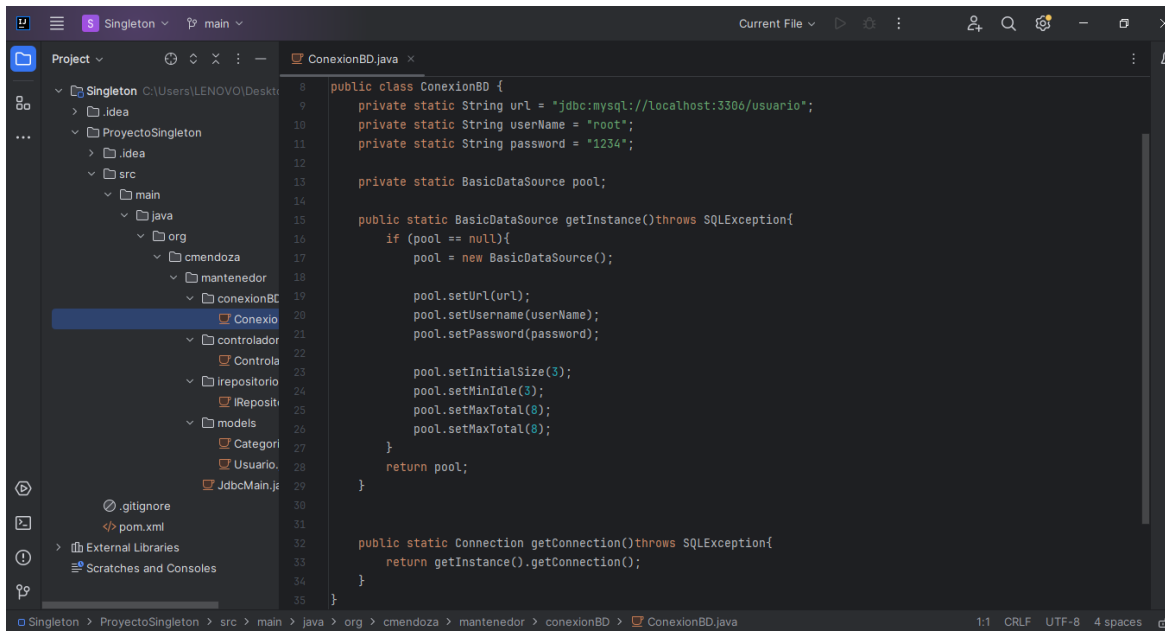


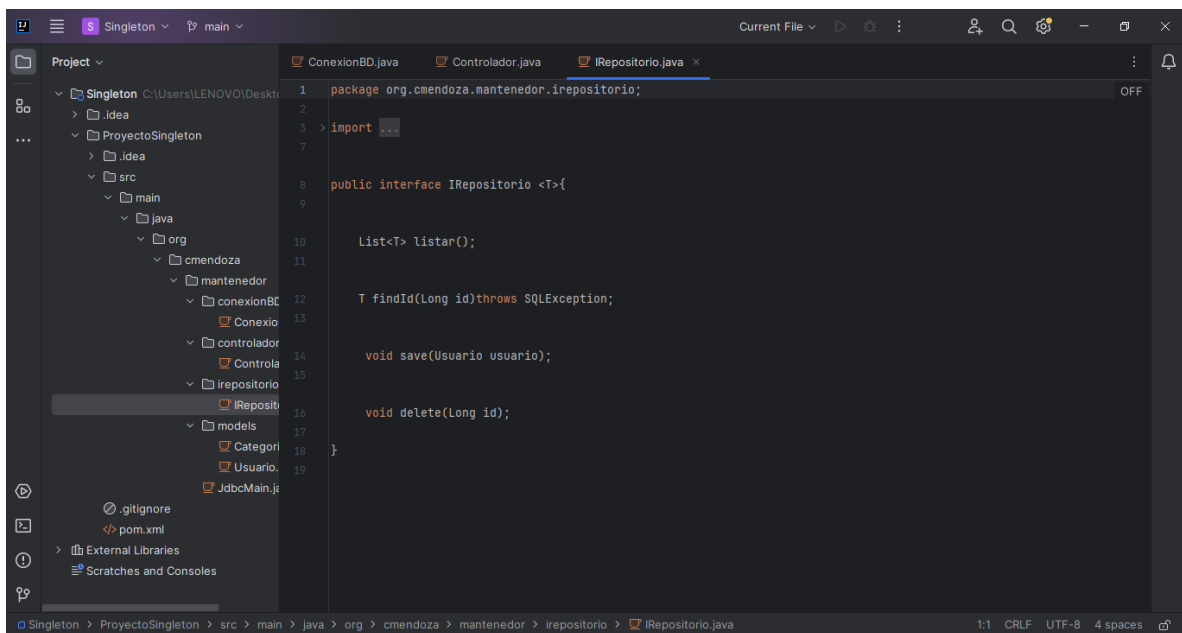
Este programa es un mantenedor de usuarios con el cual podemos agregar, eliminar, actualizar y listar los usuarios mediante un Crud

Clase singleton o conexión a BD



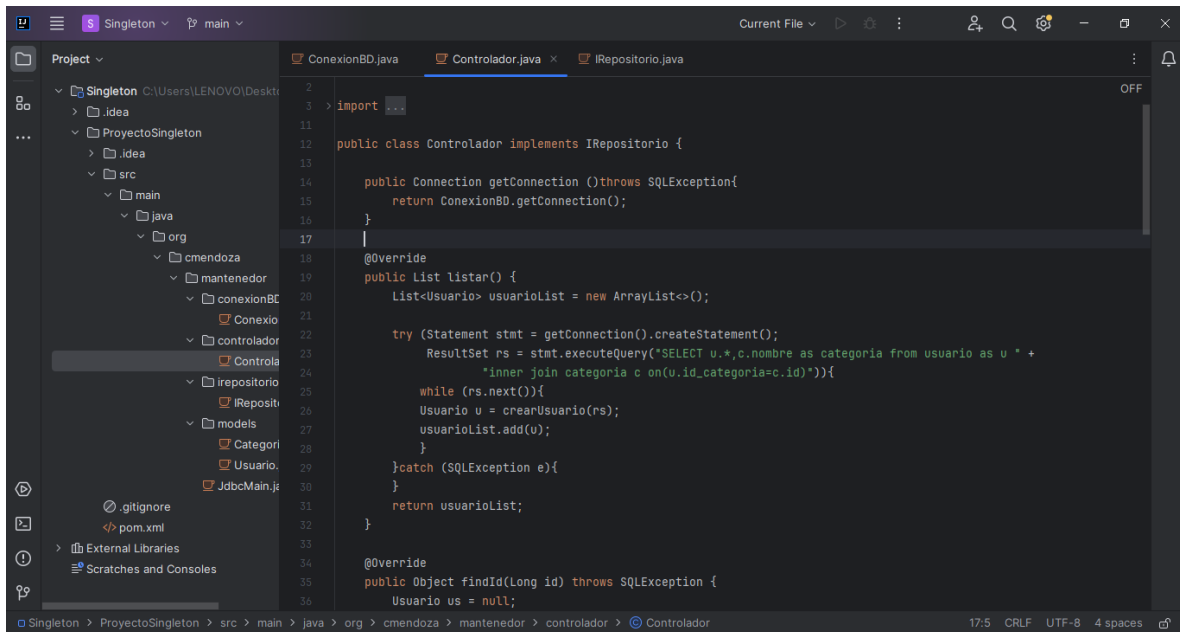
```
1 public class ConexionBD {
2     private static String url = "jdbc:mysql://localhost:3306/usuario";
3     private static String userName = "root";
4     private static String password = "1234";
5
6     private static BasicDataSource pool;
7
8     public static BasicDataSource getInstance() throws SQLException {
9         if (pool == null) {
10             pool = new BasicDataSource();
11
12             pool.setUrl(url);
13             pool.setUsername(userName);
14             pool.setPassword(password);
15
16             pool.setInitialSize(3);
17             pool.setMinIdle(3);
18             pool.setMaxTotal(8);
19             pool.setMaxTotal(8);
20         }
21         return pool;
22     }
23
24     public static Connection getConnection() throws SQLException {
25         return getInstance().getConnection();
26     }
27 }
```

Interfaz CRUD

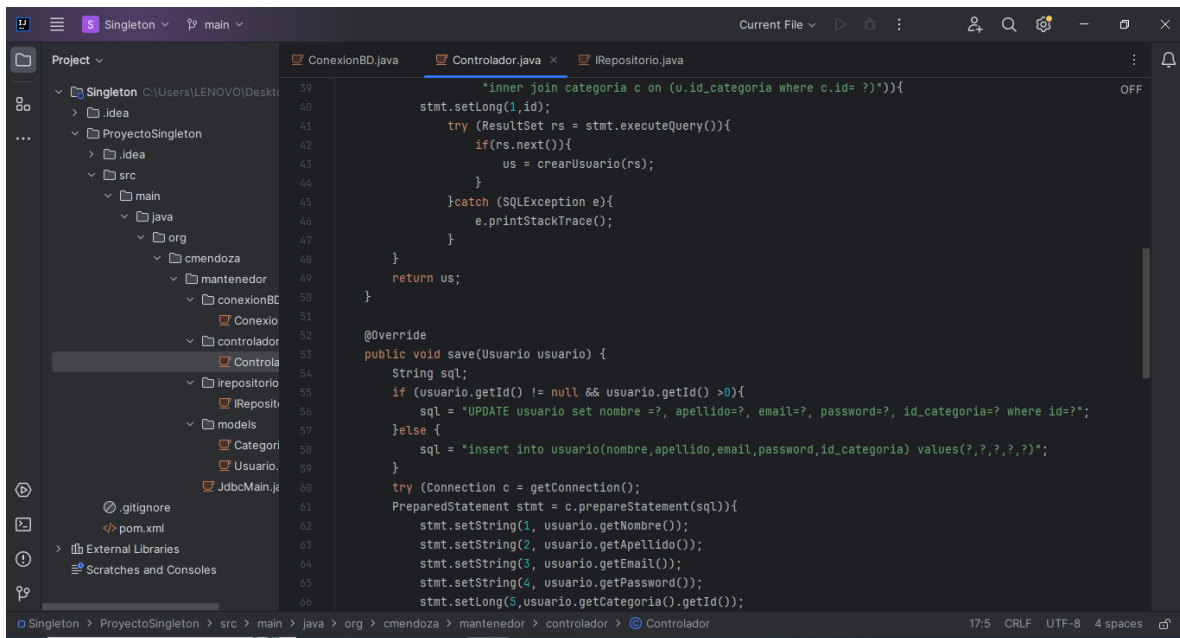


```
1 package org.cmendoza.mantenedor.irepositorio;
2
3 > import ...
4
5 public interface IRepository <T> {
6
7     List<T> listar();
8
9     T findId(Long id) throws SQLException;
10
11     void save(Usuario usuario);
12
13     void delete(Long id);
14 }
15
16
17
18
19 }
```

Controlador para realizar las consultas con la base de datos

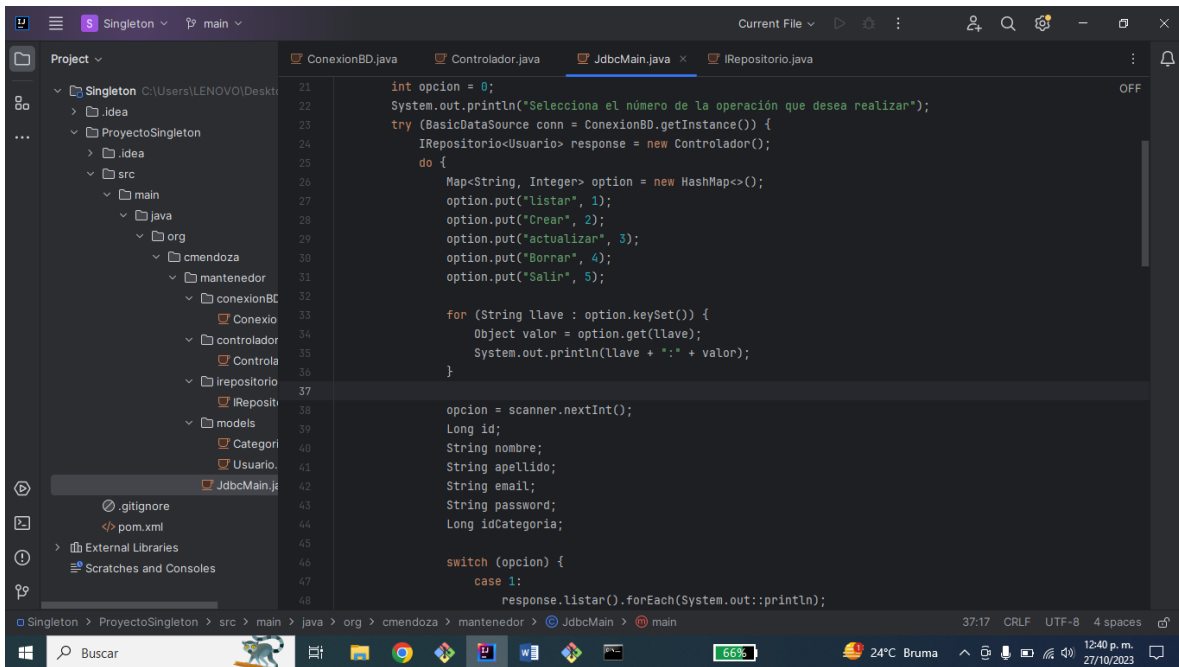


```
2
3 > import ...
11
12 public class Controlador implements IRepositoryio {
13
14     public Connection getConnection ()throws SQLException{
15         return ConexionBD.getConnection();
16     }
17
18     @Override
19     public List listar() {
20         List<Usuario> usuariolist = new ArrayList<>();
21
22         try (Statement stmt = getConnection().createStatement();
23             ResultSet rs = stmt.executeQuery("SELECT u.*,c.nombre as categoria from usuario as u " +
24                 "inner join categoria c on(u.id_categoria=c.id)")){
25             while (rs.next()){
26                 Usuario u = crearUsuario(rs);
27                 usuariolist.add(u);
28             }
29         }catch (SQLException e){
30             }
31         return usuariolist;
32     }
33
34     @Override
35     public Object findId(Long id) throws SQLException {
36         Usuario us = null;
```



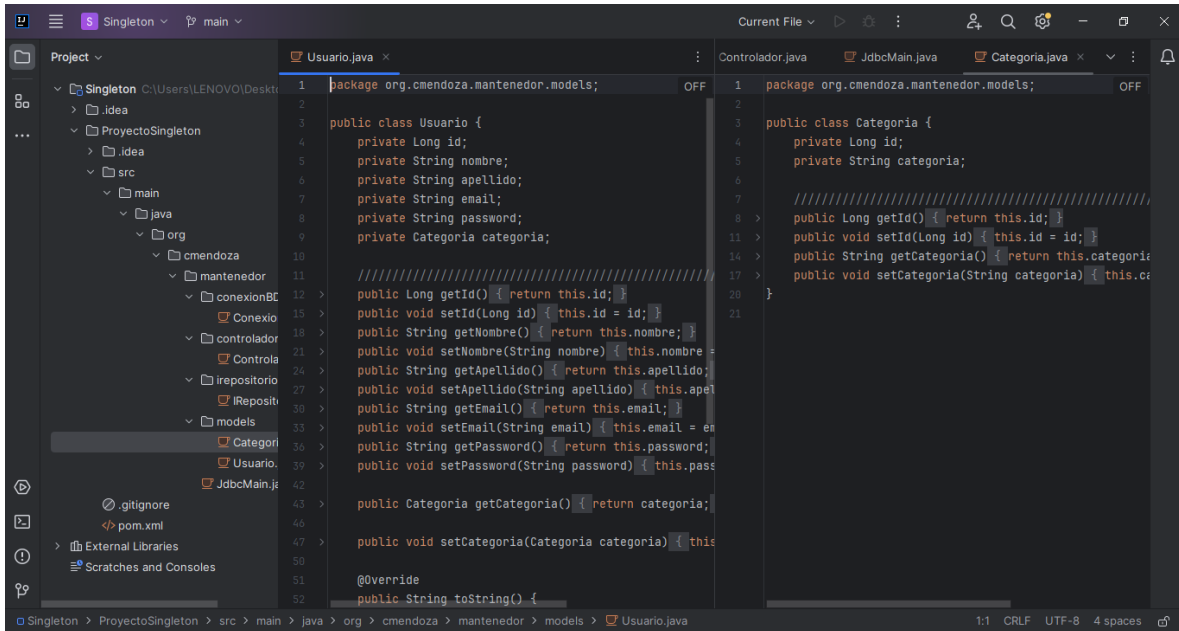
```
39         "inner join categoria c on (u.id_categoria where c.id= ?)"))){
40             stmt.setLong(1,id);
41             try (ResultSet rs = stmt.executeQuery()){
42                 if(rs.next()){
43                     us = crearUsuario(rs);
44                 }
45             }catch (SQLException e){
46                 e.printStackTrace();
47             }
48         }
49         return us;
50     }
51
52     @Override
53     public void save(Usuario usuario) {
54         String sql;
55         if (usuario.getId() != null && usuario.getId() >0){
56             sql = "UPDATE usuario set nombre=?, apellido=?, email=?, password=?, id_categoria=? where id=?";
57         }else {
58             sql = "insert into usuario(nombre,apellido,email,password,id_categoria) values(?,?,?,?,:)";
59         }
60         try (Connection c = getConnection();
61             PreparedStatement stmt = c.prepareStatement(sql)){
62             stmt.setString(1, usuario.getNombre());
63             stmt.setString(2, usuario.getApellido());
64             stmt.setString(3, usuario.getEmail());
65             stmt.setString(4, usuario.getPassword());
66             stmt.setLong(5,usuario.getCategoria().getId());
```

Clase main



```
21 int opcion = 0;
22 System.out.println("Selecciona el número de la operación que desea realizar");
23 try (BasicDataSource conn = ConexionBD.getInstance()) {
24     IRepository<Usuario> response = new Controlador();
25     do {
26         Map<String, Integer> option = new HashMap<>();
27         option.put("listar", 1);
28         option.put("Crear", 2);
29         option.put("actualizar", 3);
30         option.put("Borrar", 4);
31         option.put("Salir", 5);
32
33         for (String llave : option.keySet()) {
34             Object valor = option.get(llave);
35             System.out.println(llave + ":" + valor);
36         }
37
38         opcion = scanner.nextInt();
39         Long id;
40         String nombre;
41         String apellido;
42         String email;
43         String password;
44         Long idCategoria;
45
46         switch (opcion) {
47             case 1:
48                 response.listar().forEach(System.out::println);
```

Clase concretas



```
1 package org.cmendoza.mantenedor.models;
2
3 public class Usuario {
4     private Long id;
5     private String nombre;
6     private String apellido;
7     private String email;
8     private String password;
9     private Categoria categoria;
10
11     // getters and setters
12     public Long getId() { return this.id; }
13     public void setId(Long id) { this.id = id; }
14     public String getNombre() { return this.nombre; }
15     public void setNombre(String nombre) { this.nombre = nombre; }
16     public String getApellido() { return this.apellido; }
17     public void setApellido(String apellido) { this.apellido = apellido; }
18     public String getEmail() { return this.email; }
19     public void setEmail(String email) { this.email = email; }
20     public String getPassword() { return this.password; }
21     public void setPassword(String password) { this.password = password; }
22
23     public Categoria getCategoria() { return categoria; }
24     public void setCategoria(Categoria categoria) { this.categoria = categoria; }
25
26     @Override
27     public String toString() {
```