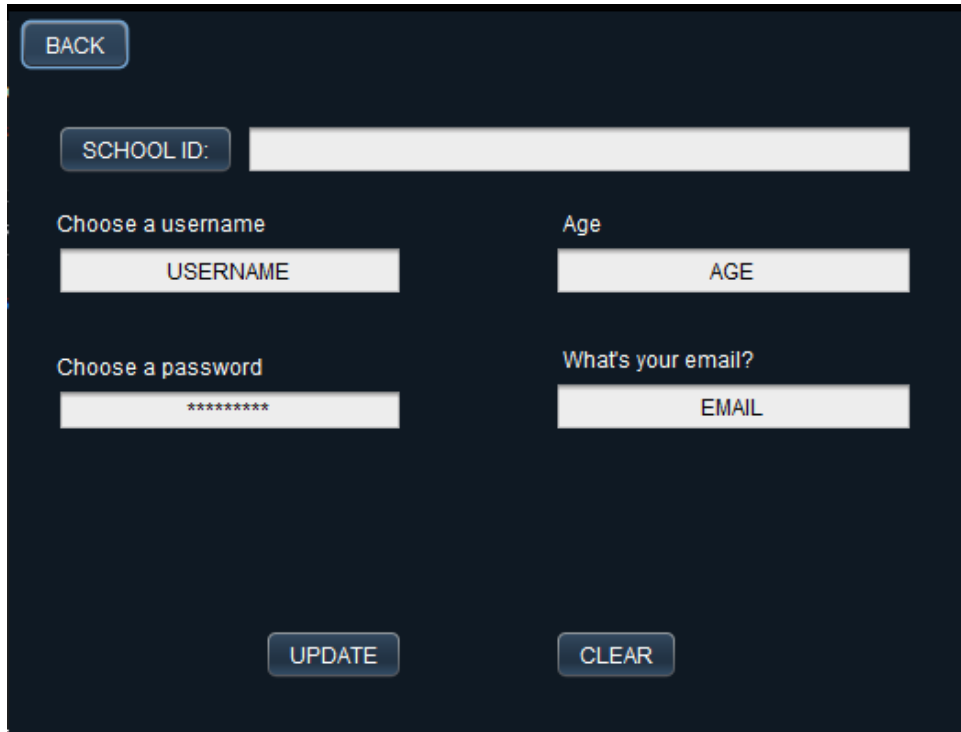


Engbino, Kaye P.

UPDATE UI

Teacher's Page (Admin)



A dark-themed web form for updating teacher information. It features a 'BACK' button at the top left. Below it is a 'SCHOOL ID:' label followed by a text input field. The form is divided into two columns. The left column has 'Choose a username' above a 'USERNAME' input field, and 'Choose a password' above a password input field with asterisks. The right column has 'Age' above an 'AGE' input field, and 'What's your email?' above an 'EMAIL' input field. At the bottom, there are 'UPDATE' and 'CLEAR' buttons.

BACK

SCHOOL ID:

Choose a username

USERNAME

Choose a password

Age

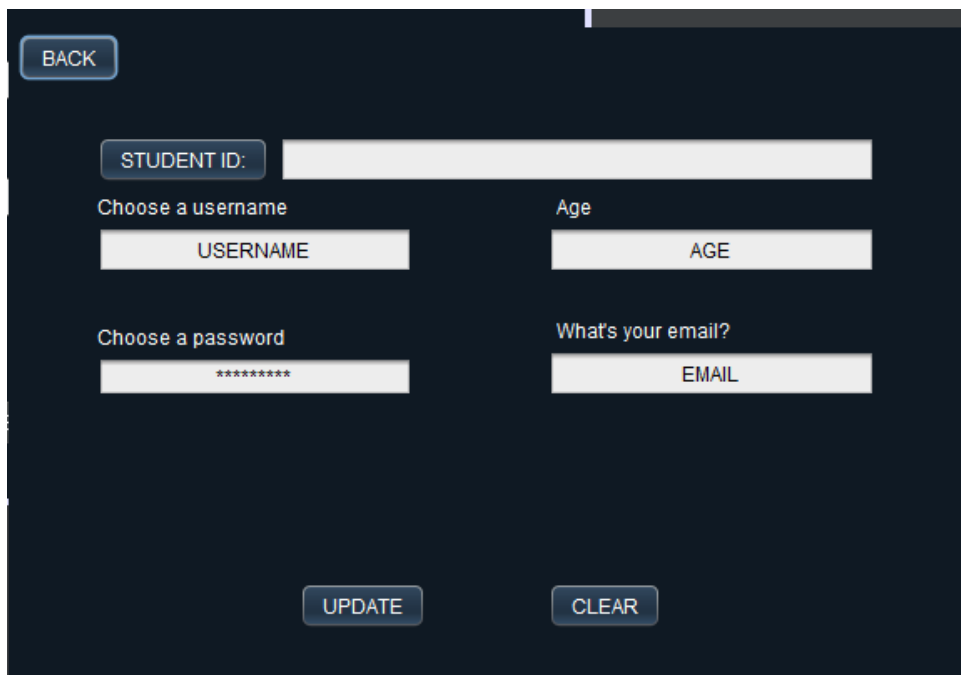
AGE

What's your email?

EMAIL

UPDATE CLEAR

Student's Page



A dark-themed web form for updating student information. It features a 'BACK' button at the top left. Below it is a 'STUDENT ID:' label followed by a text input field. The form is divided into two columns. The left column has 'Choose a username' above a 'USERNAME' input field, and 'Choose a password' above a password input field with asterisks. The right column has 'Age' above an 'AGE' input field, and 'What's your email?' above an 'EMAIL' input field. At the bottom, there are 'UPDATE' and 'CLEAR' buttons.

BACK

STUDENT ID:

Choose a username

USERNAME

Choose a password

Age

AGE

What's your email?

EMAIL

UPDATE CLEAR

DELETE UI

Teachers Page (Admin)

DELETE UI

Teachers Page (Admin)

BACK

Task Name

ID NUMBER	TASK NAME	DEADLINE	TASK DESCRIPTION

Student's Page

[illegible]

UPDATE CODES

```
private void btnUpdatePersonalInformationActionPerformed(java.awt.event.ActionEvent evt) {  
    int idn = Integer.parseInt(txtIDNumber.getText());  
    String pass = txtPassword.getText();  
    int age = Integer.parseInt(txtAge.getText());  
    String user = txtUsername.getText();  
    String email = txtEmail.getText();  
  
    boolean upd = dbh.updateTeacherRecord(idn, pass, age, user, email);  
  
    if(upd)  
        JOptionPane.showMessageDialog(this, "Successfully Signed Updated");  
    else  
        JOptionPane.showMessageDialog(this, "Sorry, you have invalid inputs at some fields");  
  
    new updateTeacherRecord().setVisible(true);  
    this.setVisible(false);  
}
```

```
private void btnUpdatePersonalInformation1ActionPerformed(java.awt.event.ActionEvent evt) {  
    int idn = Integer.parseInt(txtIDNumber.getText());  
    String pass = txtPassword.getText();  
    int age = Integer.parseInt(txtAge.getText());  
    String user = txtUsername.getText();  
    String email = txtEmail.getText();  
  
    boolean upd = dbh.updateStudentRecord(idn, pass, age, user, email);  
  
    if(upd)  
        JOptionPane.showMessageDialog(this, "Successfully Signed Updated");  
    else  
        JOptionPane.showMessageDialog(this, "Sorry, you have invalid inputs at some fields");  
  
    new updateTeacherRecord().setVisible(true);  
    this.setVisible(false);  
}
```

DELETE CODES

```
private void btnDeleteInformationActionPerformed(java.awt.event.ActionEvent evt) {  
    int idn = Integer.parseInt(txtIDNumber.getText());  
    String pass = txtPassword.getText();  
    int age = Integer.parseInt(txtAge.getText());  
    String user = txtUsername.getText();  
    String email = txtEmail.getText();  
  
    boolean del = dbh.deleteTeacherRecord(idn);  
  
    if(del)  
        JOptionPane.showMessageDialog(this, "Successfully Deleted");  
    else  
        JOptionPane.showMessageDialog(this, "Sorry, you have invalid inputs at some fields");  
  
    new deleteTeacherRecord().setVisible(true);  
    this.setVisible(false);  
}
```

```

private void deleteActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your code here
    JOptionPane.showMessageDialog(this, "Are you sure you want to delete this task?", "Delete Record", JOptionPane.YES_NO_CANCEL_OPTION);
    int response = JOptionPane.showConfirmDialog(this, "Are you sure you want to delete this task?", "Delete Record", JOptionPane.YES_NO_CANCEL_OPTION);
    if (response == JOptionPane.YES_OPTION) {
        // TODO add your code here
        // Get the task ID to delete
        int taskId = Integer.parseInt(taskIdField.getText());

        // Call the delete method
        try {
            // Get the database connection
            Connection conn = DriverManager.getConnection("jdbc:derby://localhost:1527/dbTaskManager", "kaye", "kaye");

            // Create a statement
            Statement stmt = conn.createStatement();

            // Execute the delete query
            String query = "DELETE FROM tasks WHERE task_id = " + taskId;
            stmt.executeUpdate(query);

            // Close the connection
            conn.close();

            // Show a message dialog
            JOptionPane.showMessageDialog(this, "Task Deleted Successfully", "Success", JOptionPane.INFORMATION_MESSAGE);
        } catch (SQLException e) {
            // Show an error message dialog
            JOptionPane.showMessageDialog(this, "Error deleting task: " + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
        }
    }
}

```

DBHelper

```

1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6
7  package registration.form;
8
9  import java.sql.Connection;
10 import java.sql.Date;
11 import java.sql.DriverManager;
12 import java.sql.ResultSet;
13 import java.sql.SQLException;
14 import java.sql.Statement;
15 import java.util.logging.Level;
16 import java.util.logging.Logger;
17
18 /**
19 *
20 * @author Kaye P. Engbino
21 * $Date: 11 Mar 21
22 * @Version 1.0
23 */
24 public class DBHelper {
25     Connection con = null;
26     Statement stmt = null;
27     private Object userStudent;
28
29     public void connectDB() throws Exception
30     {
31         con = DriverManager.getConnection("jdbc:derby://localhost:1527/dbTaskManager", "kaye", "kaye");
32         System.out.println("Connected to the database");
33     }

```

```

58 public boolean insertNewStudent(int idnum, String fname, String lname, String name, String pword, int age, String uname, String email)
59 {
60     boolean flag = false;
61
62     try {
63         stmt = con.createStatement();
64         String sql = "insert into tblStudentInfo values ('+idnum+', '"+fname+', '"+lname+', '"+name+', '"+pword+', '"+age+', '"+uname+', '"+email+')";
65
66         if(stmt.executeUpdate(sql) == 1)
67             flag = true;
68     } catch (SQLException ex) {
69         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
70     }
71
72     return flag;
73 }
74
75 public boolean insertNewTeacher(int idnum, String fname, String lname, String email, String uname, String pword, int age, String uname)
76 {
77     boolean flag = false;
78
79     try {
80         stmt = con.createStatement();
81         String sql = "insert into tblTeacherInfo values ('+idnum+', '"+fname+', '"+lname+', '"+email+', '"+uname+', '"+pword+', '"+age+', '"+uname+')";
82
83         if(stmt.executeUpdate(sql) == 1)
84             flag = true;
85     } catch (SQLException ex) {
86         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
87     }
88
89     return flag;
90 }

```

```

69 public boolean insertTaskStudent(int id, String name, String deadline, String description)
70 {
71     boolean flag = false;
72
73     try {
74         stmt = con.createStatement();
75         String sql = "Insert into studentTask values ('+id+', '"+name+', '"+deadline+', '"+description+')";
76
77         if(stmt.executeUpdate(sql) == 1)
78             flag = true;
79     } catch (SQLException ex) {
80         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
81     }
82
83     return flag;
84 }
85
86 public ResultSet displayAllRecords()
87 {
88     ResultSet rs = null;
89
90     try {
91         stmt = con.createStatement();
92         String sql = "Select * from tblStudentInfo";
93         rs = stmt.executeQuery(sql);
94     } catch (SQLException ex) {
95         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
96     }
97
98     return rs;
99 }

```

```
100
101 public ResultSet displayByLastName(String lastname)
102 {
103     ResultSet rs = null;
104
105     try {
106         stmt = con.createStatement();
107         String sql = "Select * from tblStudentInfo where lastname = '"+lastname+"'";
108         rs = stmt.executeQuery(sql);
109     } catch (SQLException ex) {
110         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
111     }
112
113     return rs;
114 }
115
116 public ResultSet displayAllTask()
117 {
118     ResultSet rs = null;
119
120     try {
121         stmt = con.createStatement();
122         String sql = "Select * from studentTask";
123         rs = stmt.executeQuery(sql);
124     } catch (SQLException ex) {
125         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
126     }
127
128     return rs;
129 }
```

```
130
131 public ResultSet displayAllTaskStudent(String name)
132 {
133     ResultSet rs = null;
134
135     try {
136         stmt = con.createStatement();
137         String sql = "Select * from studentTask where taskname = '"+name+"'";
138         rs = stmt.executeQuery(sql);
139     } catch (SQLException ex) {
140         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
141     }
142
143     return rs;
144 }
145
146 public ResultSet displayByTask(int id)
147 {
148     ResultSet rs = null;
149
150     try {
151         stmt = con.createStatement();
152         String sql = "Select * from studentTask where idnumber = "+id+"";
153         rs = stmt.executeQuery(sql);
154     } catch (SQLException ex) {
155         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
156     }
157
158     return rs;
159 }
160
```

```

160
161 public ResultSet displayByIDStudent(int id)
162 {
163     ResultSet rs = null;
164
165     try {
166         stmt = con.createStatement();
167         String sql = "Select * from tblStudentInfo where idnumber = "+id+"";
168         rs = stmt.executeQuery(sql);
169     } catch (SQLException ex) {
170         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
171     }
172
173     return rs;
174 }
175
176 public ResultSet displayByIDTeacher(int id)
177 {
178     ResultSet rs = null;
179
180     try {
181         stmt = con.createStatement();
182         String sql = "Select * from tblTeacherInfo where idnumber = "+id+"";
183         rs = stmt.executeQuery(sql);
184     } catch (SQLException ex) {
185         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
186     }
187
188     return rs;
189 }
190

```

```

191 public ResultSet displayByPassTeacher(int id)
192 {
193     ResultSet rs = null;
194
195     try {
196         stmt = con.createStatement();
197         String sql = "Select * from tblTeacherInfo where idnumber = "+id+"";
198         rs = stmt.executeQuery(sql);
199     } catch (SQLException ex) {
200         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
201     }
202
203     return rs;
204 }
205
206 public boolean updateStudentRecord(int idn, String pass, int age, String name, String email)
207 {
208     boolean flag = false;
209
210     try {
211         stmt = con.createStatement();
212         String sql = "Update tblStudentInfo set password = '"+pass"', age = '"+age"', name = '"+name"', email = '"+email"' where idnumber = "+idn+"";
213         if(stmt.executeUpdate(sql) == 1){
214             flag = true;
215         }
216     } catch (SQLException ex) {
217         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
218     }
219
220     return flag;
221 }
222

```



```

220 public boolean updateTaskFunction(int id, String name, int age, String desc, String result)
221 {
222     boolean flag = false;
223
224     try {
225         con = DBHelper.getConnection();
226         String sql = "Update studentTask set name = '"+name+"', age = '"+age+"', description = '"+desc+"', result = '"+result+"' where idnumber = '"+id+"'";
227         if(con.executeUpdate(sql) == 1)
228             flag = true;
229     } catch (SQLException ex) {
230         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
231     }
232
233     return flag;
234 }
235
236 public boolean updateTaskFunction(int id, String name, String description, String result)
237 {
238     boolean flag = false;
239
240     try {
241         con = DBHelper.getConnection();
242         String sql = "Update studentTask set name = '"+name+"', description = '"+description+"', result = '"+result+"' where idnumber = '"+id+"'";
243         if(con.executeUpdate(sql) == 1)
244             flag = true;
245     } catch (SQLException ex) {
246         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
247     }
248
249     return flag;
250 }
251
252 public boolean disconnectDB()
253 {
254     boolean flag = false;
255
256     try {
257         con.close();
258     } catch (SQLException ex) {
259         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
260     }
261
262     return flag;
263 }

```

```

263         String sql = "Delete from studentTask where taskname = '"+name+"'";
264
265         if(stmt.executeUpdate(sql) == 1)
266             flag = true;
267     } catch (SQLException ex) {
268         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
269     }
270
271     return flag;
272 }
273
274 public void disconnectDB()
275 {
276     try {
277         if (con != null)
278             con.close();
279     } catch (SQLException ex) {
280         Logger.getLogger(DBHelper.class.getName()).log(Level.SEVERE, null, ex);
281     }
282 }
283 }

```