



```
1 package fr.eni.vault;
2
3 import static org.junit.jupiter.api.Assertions.assertEquals;
4 import static org.junit.jupiter.api.Assertions.assertThrows;
5 import org.junit.jupiter.api.Test;
6
7 import com.eni.vault.bll.BLLException;
8 import com.eni.vault.bll.Vault;
9
10 public class TestVaultClose {
11
12     Vault vault = new Vault();
13
14     @Test
15     public void testFermetureCode() throws BLLException {
16         //Arrange - préparer le test
17
18
19         //Act - Vérifier la méthode
20         boolean canLock = vault.lock("1234");
21         //Assert - Comparer le resultat renvoyé avec le résultat attendu
22         assertEquals(canLock, true, "Echec fonction lock");
23
24
25         //Act - Vérifier la méthode
26         boolean isLocked = vault.isLocked();
27         //Assert isClosed
28         assertEquals(isLocked, true, "Echec fonction est ferme");
29
30         //Act / Assert
31         assertThrows(BLLException.class, () -> vault.lock("1234"));
32     }
33
34 public class TestOpenVault {
35
36     @Test
37     public void testOpenVaultPasswordOk() throws BLLException {
38
39         //Arrange
40         vault.lock("1234");
41
42         //Act open with code
43         vault.unlock("1234");
44         String password = vault.getPassword();
45         //Assert
46         assertEquals(password, "1234", "Echec fonction unLock");
47     }
48
49     @Test
50     public void testOpenVaultIsOpened() throws BLLException {
51
52         //Arrange
53         vault.lock("1234");
54
55         //Act open with code
56         vault.unlock("1234");
57         boolean isLocked = vault.isLocked();
58         //Assert
59         assertEquals(isLocked, false, "Echec fonction unLock");
60     }
61
62     @Test
63     public void testWrongPassword() throws BLLException {
64         //Arrange - préparer le test
65         Vault vault = new Vault();
66
67         //Act - lock the vault
68         vault.lock("1234");
69
70         //Act open with code with wrong password
71         vault.unlock("1235");
72         boolean isLocked = vault.isLocked();
73         //Assert
74         assertEquals(isLocked, true, "Echec fonction unLock - mauvais password");
75     }
76
77     @Test
78     public void testDejaOuvert() throws BLLException {
79         //Arrange - préparer le test
80         Vault vault = new Vault();
81
82         //Act / Assert
83         assertThrows(BLLException.class, () -> vault.unlock("1235"));
84     }
85
86 public class TestAjouterObjet {
87     Vault vault = new Vault();
88
89     @Test
90     public void testAddObject() throws BLLException {
91         //Arrange
92         Storable gold = new Gold(20);
93
94         //Act add Object in the vault
95         int listObjDansCoffre = vault.addObject(gold);
96
97         //Assert
98         assertEquals(listObjDansCoffre, 1, "Echec fonction addObject");
99     }
100
101     @Test
102     public void testItemIsInTheVault() throws BLLException {
103         //Arrange
104         Storable gold = new Gold(20);
105         ArrayList<Storable> listObjects = vault.getObjectsInside();
106
107         //Act add Object in the vault
108         vault.addObject(gold);
109
110         //Assert
111         assertEquals(listObjects.get(0), gold, "Echec fonction addObject");
112     }
113 }
114
115 public class TestRemoveObject {
116
117     Vault vault = new Vault();
118
119     @Test
120     public void testRemoveObject() throws BLLException {
121         //Arrange
122         Storable gold = new Gold(20);
123         vault.addObject(gold);
124
125         //Act remove Object in the vault
126         int listObjectSize = vault.removeObject(gold);
127
128         //Assert
129         assertEquals(listObjectSize, 0, "Echec fonction removeObject");
130     }
131
132
133     @Test
134     public void testRemoveObjectVaultClosed() throws BLLException {
135         //Arrange
136         Storable gold = new Gold(20);
137         vault.addObject(gold);
138         vault.lock("1234");
139
140         //Assert and act
141         assertThrows(BLLException.class, () -> vault.removeObject(gold));
142     }
143
144 }
```