File - C:\Users\sagis\IdeaProjects\CashFlowManagement\test\il\ac\hit\CashFlowManagement\test\UserLogicTest.java

```java
package il.ac.hit.CashFlowManagement.test;

import il.ac.hit.CashFlowManagement.exception.JDBCDataBaseException;
import il.ac.hit.CashFlowManagement.model.JDBCDataBase;
import il.ac.hit.CashFlowManagement.model.UserLogic;
import org.junit.After;
import org.junit.Before;
import org.junit.Test;

import java.sql.SQLException;

import static org.junit.Assert.assertEquals;
import static org.junit.Assert.assertTrue;

public class UserLogicTest {
    UserLogic userLogic = new UserLogic();
    String userName = "testUserName", password = "testPassword", country = "testCountry", gander = "testGander";

    @Before
    public void setUp() {
        userLogic.addUser(userName, password, country, gander);
    }

    @After
    public void tearDown() throws JDBCDataBaseException, SQLException {
        while(userLogic.checkIfExist("userName") || userLogic.checkIfExist(userName)) {
            JDBCDataBase.getInstance().getStatement().executeUpdate("DELETE FROM Registered_Users WHERE UserName = 'testUserName'");
            JDBCDataBase.getInstance().getStatement().executeUpdate("DELETE FROM Registered_Users WHERE UserName = 'userName'");
        }
    }

    @Test
    public void checkIfUserExist() throws JDBCDataBaseException, SQLException {

        assertEquals(false, userLogic.checkIfExist("userName"));
        assertTrue(userLogic.checkIfExist(userName));
    }

    @Test
    public void addUser() {
```

```
41          assertEquals(false, userLogic.checkIfExist("userName"));
42          userLogic.addUser("userName", password, country, gander);
43          assertTrue(userLogic.checkIfExist("userName"));
44      }
45
46      @Test
47      public void checkUserPassword() {
48          assertEquals(true, userLogic.checkUserPassword(userName,
    password));
49          assertEquals(false, userLogic.checkUserPassword(userName,
    password + "1"));
50      }
51  }
52
```

```java
package il.ac.hit.CashFlowManagement.test;

import il.ac.hit.CashFlowManagement.exception.JDBCDataBaseException;
import il.ac.hit.CashFlowManagement.model.JDBCDataBase;
import org.apache.log4j.BasicConfigurator;
import org.junit.Test;

import static junit.framework.TestCase.assertNotNull;

public class JDBCDataBaseTest {
    public JDBCDataBaseTest(){
        BasicConfigurator.configure();
    }

    @Test
    public void getInstance() {
        try {
            assertNotNull(JDBCDataBase.getInstance());
        } catch (JDBCDataBaseException e) {
            e.printStackTrace();
        }
    }

    @Test
    public void getStatement() {
        try {
            assertNotNull(JDBCDataBase.getInstance().getStatement());
        } catch (JDBCDataBaseException e) {
            e.printStackTrace();
        }
    }

    @Test
    public void getConnection() {
        try {
            assertNotNull(JDBCDataBase.getInstance().getConnection
());
        } catch (JDBCDataBaseException e) {
            e.printStackTrace();
        }
    }

}

```

File - C:\Users\sagis\IdeaProjects\CashFlowManagement\test\il\ac\hit\CashFlowManagement\test\ExpensesLogicTest.java

```java
package il.ac.hit.CashFlowManagement.test;

import il.ac.hit.CashFlowManagement.model.Expense;
import il.ac.hit.CashFlowManagement.model.ExpensesLogic;
import il.ac.hit.CashFlowManagement.model.JDBCDataBaseLogic;
import il.ac.hit.CashFlowManagement.view.LoginForm;
import il.ac.hit.CashFlowManagement.viewmodel.IViewModel;
import il.ac.hit.CashFlowManagement.viewmodel.ManagementViewModel;
import org.apache.log4j.BasicConfigurator;
import org.junit.After;
import org.junit.Test;

import java.sql.ResultSet;

import static org.junit.Assert.assertEquals;

public class ExpensesLogicTest {
    private JDBCDataBaseLogic dataBaseLogic = JDBCDataBaseLogic.getInstance();
    private String expenseDate = "01/01/2020", expenseClassification = "testClassification", expenseDetails = "testDetails";
    private double expenseCost = 5.55;
    private IViewModel viewModel;
    private String nameOfTable;
    private ExpensesLogic expensesLogic;
    private Expense expense = new Expense(expenseDate, expenseClassification, expenseDetails, expenseCost);

    public ExpensesLogicTest(){
        BasicConfigurator.configure();
        LoginForm.username = "testUserName";
        viewModel = ManagementViewModel.getInstance();
        nameOfTable = LoginForm.username.toUpperCase() + "_Expenses".toUpperCase();
        expensesLogic = new ExpensesLogic();
    }

    @After
    public void tearDown() throws Exception {
        dataBaseLogic.removeTable(nameOfTable);
    }

    @Test
    public void addExpense() throws Exception {
        expensesLogic.addExpense(expense);
        ResultSet rs = expensesLogic.getAllUserExpenses();
```

```
43          rs.next();
44          String date = rs.getString("Date");
45          String cost = rs.getString("Cost");
46          String classification = rs.getString("Classification");
47          String details = rs.getString("Details");
48          double costDouble = Double.parseDouble(cost);
49
50          assertEquals(expenseDate, date);
51          assertEquals((long)expenseCost, (long)costDouble);
52          assertEquals(expenseClassification, classification);
53          assertEquals(expenseDetails, details);
54      }
55
56      @Test
57      public void getAllUserExpenses() throws Exception {
58          int expensesCounter = 0;
59          expensesLogic.addExpense(expense);
60          ResultSet rs = expensesLogic.getAllUserExpenses();
61          while (rs.next()){
62              expensesCounter++;
63          }
64          assertEquals(1, expensesCounter);
65      }
66  }
67
```

```java
package il.ac.hit.CashFlowManagement.test;


import il.ac.hit.CashFlowManagement.exception.JDBCDataBaseLogicException;
import il.ac.hit.CashFlowManagement.model.JDBCDataBaseLogic;
import org.apache.log4j.BasicConfigurator;

import static org.hamcrest.CoreMatchers.is;
import static org.junit.Assert.*;


public class JDBCDataBaseLogicTest {
    private JDBCDataBaseLogic dataBaseLogic = JDBCDataBaseLogic.getInstance();
    private final String iNameOfTable = "Test";

    public JDBCDataBaseLogicTest(){
        BasicConfigurator.configure();
    }

    @org.junit.Before
    public void setUp() throws Exception {
        dataBaseLogic.createTableIfNotExist(iNameOfTable,"Password", "varchar(300)");
    }


    @org.junit.After
    public void tearDown() throws Exception {
        dataBaseLogic.removeTable(iNameOfTable);
    }

    @org.junit.Test
    public void createTableIfNotExist() throws Exception {
        String tableName = "CreateTableTest";

        assertEquals(true,dataBaseLogic.createTableIfNotExist(tableName,"Password", "varchar(300)"));
        assertEquals(false,dataBaseLogic.createTableIfNotExist(tableName,"Password", "varchar(300)"));
        dataBaseLogic.removeTable(tableName);
        try{
            dataBaseLogic.createTableIfNotExist(iNameOfTable,"Password");
            fail("Expected exception has not been thrown");
```

```
41          } catch (Exception e) {
42              assertThat(e.getMessage(), is("Please enter the
    parameters as follows: Param1Name ,Param1Type, Param2Name ,Param2Type
     .... , ParamName ,ParamType"));
43          }
44      }
45
46      @org.junit.Test
47      public void insertValue() throws Exception {
48          String params = "'" + "134" + "'";
49
50          dataBaseLogic.insertValue(iNameOfTable,params);
51          try {
52              dataBaseLogic.insertValue(iNameOfTable, null);
53              // If the exception is not thrown, the test will fail
54              fail("Expected exception has not been thrown");
55          } catch (Exception e) {
56          assertThat(e.getMessage(), is("Argument for @NotNull
    parameter 'iParameters' of il/ac/hit/CashFlowManagement/model/
    JDBCDataBaseLogic.insertValue must not be null"));
57          }
58      }
59
60      @org.junit.Test
61      public void query() throws JDBCDataBaseLogicException {
62          assertNotNull(dataBaseLogic.query("select * from " +
    iNameOfTable));
63
64          try {
65              assertNotNull(dataBaseLogic.query("select * from " +
    iNameOfTable + "1"));
66              fail("Expected exception has not been thrown");
67          } catch (Exception e) {
68              assertThat(e.getMessage(), is("query execution failed
    Table/View 'TEST1' does not exist."));
69          }
70
71          try {
72              dataBaseLogic.query("create table TEST (Password varchar(
    300))");
73              fail("Expected exception has not been thrown");
74          } catch (Exception e) {
75              assertThat(e.getMessage(), is("query execution failed
    Table/View 'TEST' already exists in Schema 'APP'."));
76          }
77      }
```

```
78
79      @org.junit.Test
80      public void removeTable() throws JDBCDataBaseLogicException {
81          String tableName = "RemoveTestTable";
82
83          dataBaseLogic.createTableIfNotExist(tableName,"Password", "
    varchar(300)");
84          dataBaseLogic.removeTable(tableName);
85
86          try {
87              dataBaseLogic.removeTable(tableName+"1");
88              fail("Expected exception has not been thrown");
89          } catch (Exception e) {
90              assertThat(e.getMessage(), is("failed to remove
    RemoveTestTable1 from DB"));
91          }
92      }
93
94      @org.junit.Test
95      public void getInstance() throws JDBCDataBaseLogicException {
96          assertNotNull(JDBCDataBaseLogic.getInstance());
97      }
98
99  }
```

```java
package il.ac.hit.CashFlowManagement.test;

import il.ac.hit.CashFlowManagement.exception.FormCastingException;
import il.ac.hit.CashFlowManagement.exception.
GetAllUserExpensesException;
import il.ac.hit.CashFlowManagement.exception.JDBCDataBaseException;
import il.ac.hit.CashFlowManagement.model.Expense;
import il.ac.hit.CashFlowManagement.model.ExpensesLogic;
import il.ac.hit.CashFlowManagement.model.JDBCDataBase;
import il.ac.hit.CashFlowManagement.model.UserLogic;
import il.ac.hit.CashFlowManagement.view.LoginForm;
import il.ac.hit.CashFlowManagement.viewmodel.ManagementViewModel;
import org.junit.After;
import org.junit.Before;
import org.junit.Test;

import java.sql.ResultSet;
import java.sql.SQLException;

import static org.junit.Assert.assertEquals;
import static org.junit.Assert.assertNotNull;

public class ManagementViewModelTest {
    private ManagementViewModel viewModel;
    private UserLogic userLogic = new UserLogic();
    private String userName = "testUserName", password = "
testPassword", country = "testCountry", gander = "testGander",
nameOfTable;
    private String expenseDate = "01/01/2020", expenseClassification
 = "testClassification", expenseDetails = "testDetails";
    private double expenseCost = 5.55;
    private Expense expense = new Expense(expenseDate,
expenseClassification, expenseDetails, expenseCost);


    public ManagementViewModelTest(){
        LoginForm.username = "testUserName";
        viewModel = ManagementViewModel.getInstance();
        viewModel.setExpensesLogic(new ExpensesLogic());
        nameOfTable = LoginForm.username.toUpperCase() + "_Expenses".
toUpperCase();
    }

    @Before
    public void setUp() {
        userLogic.addUser(userName, password, country, gander);
```

```
41      }
42
43      @After
44      public void tearDown() throws JDBCDataBaseException, SQLException
        {
45          while(userLogic.checkIfExist("userName") || userLogic.
    checkIfExist(userName)) {
46              JDBCDataBase.getInstance().getStatement().executeUpdate("
    DELETE FROM Registered_Users WHERE UserName = 'testUserName'");
47              JDBCDataBase.getInstance().getStatement().executeUpdate("
    DELETE FROM Registered_Users WHERE UserName = 'userName'");
48          }
49      }
50
51      @Test
52      public void getInstance() {
53          assertNotNull(ManagementViewModel.getInstance());
54      }
55
56      @Test
57      public void verifyUser() {
58          assertEquals(true, viewModel.verifyUser(userName, password));
59      }
60
61      @Test
62      public void register() {
63          assertEquals(true, viewModel.register("userName", password,
    country, gander));
64          assertEquals(false, viewModel.register(userName, password,
    country, gander));
65      }
66
67      @Test
68      public void getAllExpenses() throws GetAllUserExpensesException,
    SQLException, FormCastingException {
69          ResultSet rs = viewModel.getAllExpenses();
70          int counter = 0;
71
72          while (rs.next()){
73              counter++;
74          }
75          assertEquals(0, counter);
76      }
77  }
78
```