## Laboratory 7

Title of the Laboratory Exercise: To integrate database using JDBC to web application

1. Introduction and Purpose of Experiment

   Students will learn to implement JDBC component for all the functional requirements identified.

2. Aim and Objectives

   Aim

3. Experimental Procedure
4. Calculations/Computations/Algorithms

UserDao.java

```
package webarch.dao;

import java.math.BigInteger;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import webarch.db.DBConnection;

public class UserDao {

    public static String getMd5(String input) {
        try {

            // Static getInstance method is called with hashing MD5
            MessageDigest md = MessageDigest.getInstance("MD5");

            // digest() method is called to calculate message digest
            //  of an input digest() return array of byte
            byte[] messageDigest = md.digest(input.getBytes());

            // Convert byte array into signum representation
            BigInteger no = new BigInteger(1, messageDigest);

            // Convert message digest into hex value
```

```
            String hashtext = no.toString(16);
            while (hashtext.length() < 32) {
                hashtext = "0" + hashtext;
            }
            return hashtext;
        }

        // For specifying wrong message digest algorithms
        catch (NoSuchAlgorithmException e) {
            throw new RuntimeException(e);
        }
    }

    public static boolean registerUser(String username, String password, String fullname, String usnno, String d
ept, String course ) {
        Connection conn = DBConnection.getDbConnection();

        try {
            PreparedStatement stmt = conn.prepareStatement("INSERT INTO `STUDENT_LOGIN` (`user_name`, `hashed_pa
ssword`) VALUES(?, ?)", Statement.RETURN_GENERATED_KEYS);
            stmt.setString(1, username);
            stmt.setString(2, getMd5(password));

            stmt.executeUpdate();

            ResultSet rs = stmt.getGeneratedKeys();
            if (rs.next()) {
                Integer id = rs.getInt(1);

                PreparedStatement stmt2 = conn.prepareStatement("INSERT INTO STUDENT( id, reg_no, name, departme
nt, course, contact_no ) VALUES (?, ?, ?, ?, ?, ?)");
                stmt2.setInt(1, id);
                stmt2.setString(2, usnno);
                stmt2.setString(3, fullname);
                stmt2.setString(4, dept);
                stmt2.setString(5, course);
                stmt2.setString(6, "9999999999"); // hardcoded value for now

                int count = stmt2.executeUpdate();

                if (count > 0)
                    return true;
                else
                    return false;
            }
            rs.close();
        } catch (SQLException e) {
            e.printStackTrace();
            return false;
        }

        return false;

    }
```

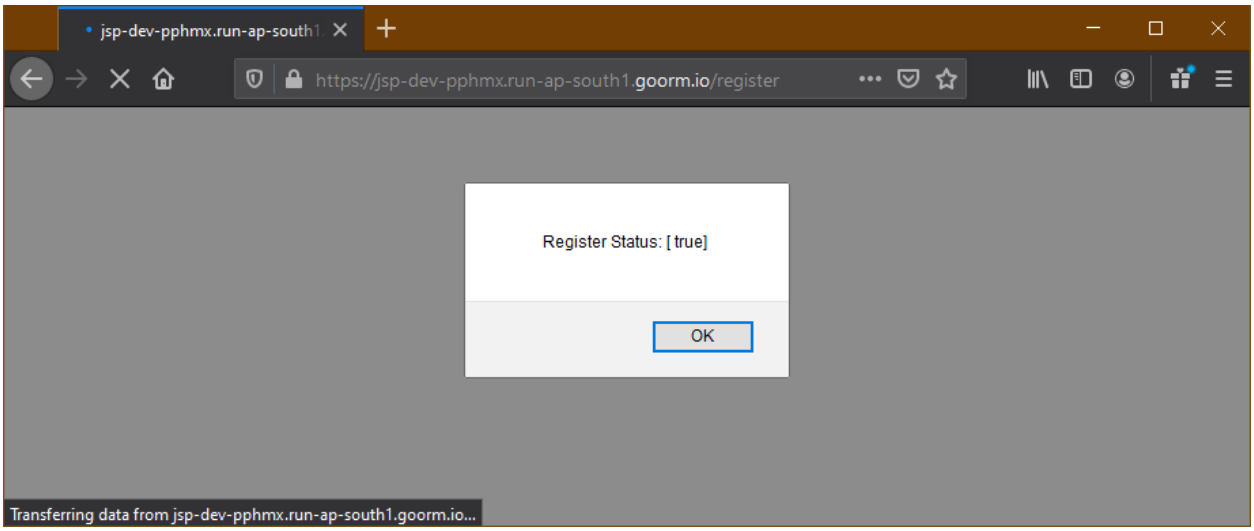
```
}
```

DBConnection.java

```
package webarch.db;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class DBConnection {
    private static Connection conn;

    public static Connection getDbConnection() {
        if (conn == null) {
            try {
                Class.forName("com.mysql.jdbc.Driver");
                conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/webarch", "root", "");
            } catch (SQLException e) {
                e.printStackTrace();
            } catch (ClassNotFoundException e) {
                e.printStackTrace();
            }
        }

        return conn;
    }
}
```

5. Presentation of Results

Registration of User



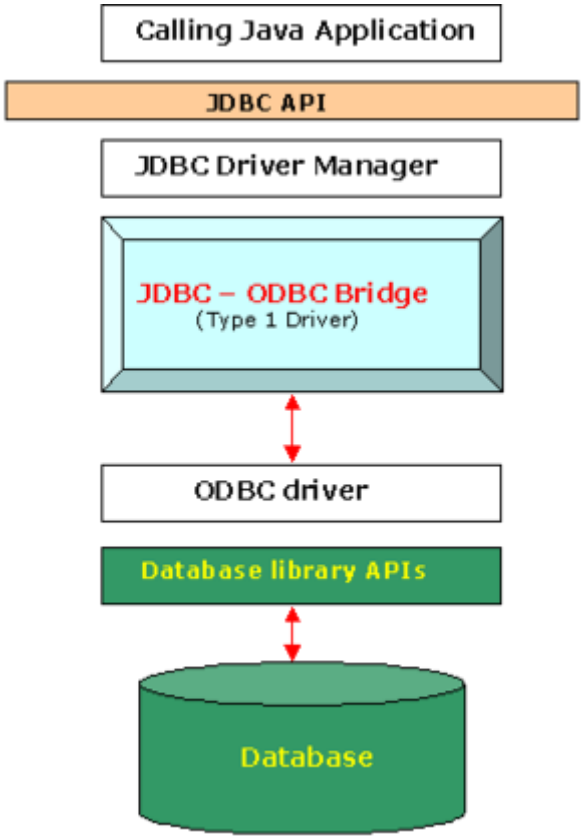

User successfully added in the database

```
mysql> select * from STUDENT_LOGIN where user_name="satyajitghana";
+----+---------------+----------------------------------+
| id | user_name     | hashed_password                  |
+----+---------------+----------------------------------+
| 11 | satyajitghana | e2d60cfcaaf71ad64ba36fbd103374c5 |
+----+---------------+----------------------------------+
1 row in set (0.00 sec)

mysql> select * from STUDENT where id=11;
+----+-------------+---------------+------------+--------+------------+
| id | reg_no      | name          | department | course | contact_no |
+----+-------------+---------------+------------+--------+------------+
| 11 | 17CSET002159 | satyajit ghana | CSE        | B.Tech | 9999999999 |
+----+-------------+---------------+------------+--------+------------+
1 row in set (0.00 sec)

mysql>
```

6. Analysis and Discussions

JDBC Driver is a software component that enables java application to interact with the database. There are 4 types of JDBC drivers:

1. JDBC-ODBC bridge driver
2. Native-API driver (partially java driver)
3. Network Protocol driver (fully java driver)
4. Thin driver (fully java driver)




7. Conclusions

JDBC can be used with Java to Connect to an existing DB by,

```
public Connection getConnection() throws SQLException {

    Connection conn = null;
    Properties connectionProps = new Properties();
    connectionProps.put("user", this.userName);
    connectionProps.put("password", this.password);

    if (this.dbms.equals("mysql")) {
        conn = DriverManager.getConnection(
                    "jdbc:" + this.dbms + "://" +
                    this.serverName +
                    ":" + this.portNumber + "/",
                    connectionProps);
    } else if (this.dbms.equals("derby")) {
```

```
            conn = DriverManager.getConnection(
                        "jdbc:" + this.dbms + ":" +
                        this.dbName +
                        ";create=true",
                        connectionProps);
        }
        System.out.println("Connected to database");
        return conn;
    }
```

8. Comments

a. Limitations of Experiments

Performance is degraded since the JDBC call goes through the bridge to the ODBC driver then to the native database connectivity interface. The results are then sent back through the reverse process

b. Limitations of Results

None

c. Learning happened

We learnt how to connect to MySQL DB using JDBC from Java.

d. Recommendations

None

| Component | Max Marks | Marks Obtained |
|---|---|---|
| Viva | 6 | |
| Results | 7 | |
| Documentation | 7 | |
| Total | 20 | |