Task 2.2 Evidence

Create Record method

```
public boolean createRecord(MyUser myuser) {
    Connection cnnct = null;
    PreparedStatement pStmnt = null;

    try {
        cnnct = getConnection();
        // Check if the User exists and return false if it does
        String selectStatement = "SELECT * FROM MYUSER WHERE USERID ='" + myuser.getUserid() + "'";
        pStmnt = cnnct.prepareStatement(selectStatement);
        ResultSet rslt = pStmnt.executeQuery();
        if (rslt.next()) {
            return false;
        } else {
            // Reset var for next query
            pStmnt = null;
            String query = "INSERT INTO MYUSER VALUES (?, ?, ?, ?, ?, ?, ?, ?)";
            pStmnt = cnnct.prepareStatement(query);

            pStmnt.setString(1, myuser.getUserid());
            pStmnt.setString(2, myuser.getName());
            pStmnt.setString(3, myuser.getPassword());
            pStmnt.setString(4, myuser.getEmail());
            pStmnt.setString(5, myuser.getPhone());
            pStmnt.setString(6, myuser.getAddress());
            pStmnt.setString(7, myuser.getSecQn());
            pStmnt.setString(8, myuser.getSecAns());
            int rowCount = pStmnt.executeUpdate();
            if (rowCount == 0) {
                throw new SQLException("Cannot insert records!");
            }
        }

    } 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) {
            }
        }
    }
    // If this code is reached the user was inserted successfully
    return true;
}
```

Get Record Method

```
public MyUser getRecord(String userId) {
    Connection cnnct = null;
    PreparedStatement pStmnt = null;
    MyUser result = null;
    try {
        cnnct = getConnection();

        String selectStatement = "SELECT * FROM MYUSER WHERE USERID ='" + userId + "'";
        pStmnt = cnnct.prepareStatement(selectStatement);
        ResultSet rslt = pStmnt.executeQuery();

        while (rslt.next()) {
            result = new MyUser(rslt.getString("USERID"),
                    rslt.getString("NAME"),
                    rslt.getString("PASSWORD"),
                    rslt.getString("EMAIL"),
                    rslt.getString("PHONE"),
                    rslt.getString("ADDRESS"),
                    rslt.getString("SECQN"),
                    rslt.getString("SECANS")
            );
        }

    } 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 result;
}
```

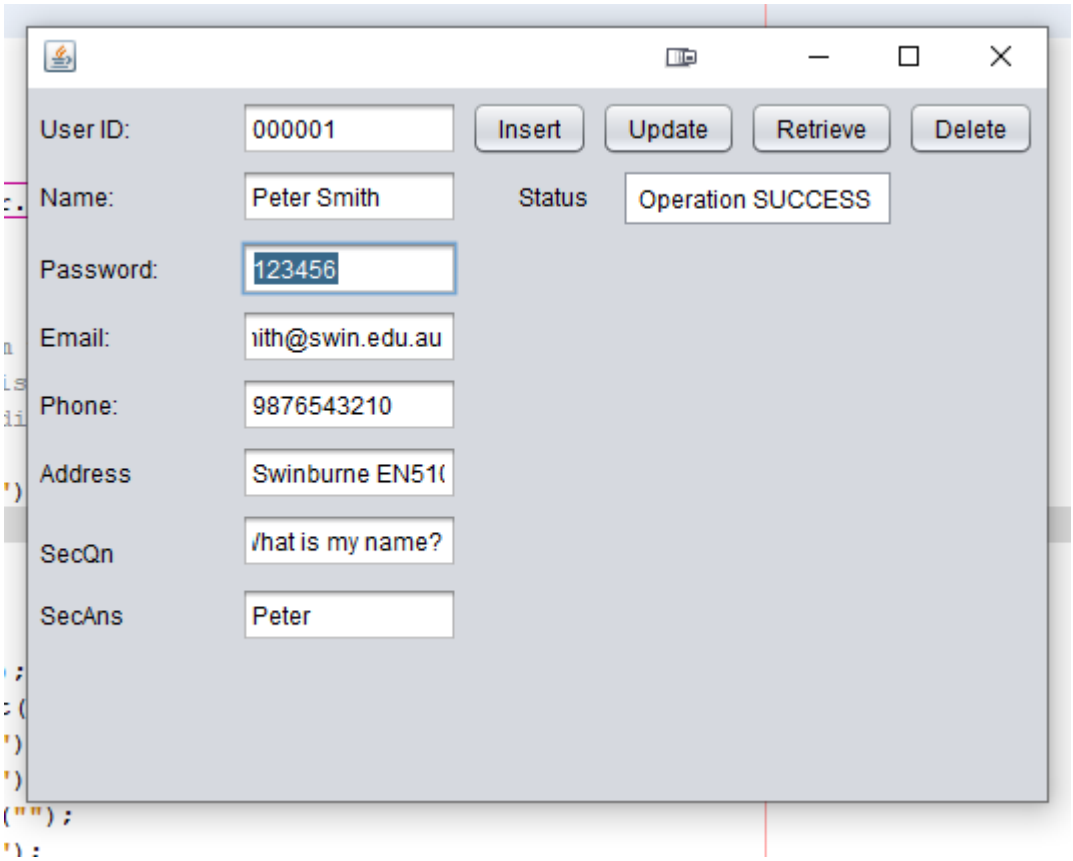
Update Record Method

```
public boolean updateRecord(MyUser myuser) {
    Connection cnnct = null;
    PreparedStatement pStmnt = null;

    try {
        cnnct = getConnection();
        // Check if the User exists and return false if it does
        String selectStatement = "SELECT * FROM MYUSER WHERE USERID ='" + myuser.getUserid() + "'";
        pStmnt = cnnct.prepareStatement(selectStatement);
        ResultSet rslt = pStmnt.executeQuery();
        if (!rslt.next()) {
            return false;
        } else {
            // Reset var for next query
            pStmnt = null;
            String query = "UPDATE MYUSER SET NAME = ?, "
                    + "PASSWORD = ?,"
                    + "EMAIL    = ?,"
                    + "PHONE    = ?,"
                    + "ADDRESS  = ?,"
                    + "SECQN    = ?,"
                    + "SECANS   = ?"
                    + "WHERE USERID = ?";
            pStmnt = cnnct.prepareStatement(query);

            pStmnt.setString(1, myuser.getName());
            pStmnt.setString(2, myuser.getPassword());
            pStmnt.setString(3, myuser.getEmail());
            pStmnt.setString(4, myuser.getPhone());
            pStmnt.setString(5, myuser.getAddress());
            pStmnt.setString(6, myuser.getSecQn());
            pStmnt.setString(7, myuser.getSecAns());
            pStmnt.setString(8, myuser.getUserid());

            int rowCount = pStmnt.executeUpdate();
            if (rowCount == 0) {
                throw new SQLException("Cannot insert records!");
            }
        }

    } 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) {
            }
        }
    }
    // If this code is reached the user was Updated successfully
    return true;
}
```

Delete Record Method

```
public boolean deleteRecord(String userId) {
    Connection cnnct = null;
     PreparedStatement pStmnt = null;
     try {
         cnnct = getConnection();

         String selectStatement = "DELETE FROM MYUSER WHERE USERID = ?";
         pStmnt = cnnct.prepareStatement(selectStatement);
         pStmnt.setString(1, userId);
         Integer rslt = pStmnt.executeUpdate();

         if (rslt != 0)
         {
             return true;
         }

     } 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;
}
```

GUI



Task 4 : Questions

4.1:

The myuser class is neither a DAO or DTO as it provides no means of sending changes to the database. It's merely a representation of the user entity in the database as an object.

4.1.1:

From Wikipedia: "a DAO is a pattern that provides an abstract interface to some kind of database". The purpose of a DAO is to provide means to access information in the database without exposing database operations.

Here the database object acts as a DAO because it provides methods for storing and retrieving data in the database and does NOT expose the queries themselves to outside objects.

4.2 & 4.3

The answer to both of these is no. The setName() method is an automatically generated setter method for the user object. The setter methods only change the internal value of the name property of the object calling the method.