**METODOS DE CAJA BLANCA**

-*Método elegido: jButtonIngresarActionPerformed de la clase Login*.

private void jButtonIngresarActionPerformed(java.awt.event.ActionEvent evt) {

//GEN-FIRST:event\_jButtonIngresarActionPerformed

String usuario = jTextFieldUser.getText();

String pass = String.copyValueOf(jPassword.getPassword()); 1

//Valido que ningún campo este vacío

if(usuario != null && !usuario.isEmpty() && pass!=null && !pass.isEmpty()){ 2,3,4,5

User user = new User(usuario, pass); 6

try {

//obtengo la conexión

factory = SessionFactoryManager.getSessionFactory();

UserR userRepo = new UserR(factory);

7

//Busco el usuario en la base de datos

User retorno = (User) userRepo.getById(User.class, user.getUsuario());

if(retorno != null){ 8

//si existe comparo las contraseñas

if(user.getPassword().equals(retorno.getPassword())){ 9

//si las contraseñas son iguales le abro la ventana principal

Programa ventana = new Programa();

ventana.requestFocusInWindow(); 10

ventana.setVisible(true);

this.setVisible(false);

}else{

//si la contraseña no es igual le informo que existe un error en las credenciales

jLabelRespuestaE.setText("Usuario y/o contraseña incorrectos"); 11

}

}else{

//si el usuario no existe le informo un error de credenciales

jLabelRespuestaE.setText("Usuario y/o contraseña incorrectos"); 12

}

userRepo.getFactory().close(); 13

} catch (HibernateException e) {

System.out.println(e); 14

} finally {

factory.close(); 15

}

}else {

jLabelRespuestaE.setText("Complete ambos campos"); 16

}

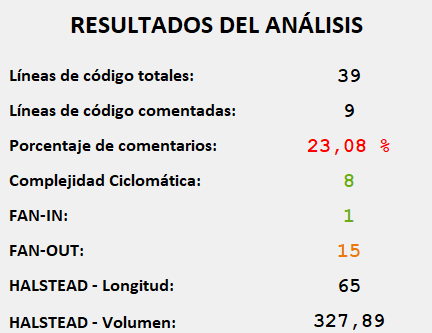
} 17

//GEN-LAST:event\_jButtonIngresarActionPerformed

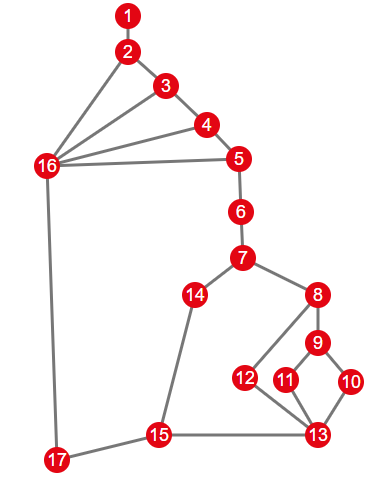
**Elección:**

Elegimos este método, ya que si bien no posee un alto FAN IN (Cantidad de métodos que lo llaman), es un método bastante importante dentro de nuestro programa, y posee una complejidad ciclomática alta en comparación a los demás métodos.

Nuestra herramienta de testing nos mostró el siguiente reporte sobre el método elegido:



**Método de McCabe**

****

**Complejidad Ciclomática:**

V(G) = Cantidad de Regiones = 8

V(G) = Nodos Predicados + 1 = 7 + 1 = 8

V(G) = Aristas – Nodos + 2 = 23-17+2 = 8

**Complejidad Ciclomática: 8**

Caminos:

1. 1 – 2 – 16 – 17
2. 1 – 2 – 3 – 16 – 17
3. 1 – 2 – 3 – 4 – 16 – 17
4. 1 – 2 – 3 – 4 – 5 – 16 – 17
5. 1 – 2 – 3 – 4 – 5 – 6 – 7 – 14 – 15 – 17
6. 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 12 – 13 – 15 – 17
7. 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 11 – 13 – 15 – 17
8. 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 13 – 15 - 17

**Casos de Prueba:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **N°** | **Usuario** | **Pass** | **Salida Esperada** | **Salida Obtenida** |
| 1 | null | null | Complete ambos campos | Complete ambos campos |
| 2 |  | null | Complete ambos campos | Complete ambos campos |
| 3 | Lautaro | null | Complete ambos campos | Complete ambos campos |
| 4 | Lautaro |  | Complete ambos campos | Complete ambos campos |
| 5 | Lautaro | 123 | Error al traer de la Base de Datos | Error al traer de la Base de Datos |
| 6 | Lautaro | 1234 | Usuario y/o contraseña incorrectos | Usuario y/o contraseña incorrectos |
| 7 | Lautaro | 12345 | Usuario y/o contraseña incorrectos | Usuario y/o contraseña incorrectos |
| 8 | Lautaro | 123456 | Ingreso al Sistema | Ingreso al Sistema |

**Métodos de Cobertura**

**Condiciones y decisiones:**

* C1: usuario != null
* C2: !usuario.isEmpty()
* C3: pass!=null
* C4: !pass.isEmpty()
* C5: User retorno = (User) userRepo.getById(User.class, user.getUsuario());
* C6: retorno != null
* C7: user.getPassword().equals(retorno.getPassword())
* D1: (usuario != null && !usuario.isEmpty() && pass!=null && !pass.isEmpty())
* D2: User retorno = (User) userRepo.getById(User.class, user.getUsuario());
* D3: (retorno != null)
* D4: (user.getPassword().equals(retorno.getPassword()))

\*Se considera la Condición 5 y la Decisión 2 por la presencia del try-catch.

-Método de condición:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Usuario** | **Pass** | **C1** | **C2** | **C3** | **C4** | **C5** | **C6** | **C7** |
| NULL | NULL | F | - | - | - | - | - | - |
|  | NULL | T | F | - | - | - | - | - |
| Lautaro | NULL | T | T | F | - | - | - | - |
| Lautaro |  | T | T | T | F | - | - | - |
| Lautaro | 123 | T | T | T | T | F |  |  |
| Lautaro | 1234 | T | T | T | T | T | F | - |
| Lautaro | 12345 | T | T | T | T | T | T | F |
| Lautaro | 123456 | T | T | T | T | T | T | T |

-Método de decisión:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Usuario** | **Pass** | **D1** | **D2** | **D3** | **D4** |
| Lautaro |  | F | - | - | - |
| Lautaro | 123 | T | F | - | - |
| Lautaro | 1234 | T | T | F | - |
| Lautaro | 12345 | T | T | T | F |
| Lautaro | 123456 | T | T | T | T |

-Método de condición/decisión:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Usuario** | **Pass** | **C1** | **C2** | **C3** | **C4** | **D1** | **C5** | **D2** | **C6** | **D3** | **C7** | **D4** |
| NULL |  | F | - | - | - | F | - | - | - | - | - | - |
|  |  | T | F | - | - | F | - | - | - | - | - | - |
| Lautaro | NULL | T | T | F | - | F | - | - | - | - | - | - |
| Lautaro |  | T | T | T | F | F | - | - | - | - | - | - |
| Lautaro | 123 | T | T | T | T | T | F | F | - | - | - | - |
| Lautaro | 1234 | T | T | T | T | T | T | T | F | F | - | - |
| Lautaro | 12345 | T | T | T | T | T | T | T | T | T | F | F |
| Lautaro | 123456 | T | T | T | T | T | T | T | T | T | T | T |