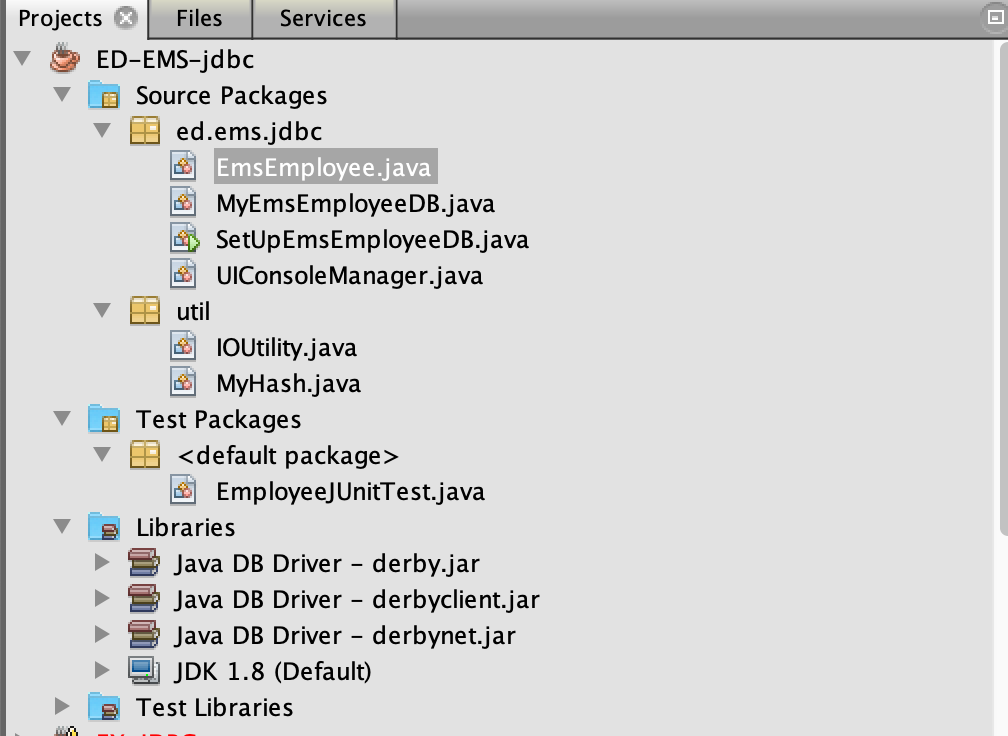
ENTERPRISE DEVELOPMENT

**Student name: Duy Thuc Pham – 101767225**

**Tutor name: Man Lau**

# Submission for Assignment 2.1

## Task 1. Complete Lab\_02\_Database\_Connectivity



## Task 2. Add CRUD methods

### Create Employee Method

public boolean createRecord(EmsEmployee emsEmployee) {

if (emsEmployee != null) {

// TODO: Going to do the business here

// Check the record exist in the current database

EmsEmployee existedEmployee = getRecord(emsEmployee.getEmpid());

// If the record does not exist in the database

if (existedEmployee == null) {

insertEmployee(emsEmployee);

return true;

}

return false;

}

return false;

}

private void insertEmployee(EmsEmployee emsEmployee) {

Connection cnnct = null;

PreparedStatement pStmnt = null;

try {

cnnct = getConnection();

String preQueryStatement

= "INSERT INTO EMS\_EMPLOYEE VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)";

pStmnt = cnnct.prepareStatement(preQueryStatement);

pStmnt.setString(1, emsEmployee.getEmpid());

pStmnt.setString(2, emsEmployee.getName());

pStmnt.setString(3, emsEmployee.getPassword());

pStmnt.setString(4, emsEmployee.getEmail());

pStmnt.setString(5, emsEmployee.getPhone());

pStmnt.setString(6, emsEmployee.getAddress());

pStmnt.setString(7, emsEmployee.getSecqn());

pStmnt.setString(8, emsEmployee.getSecans());

pStmnt.setString(9, emsEmployee.getBsbid());

pStmnt.setString(10, emsEmployee.getAccid());

pStmnt.setDouble(11, emsEmployee.getSalary());

pStmnt.setString(12, emsEmployee.getAppgroup());

pStmnt.setBoolean(13, emsEmployee.isActive());

pStmnt.setString(14, emsEmployee.getMemo());

pStmnt.executeUpdate();

} catch (SQLException ex) {

while (ex != null) {

ex.printStackTrace();

ex = ex.getNextException();

}

} catch (IOException ex) {

ex.printStackTrace();

} finally {

if (pStmnt != null) {

try {

pStmnt.close();

} catch (SQLException e) {

}

}

if (cnnct != null) {

try {

cnnct.close();

} catch (SQLException sqlEx) {

}

}

}

}

### Get Employee Method

public EmsEmployee getRecord(String empid) {

if (empid == null) {

return null;

}

Connection connection = null;

try {

connection = getConnection();

Statement stmt = connection.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM EMS\_EMPLOYEE WHERE empid='" + empid + "'");

if (rs.next()) {

EmsEmployee employee = extractEmployeeFromResultSet(rs);

return employee;

}

} catch (Exception ex) {

ex.printStackTrace();

}

return null;

}

private EmsEmployee extractEmployeeFromResultSet(ResultSet rs) throws SQLException {

EmsEmployee employee = new EmsEmployee();

employee.setAccid(rs.getString("accid"));

employee.setActive(rs.getBoolean("active"));

employee.setAddress(rs.getString("address"));

employee.setAppgroup(rs.getString("appgroup"));

employee.setBsbid(rs.getString("bsbid"));

employee.setEmail(rs.getString("email"));

employee.setEmpid(rs.getString("empid"));

employee.setMemo(rs.getString("memo"));

employee.setName(rs.getString("name"));

employee.setPassword(rs.getString("password"));

employee.setPhone(rs.getString("phone"));

employee.setSalary(rs.getDouble("salary"));

employee.setSecans(rs.getString("secans"));

employee.setSecqn(rs.getString("secqn"));

return employee;

}

### Update Employee Method

public boolean updateRecord(EmsEmployee emsEmployee) {

if (emsEmployee != null) {

Connection cnnct = null;

PreparedStatement pStmnt = null;

try {

cnnct = getConnection();

String preQueryStatement

= "UPDATE EMS\_EMPLOYEE SET name=?, password=?, email=?, phone=?, address=?, secqn=?, secans=?, bsbid=?, accid=?, salary=?, appgroup=?, active=?, memo=? WHERE empid=?";

pStmnt = cnnct.prepareStatement(preQueryStatement);

pStmnt.setString(1, emsEmployee.getName());

pStmnt.setString(2, emsEmployee.getPassword());

pStmnt.setString(3, emsEmployee.getEmail());

pStmnt.setString(4, emsEmployee.getPhone());

pStmnt.setString(5, emsEmployee.getAddress());

pStmnt.setString(6, emsEmployee.getSecqn());

pStmnt.setString(7, emsEmployee.getSecans());

pStmnt.setString(8, emsEmployee.getBsbid());

pStmnt.setString(9, emsEmployee.getAccid());

pStmnt.setDouble(10, emsEmployee.getSalary());

pStmnt.setString(11, emsEmployee.getAppgroup());

pStmnt.setBoolean(12, emsEmployee.isActive());

pStmnt.setString(13, emsEmployee.getMemo());

pStmnt.setString(14, emsEmployee.getEmpid());

int i = pStmnt.executeUpdate();

return i == 1;

} catch (SQLException ex) {

while (ex != null) {

ex.printStackTrace();

ex = ex.getNextException();

}

} catch (IOException ex) {

ex.printStackTrace();

} finally {

if (pStmnt != null) {

try {

pStmnt.close();

} catch (SQLException e) {

}

}

if (cnnct != null) {

try {

cnnct.close();

} catch (SQLException sqlEx) {

}

}

}

}

return false;

}

### Delete Employee Method

public boolean deleteRecord(String empid) {

if (empid != null) {

Connection connection = null;

try {

connection = getConnection();

Statement statement = connection.createStatement();

int i = statement.executeUpdate("DELETE FROM EMS\_EMPLOYEE WHERE empid='" + empid + "'");

return i == 1;

} catch (SQLException ex) {

ex.printStackTrace();

} catch (IOException ex) {

ex.printStackTrace();

} finally {

if (connection != null) {

try {

connection.close();

} catch (SQLException sqlEx) {

}

}

}

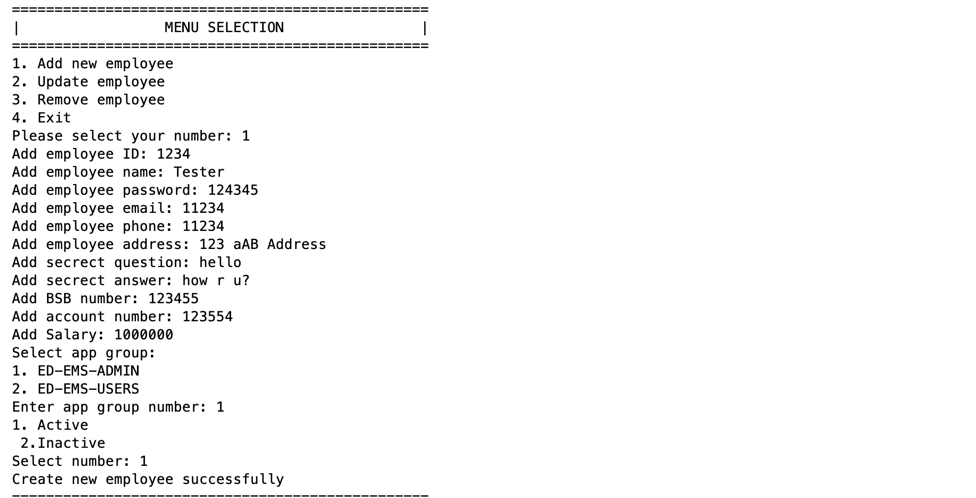
}

return false;

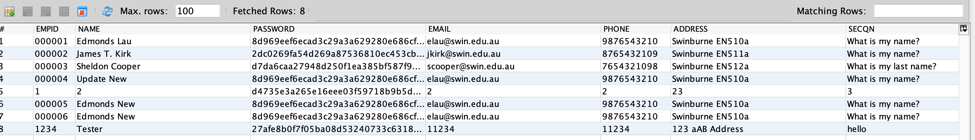
}

## Task 3. Develop Console Client Program

### UI



### Result



## Task 4. Testcases

@Test

public void testCreateEmployee() {

String memo1 = "123456";

String pwd1 = MyHash.getSHA256HashedString(memo1);

EmsEmployee emsEmployee1 = new EmsEmployee("000006", "Edmonds New", pwd1,

"elau@swin.edu.au", "9876543210", "Swinburne EN510a", "What is my name?",

"Edmonds", "123456", "12345678", 50000, "ED-EMS-ADMIN", true, memo1);

MyEmsEmployeeDB dao = new MyEmsEmployeeDB();

dao.createRecord(emsEmployee1);

EmsEmployee test1 = dao.getRecord("000006");

// Check if it has already created

assertEquals(test1.getEmpid(), emsEmployee1.getEmpid());

}

@Test

public void testUpdateEmployee() {

String memo1 = "123456";

String pwd1 = MyHash.getSHA256HashedString(memo1);

EmsEmployee emsEmployee1 = new EmsEmployee("000004", "Update New", pwd1,

"elau@swin.edu.au", "9876543210", "Swinburne EN510a", "What is my name?",

"Edmonds", "123456", "12345678", 50000, "ED-EMS-ADMIN", true, memo1);

MyEmsEmployeeDB dao = new MyEmsEmployeeDB();

dao.updateRecord(emsEmployee1);

EmsEmployee test1 = dao.getRecord("000004");

// Check if it has already update

assertEquals(test1.getName(), "Update New");

}

@Test

public void testDeleteEmployee() {

MyEmsEmployeeDB dao = new MyEmsEmployeeDB();

dao.deleteRecord("000006");

EmsEmployee test1 = dao.getRecord("000006");

// Check if it has already deleted

assertNull(test1);

}

@Test

public void testGetEmployeeById() {

MyEmsEmployeeDB dao = new MyEmsEmployeeDB();

EmsEmployee test1 = dao.getRecord("000001");

// Check if it has already deleted

assertNotNull(test1);

}

@Test

public void testGetAllEmployee() {

MyEmsEmployeeDB dao = new MyEmsEmployeeDB();

Set set1 = dao.getAll();

assertEquals(set1.size(), 6);

}

### Test Result

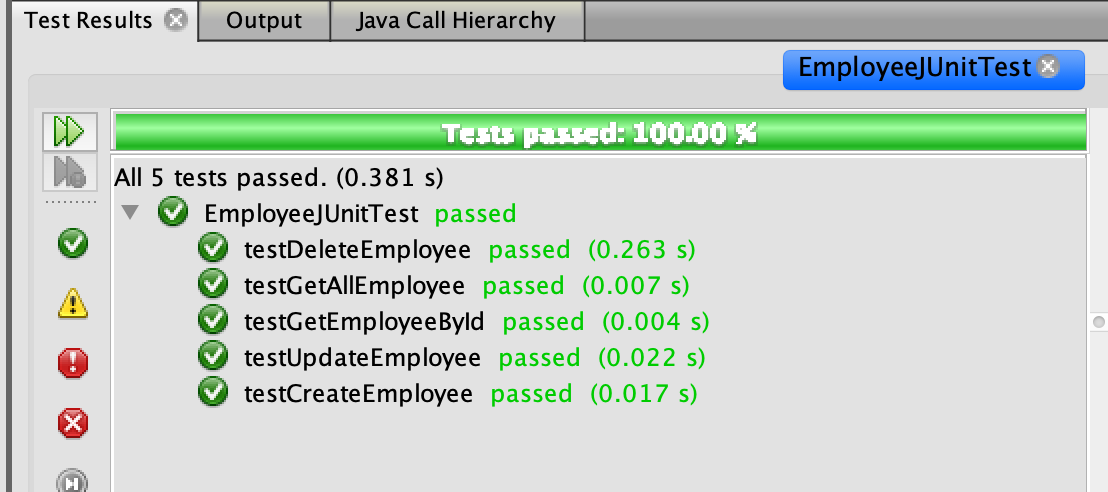


Figure 1: Test Result

## Task 6. Answer questions

### 1. What is the role of the EmsEmployee class?

EmsEmployee class is the Data Transfer Object (DTO), which is responsible for transferring data between classes and modules. It should not handle any business logics like what DAO does.

### 2. In Machine A, if a program (e.g. SetUpMyEmployee) calls "emsEmployee.setName("ABC")", will this change the corresponding value in the database server? Why or Why not?

No. Setting the new value for emsEmployee will not change the value in database server but only change the value of this particular instance. Because firstly, EmsEmployee is a DTO object which only uses to store, retrieval, serialization and deserialization. Secondly, it does not access to the database to invoke the SQL queries like DAO object, and thus data in database server will remain unchanged.

### 3. In Machine B, if MyEmsEmployeeDB calls "emsEmployee.setName("ABC")", will this change the corresponding value in the database server? Why or Why not?

No. Setting the new value for emsEmployee will not change the value in database server but only change the value of this particular instance. Because firstly, EmsEmployee is a DTO object which only uses to store, retrieval, serialization and deserialization. Secondly, it does not access to the database to invoke the SQL queries like DAO object, and thus data in database server will remain unchanged.