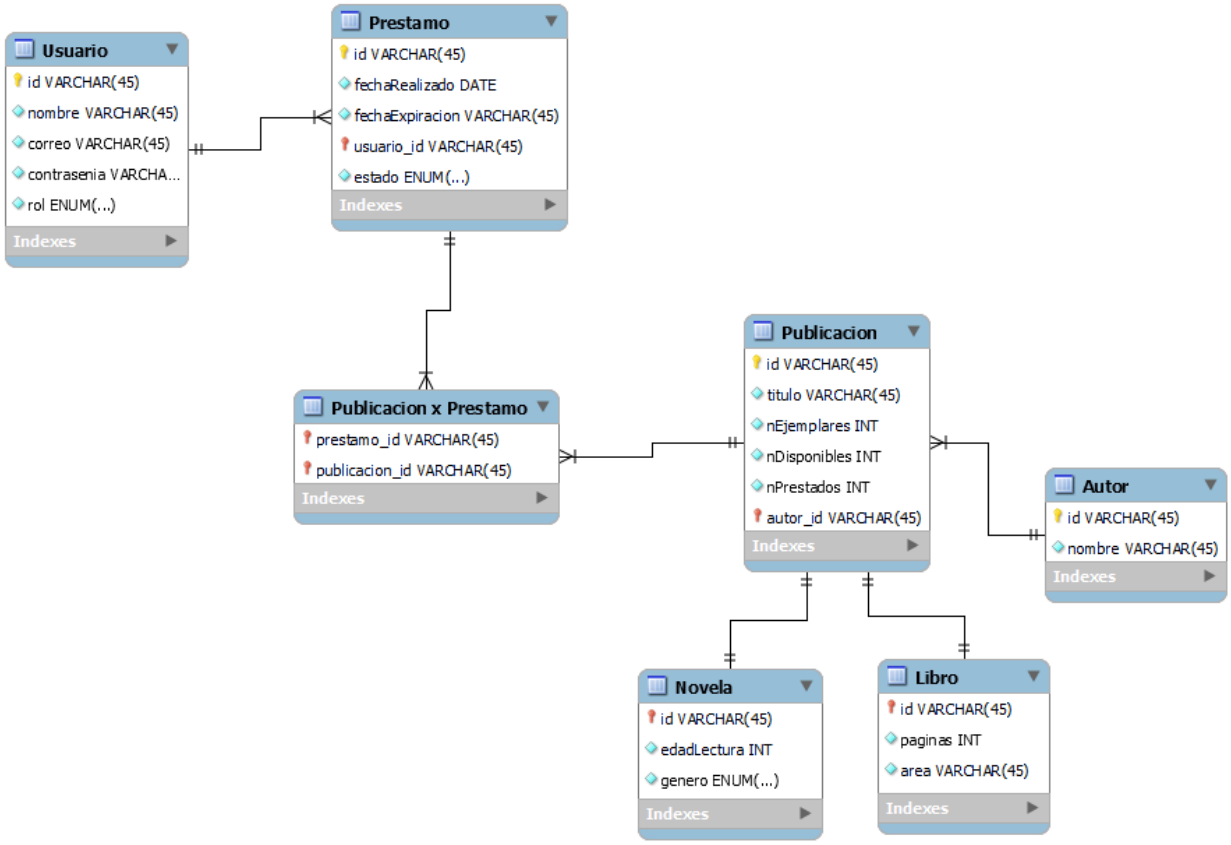
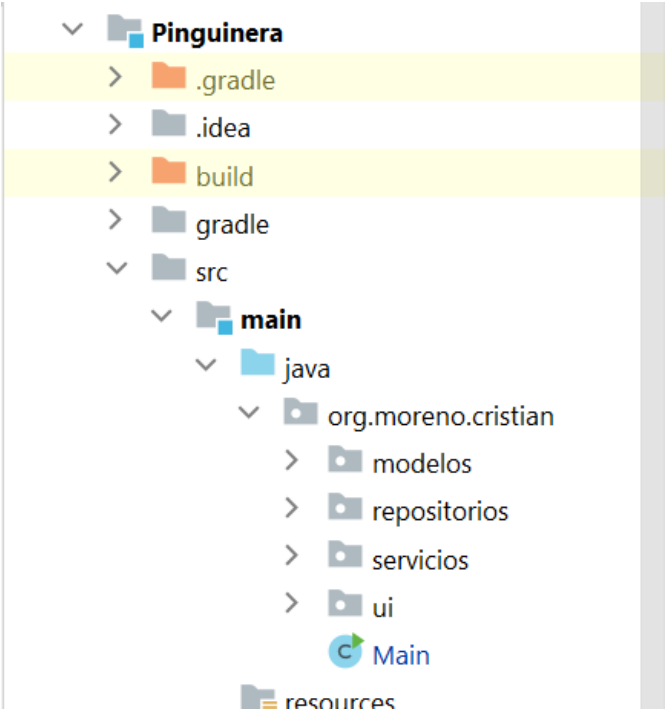
Primero que todo vamos a hacer el diseño de nuestra base de datos. Se identificaron 6 entidades que se muestran a continuación.





Para la construcción de la aplicación en Java seguimos una arquitectura por capas y separamos las clases e interfaces en los paquetes de modelos, repositorios, servicios y ui (interfaz de usuario).



Los menú utilizan los servicios para acceder a la base de datos utilizando modelos para representar las tablas y registros de la bd.

Utilizamos el patrón singleton para crear una única conexión a la bd y un único scanner

```java
public class ConexionBD {  20 usages  Cristian
    private static String URL;  2 usages
    private static String USER;  2 usages
    private static String PASSWORD;  2 usages

    private static String propertiesFilePath = "C:\\Users\\crism\\Desktop\\sofka\\Trabajos\\2024-C1-QA-JAV

    private static Connection connection;  6 usages

    private ConexionBD() {  no usages  Cristian
    }

    public static Connection obtenerConexion() throws SQLException {  18 usages  Cristian
        if (connection == null || connection.isClosed()) {
            try {

                Properties appProps = new Properties();
                appProps.load(new FileInputStream(propertiesFilePath));

                URL = System.getenv(appProps.get("db.url").toString());

                USER = System.getenv(appProps.get("db.user").toString());
```

```java
public class ScannerUtil {  Cristian
    private static Scanner scanner = new Scanner(System.in);  6 usages
    static Logger log = LogManager.getLogger(String.valueOf(ScannerUtil.class));  1 usage

    private ScannerUtil() {  no usages  Cristian
    }

    public static Scanner obtenerScanner() {  5 usages  Cristian
        return scanner;
    }

    public static void cerrarScanner() {  1 usage  Cristian
        if (scanner != null) {
            scanner.close();
        }
    }

    public static int pedirEntero() {  11 usages  Cristian

        int entero =0;

        while (true) {
            try {
                entero = scanner.nextInt();
```

La información de contraseña, usuario y url para acceder a la bd la almacenamos como variables de entorno que referenciamos desde app.properties