```xml
<?xml version="1.0" encoding="UTF-8"?>
<form xmlns="http://www.intellij.com/uidesigner/form/" version="1" bind-to-class="Login">
  <grid id="27dc6" binding="panelMain" layout-manager="GridLayoutManager" row-count="1" column-count="7" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
    <margin top="20" left="20" bottom="20" right="20"/>
    <constraints>
      <xy x="20" y="20" width="695" height="349"/>
    </constraints>
    <properties/>
    <border type="none"/>
    <children>
      <grid id="30fc3" layout-manager="GridLayoutManager" row-count="7" column-count="8" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
        <margin top="20" left="20" bottom="20" right="20"/>
        <constraints>
          <grid row="0" column="0" row-span="1" col-span="6" vsize-policy="3" hsize-policy="3" anchor="0" fill="0" indent="0" use-parent-layout="false"/>
        </constraints>
        <properties/>
        <border type="none"/>
        <children>
          <vspacer id="20dd7">
            <constraints>
              <grid row="6" column="0" row-span="1" col-span="8" vsize-policy="6" hsize-policy="1" anchor="0" fill="2" indent="0" use-parent-layout="false"/>
            </constraints>
          </vspacer>
          <vspacer id="f6212">
            <constraints>
              <grid row="1" column="0" row-span="1" col-span="8" vsize-policy="6" hsize-policy="1" anchor="0" fill="2" indent="0" use-parent-layout="false"/>
            </constraints>
          </vspacer>
          <component id="33b8" class="javax.swing.JLabel">
            <constraints>
              <grid row="2" column="0" row-span="1" col-span="6" vsize-policy="0" hsize-policy="0" anchor="8" fill="0" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties>
```

```
34                <font size="16"/>
35                <text value="Email : "/>
36              </properties>
37            </component>
38            <component id="f2aa9" class="javax.swing.JTextField" binding="textFieldEmail">
39              <constraints>
40                <grid row="2" column="6" row-span="1" col-span="2" vsize-policy="0" hsize-policy="6" anchor="8"
    fill="1" indent="0" use-parent-layout="false">
41                  <preferred-size width="150" height="-1"/>
42                </grid>
43              </constraints>
44              <properties/>
45            </component>
46            <component id="a3be3" class="javax.swing.JLabel">
47              <constraints>
48                <grid row="3" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8"
    fill="0" indent="0" use-parent-layout="false"/>
49              </constraints>
50              <properties>
51                <font size="16"/>
52                <text value="Password : "/>
53              </properties>
54            </component>
55            <grid id="5faa6" layout-manager="GridLayoutManager" row-count="1" column-count="2" same-size-
    horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
56              <margin top="0" left="0" bottom="0" right="0"/>
57              <constraints>
58                <grid row="5" column="6" row-span="1" col-span="2" vsize-policy="3" hsize-policy="3" anchor="0"
    fill="3" indent="0" use-parent-layout="false"/>
59              </constraints>
60              <properties/>
61              <border type="none"/>
62              <children>
63                <component id="7312c" class="javax.swing.JButton" binding="buttonRegisterEmployer">
64                  <constraints>
65                    <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor
    ="0" fill="1" indent="0" use-parent-layout="false"/>
66                  </constraints>
67                  <properties>
```

```
68                  <enabled value="false"/>
69                  <font size="15"/>
70                  <text value="Register As Employer"/>
71                </properties>
72              </component>
73              <component id="42efc" class="javax.swing.JButton" binding="buttonRegisterCandidate">
74                <constraints>
75                  <grid row="0" column="1" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3"
   anchor="0" fill="1" indent="0" use-parent-layout="false"/>
76                </constraints>
77                <properties>
78                  <font size="15"/>
79                  <text value="Register As Candidate"/>
80                </properties>
81              </component>
82            </children>
83          </grid>
84          <grid id="11465" layout-manager="GridLayoutManager" row-count="1" column-count="1" same-size-
   horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
85            <margin top="0" left="0" bottom="0" right="0"/>
86            <constraints>
87              <grid row="5" column="0" row-span="1" col-span="6" vsize-policy="3" hsize-policy="3" anchor="0
   " fill="3" indent="0" use-parent-layout="false"/>
88            </constraints>
89            <properties/>
90            <border type="none"/>
91            <children>
92              <component id="ac66b" class="javax.swing.JButton" binding="buttonLogin">
93                <constraints>
94                  <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3"
   anchor="0" fill="1" indent="0" use-parent-layout="false"/>
95                </constraints>
96                <properties>
97                  <font size="15"/>
98                  <text value="Login"/>
99                </properties>
100             </component>
101           </children>
102         </grid>
```

```
103           <vspacer id="69a1f">
104             <constraints>
105               <grid row="4" column="5" row-span="1" col-span="1" vsize-policy="6" hsize-policy="1" anchor="0
    " fill="2" indent="0" use-parent-layout="false"/>
106             </constraints>
107           </vspacer>
108           <grid id="ccb52" layout-manager="GridLayoutManager" row-count="1" column-count="3" same-size-
    horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
109             <margin top="0" left="0" bottom="0" right="0"/>
110             <constraints>
111               <grid row="0" column="0" row-span="1" col-span="8" vsize-policy="3" hsize-policy="3" anchor="0
    " fill="3" indent="0" use-parent-layout="false"/>
112             </constraints>
113             <properties/>
114             <border type="none"/>
115             <children>
116               <component id="11dc0" class="javax.swing.JLabel" binding="textFieldTittle">
117                 <constraints>
118                   <grid row="0" column="1" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0"
    anchor="8" fill="0" indent="0" use-parent-layout="false"/>
119                 </constraints>
120                 <properties>
121                   <enabled value="true"/>
122                   <font size="25"/>
123                   <text value="User Login"/>
124                 </properties>
125               </component>
126               <hspacer id="7be22">
127                 <constraints>
128                   <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="1" hsize-policy="6"
    anchor="0" fill="1" indent="0" use-parent-layout="false"/>
129                 </constraints>
130               </hspacer>
131               <hspacer id="f2737">
132                 <constraints>
133                   <grid row="0" column="2" row-span="1" col-span="1" vsize-policy="1" hsize-policy="6"
    anchor="0" fill="1" indent="0" use-parent-layout="false"/>
134                 </constraints>
135               </hspacer>
```

```
            </children>
          </grid>
          <component id="7ceb" class="javax.swing.JPasswordField" binding="passwordFieldPassword">
            <constraints>
              <grid row="3" column="6" row-span="1" col-span="1" vsize-policy="0" hsize-policy="6" anchor="8
" fill="1" indent="0" use-parent-layout="false">
                <preferred-size width="150" height="-1"/>
              </grid>
            </constraints>
            <properties/>
          </component>
        </children>
      </grid>
    </children>
  </grid>
</form>
```

```java
import entities.Candidate;
import entities.Employee;
import entities.Employer;
import entities.User;
import org.json.JSONArray;
import org.json.JSONObject;

import javax.swing.*;
import java.io.File;
import java.io.IOException;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.ArrayList;
import java.util.Scanner;

public class Login extends JFrame{
    private JTextField textFieldEmail;
    private JButton buttonLogin;
    //private JTextField textFieldPassword;
    private JPanel panelMain;
    private JButton buttonRegisterEmployer;
    private JButton buttonRegisterCandidate;
    private JLabel textFieldTittle;
    private JPasswordField passwordFieldPassword;
    private User user;

    public Login(){
        this.add(panelMain);
        this.setSize(550,250);
        this.setTitle("Login");
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setResizable(false);

        File currentDirFile = new File("");

        String helper = "";

        try {
            helper = currentDirFile.getCanonicalPath() + "\\src\\main\\java\\images\\login.png";
```

```
40          } catch (IOException e) {
41              e.printStackTrace();
42          }
43
44
45          ImageIcon icon = new ImageIcon(helper);
46          textFieldTittle.setIcon(icon);
47
48          buttonLogin.addActionListener(e->{
49              checkUser(textFieldEmail.getText(), String.valueOf(passwordFieldPassword.getPassword()));
50          });
51
52          buttonRegisterCandidate.addActionListener(e -> {
53              this.setVisible(false);
54              new RegisterCandidate().setVisible(true);
55          });
56
57      }
58
59      public void checkUser(String email,String password) {
60
61          try {
62              if (checkIfUserIsCandidate(email,password)){
63                  this.setVisible(false);
64
65                  CandidateHomepage form = new CandidateHomepage(user);
66                  form.setVisible(true);
67              }
68
69              else if (checkIfUserIsEmployer(email,password))
70                  JOptionPane.showMessageDialog(this,"You are logging as employer");
71              else if (checkIfUserIsEmployee(email,password))
72                  JOptionPane.showMessageDialog(this,"You are logging as employee");
73              else
74                  JOptionPane.showMessageDialog(this,"Your email or password is incorrect");
75          } catch (Exception e) {
76              JOptionPane.showMessageDialog(this,e.getMessage());
77          }
78      }
```

```
 79
 80      public boolean checkIfUserIsCandidate(String email,String password) throws Exception{
 81          ArrayList<Candidate> candidates = new ArrayList<Candidate>();
 82
 83          URL candidateGetAll = new URL("http://localhost:8080/api/candidates/getAll");
 84          HttpURLConnection conn = (HttpURLConnection) candidateGetAll.openConnection();
 85          conn.setRequestMethod("GET");
 86          conn.connect();
 87          int responseCode = conn.getResponseCode();
 88
 89          if (responseCode != 200) {
 90              throw new RuntimeException("HttpResponseCode: " + responseCode);
 91          } else {
 92
 93              String inline = "";
 94              Scanner scanner = new Scanner(candidateGetAll.openStream());
 95
 96              //Write all the JSON data into a string using a scanner
 97              while (scanner.hasNext()) {
 98                  inline += scanner.nextLine();
 99              }
100
101              //Close the scanner
102              scanner.close();
103
104              String jsonString = inline ;
105              JSONObject obj = new JSONObject(jsonString);
106
107              JSONArray arr = obj.getJSONArray("data");
108              for (int i = 0; i < arr.length(); i++)
109              {
110                  Candidate candidate = new Candidate();
111                  candidate.id = arr.getJSONObject(i).getInt("id");
112                  candidate.firstName = arr.getJSONObject(i).getString("firstName");
113                  candidate.lastName = arr.getJSONObject(i).getString("lastName");
114                  candidate.identityId = arr.getJSONObject(i).getString("identityId");
115                  candidate.birthYear = arr.getJSONObject(i).getInt("birthYear");
116                  candidate.password = arr.getJSONObject(i).getString("password");
117                  candidate.email = arr.getJSONObject(i).getString("email");
```

```
118                 candidates.add(candidate);
119             }
120 
121             for(int i = 0 ; i < candidates.size(); i++){
122 
123                 if (candidates.get(i).email.equals(email) && candidates.get(i).password.equals(password)){
124                     user = candidates.get(i);
125                     return true;
126                 }
127 
128             }
129 
130             return false;
131         }
132     }
133 
134     public boolean checkIfUserIsEmployee(String email,String password) throws Exception{
135         ArrayList<Employee> employees = new ArrayList<Employee>();
136 
137         URL employeeGetAll = new URL("http://localhost:8080/api/employees/getAll");
138         HttpURLConnection conn = (HttpURLConnection) employeeGetAll.openConnection();
139         conn.setRequestMethod("GET");
140         conn.connect();
141         int responseCode = conn.getResponseCode();
142 
143         if (responseCode != 200) {
144             throw new RuntimeException("HttpResponseCode: " + responseCode);
145         } else {
146 
147             String inline = "";
148             Scanner scanner = new Scanner(employeeGetAll.openStream());
149 
150             //Write all the JSON data into a string using a scanner
151             while (scanner.hasNext()) {
152                 inline += scanner.nextLine();
153             }
154 
155             //Close the scanner
156             scanner.close();
```

```
157
158             String jsonString = inline ;
159             JSONObject obj = new JSONObject(jsonString);
160
161             JSONArray arr = obj.getJSONArray("data");
162             for (int i = 0; i < arr.length(); i++)
163             {
164                 Employee employee = new Employee();
165                 employee.firstName = arr.getJSONObject(i).getString("firstName");
166                 employee.lastName = arr.getJSONObject(i).getString("lastName");
167                 employee.password = arr.getJSONObject(i).getString("password");
168                 employee.email = arr.getJSONObject(i).getString("email");
169                 employees.add(employee);
170             }
171
172             for(int i = 0 ; i < employees.size(); i++){
173
174                 if (employees.get(i).email.equals(email) && employees.get(i).password.equals(password))
175                     return true;
176             }
177
178             return false;
179         }
180     }
181
182     public boolean checkIfUserIsEmployer(String email,String password) throws Exception{
183         ArrayList<Employer> employers = new ArrayList<Employer>();
184
185         URL employerGetAll = new URL("http://localhost:8080/api/employers/getAll");
186         HttpURLConnection conn = (HttpURLConnection) employerGetAll.openConnection();
187         conn.setRequestMethod("GET");
188         conn.connect();
189         int responseCode = conn.getResponseCode();
190
191         if (responseCode != 200) {
192             throw new RuntimeException("HttpResponseCode: " + responseCode);
193         } else {
194
195             String inline = "";
```

```
196             Scanner scanner = new Scanner(employerGetAll.openStream());
197
198             //Write all the JSON data into a string using a scanner
199             while (scanner.hasNext()) {
200                 inline += scanner.nextLine();
201             }
202
203             //Close the scanner
204             scanner.close();
205
206             String jsonString = inline ;
207             JSONObject obj = new JSONObject(jsonString);
208
209             JSONArray arr = obj.getJSONArray("data");
210             for (int i = 0; i < arr.length(); i++)
211             {
212                 Employer employer = new Employer();
213                 employer.companyName = arr.getJSONObject(i).getString("companyName");
214                 employer.webSite = arr.getJSONObject(i).getString("webSite");
215                 employer.password = arr.getJSONObject(i).getString("password");
216                 employer.email = arr.getJSONObject(i).getString("email");
217                 employers.add(employer);
218             }
219
220             for(int i = 0 ; i < employers.size(); i++){
221
222                 if (employers.get(i).email.equals(email) && employers.get(i).password.equals(password))
223                     return true;
224             }
225
226             return false;
227         }
228     }
229
230     public static void main(String[] args){
231         try {
232             UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
233         } catch (ClassNotFoundException e) {
234             e.printStackTrace();
```

```
235         } catch (InstantiationException e) {
236             e.printStackTrace();
237         } catch (IllegalAccessException e) {
238             e.printStackTrace();
239         } catch (UnsupportedLookAndFeelException e) {
240             e.printStackTrace();
241         }
242 
243         //Set UI Thread to another thread
244         SwingUtilities.invokeLater(() -> {
245             Login gui = new Login();
246             gui.setVisible(true);
247         });
248     }
249 
250 }
251 
252 
253 
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<form xmlns="http://www.intellij.com/uidesigner/form/" version="1" bind-to-class="CandidateCv">
  <grid id="27dc6" binding="panelMain" layout-manager="GridLayoutManager" row-count="3" column-count="2" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
    <margin top="20" left="20" bottom="20" right="20"/>
    <constraints>
      <xy x="20" y="20" width="1024" height="768"/>
    </constraints>
    <properties>
      <minimumSize width="1024" height="568"/>
      <preferredSize width="1024" height="768"/>
    </properties>
    <border type="none"/>
    <children>
      <grid id="3e928" binding="panelButtons" layout-manager="GridLayoutManager" row-count="1" column-count="4" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
        <margin top="0" left="0" bottom="0" right="0"/>
        <constraints>
          <grid row="1" column="0" row-span="1" col-span="2" vsize-policy="3" hsize-policy="3" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
        </constraints>
        <properties/>
        <border type="none"/>
        <children>
          <component id="8cdf4" class="javax.swing.JButton" binding="buttonCreateCv">
            <constraints>
              <grid row="0" column="1" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor="0" fill="1" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties>
              <text value="Create Cv"/>
            </properties>
          </component>
          <component id="87bc7" class="javax.swing.JButton" binding="buttonEditCv">
            <constraints>
              <grid row="0" column="2" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor="0" fill="1" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties>
```

```
35                <text value="Edit Selected Cv"/>
36              </properties>
37            </component>
38            <component id="a274a" class="javax.swing.JButton" binding="buttonDeleteCv">
39              <constraints>
40                <grid row="0" column="3" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor="0"
   fill="1" indent="0" use-parent-layout="false"/>
41              </constraints>
42              <properties>
43                <text value="Delete Selected Cv"/>
44              </properties>
45            </component>
46            <component id="7ba02" class="javax.swing.JButton" binding="buttonDeleteTextFields">
47              <constraints>
48                <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor="0"
   fill="1" indent="0" use-parent-layout="false"/>
49              </constraints>
50              <properties>
51                <text value="Refresh Page"/>
52              </properties>
53            </component>
54          </children>
55        </grid>
56        <grid id="17e09" binding="panelCvInformations" layout-manager="GridLayoutManager" row-count="9" column-
  count="7" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
57          <margin top="0" left="0" bottom="0" right="0"/>
58          <constraints>
59            <grid row="2" column="0" row-span="1" col-span="2" vsize-policy="3" hsize-policy="3" anchor="0"
  fill="3" indent="0" use-parent-layout="false"/>
60          </constraints>
61          <properties/>
62          <border type="none"/>
63          <children>
64            <component id="fbe17" class="javax.swing.JTextField" binding="textFieldCvId">
65              <constraints>
66                <grid row="0" column="1" row-span="1" col-span="6" vsize-policy="0" hsize-policy="6" anchor="8"
   fill="1" indent="0" use-parent-layout="false">
67                  <preferred-size width="150" height="-1"/>
68                </grid>
```

```
            </constraints>
            <properties>
              <editable value="false"/>
              <enabled value="false"/>
            </properties>
          </component>
          <component id="e40b6" class="javax.swing.JLabel" binding="labelCvId">
            <constraints>
              <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
" fill="0" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties>
              <text value="Id : "/>
            </properties>
          </component>
          <component id="48383" class="javax.swing.JTextField" binding="textFieldLinkedinAdress">
            <constraints>
              <grid row="1" column="1" row-span="1" col-span="6" vsize-policy="0" hsize-policy="6" anchor="8
" fill="1" indent="0" use-parent-layout="false">
                <preferred-size width="150" height="-1"/>
              </grid>
            </constraints>
            <properties/>
          </component>
          <component id="69225" class="javax.swing.JTextField" binding="textFieldGithubAdress">
            <constraints>
              <grid row="2" column="1" row-span="1" col-span="6" vsize-policy="0" hsize-policy="6" anchor="8
" fill="1" indent="0" use-parent-layout="false">
                <preferred-size width="150" height="-1"/>
              </grid>
            </constraints>
            <properties/>
          </component>
          <component id="9df2c" class="javax.swing.JLabel" binding="labelCvLinkedinAdress">
            <constraints>
              <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
" fill="0" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties>
```

```
104               <text value="Linked-in Adress :"/>
105             </properties>
106           </component>
107           <component id="d2cf0" class="javax.swing.JLabel" binding="labelCvGithubAdress">
108             <constraints>
109               <grid row="2" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
110             </constraints>
111             <properties>
112               <text value="Github Adress : "/>
113             </properties>
114           </component>
115           <component id="7327c" class="javax.swing.JLabel" binding="labelCvCoveringLetter">
116             <constraints>
117               <grid row="3" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
118             </constraints>
119             <properties>
120               <text value="Covering Letter : "/>
121             </properties>
122           </component>
123           <component id="a43bd" class="javax.swing.JLabel" binding="labelCvImageURL">
124             <constraints>
125               <grid row="4" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
126             </constraints>
127             <properties>
128               <text value="Cv Image URL : "/>
129             </properties>
130           </component>
131           <component id="54668" class="javax.swing.JTextField" binding="textFieldCoveringLetter">
132             <constraints>
133               <grid row="3" column="1" row-span="1" col-span="6" vsize-policy="0" hsize-policy="6" anchor="8
    " fill="1" indent="0" use-parent-layout="false">
134                 <preferred-size width="150" height="-1"/>
135               </grid>
136             </constraints>
137             <properties/>
138           </component>
```

```
139           <component id="15c29" class="javax.swing.JLabel">
140             <constraints>
141               <grid row="5" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
142             </constraints>
143             <properties>
144               <text value="Foreign Languages : "/>
145             </properties>
146           </component>
147           <component id="420fb" class="javax.swing.JLabel">
148             <constraints>
149               <grid row="6" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
150             </constraints>
151             <properties>
152               <text value="Knowledges : "/>
153             </properties>
154           </component>
155           <scrollpane id="8beed">
156             <constraints>
157               <grid row="6" column="1" row-span="1" col-span="6" vsize-policy="7" hsize-policy="7" anchor="0
    " fill="3" indent="0" use-parent-layout="false"/>
158             </constraints>
159             <properties/>
160             <border type="none"/>
161             <children>
162               <grid id="7a072" binding="panelProgrammingTechnologies" layout-manager="BorderLayout" hgap="0"
     vgap="0">
163                 <constraints/>
164                 <properties/>
165                 <border type="none"/>
166                 <children/>
167               </grid>
168             </children>
169           </scrollpane>
170           <scrollpane id="9de4a">
171             <constraints>
172               <grid row="5" column="1" row-span="1" col-span="6" vsize-policy="7" hsize-policy="7" anchor="0
    " fill="3" indent="0" use-parent-layout="false"/>
```

```
            </constraints>
            <properties/>
            <border type="none"/>
            <children>
              <grid id="16c5f" binding="panelForeignLanguages" layout-manager="FlowLayout" hgap="5" vgap="5" flow-align="1">
                <constraints/>
                <properties/>
                <border type="none"/>
                <children/>
              </grid>
            </children>
          </scrollpane>
          <component id="15d69" class="javax.swing.JLabel">
            <constraints>
              <grid row="7" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8" fill="0" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties>
              <text value="Schools : "/>
            </properties>
          </component>
          <component id="89fbd" class="javax.swing.JLabel">
            <constraints>
              <grid row="8" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8" fill="0" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties>
              <text value="Job Histories : "/>
            </properties>
          </component>
          <grid id="1bd2c" layout-manager="GridLayoutManager" row-count="1" column-count="2" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
            <margin top="0" left="0" bottom="0" right="0"/>
            <constraints>
              <grid row="7" column="1" row-span="1" col-span="6" vsize-policy="3" hsize-policy="3" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties/>
```

```
            <border type="none"/>
            <children>
              <component id="6be54" class="javax.swing.JButton" binding="buttonSchoolForSavedCvs">
                <constraints>
                  <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor="0" fill="1" indent="0" use-parent-layout="false"/>
                </constraints>
                <properties>
                  <text value="School For Saved Cvs"/>
                </properties>
              </component>
              <component id="6053d" class="javax.swing.JButton" binding="buttonSchoolForUnsavedCvs">
                <constraints>
                  <grid row="0" column="1" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor="0" fill="1" indent="0" use-parent-layout="false"/>
                </constraints>
                <properties>
                  <text value="School For Unsaved Cvs"/>
                </properties>
              </component>
            </children>
          </grid>
          <grid id="d5e8b" layout-manager="GridLayoutManager" row-count="1" column-count="2" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
            <margin top="0" left="0" bottom="0" right="0"/>
            <constraints>
              <grid row="8" column="1" row-span="1" col-span="6" vsize-policy="3" hsize-policy="3" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties/>
            <border type="none"/>
            <children>
              <component id="6c654" class="javax.swing.JButton" binding="buttonJobHistoryForSavedCvs">
                <constraints>
                  <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor="0" fill="1" indent="0" use-parent-layout="false"/>
                </constraints>
                <properties>
                  <text value="Job History For Saved Cvs"/>
```

```
241                 </properties>
242               </component>
243               <component id="862bf" class="javax.swing.JButton" binding="buttonJobHistoryForUnsavedCvs">
244                 <constraints>
245                   <grid row="0" column="1" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3"
    anchor="0" fill="1" indent="0" use-parent-layout="false"/>
246                 </constraints>
247                 <properties>
248                   <text value="Job History For Unsaved Cvs"/>
249                 </properties>
250               </component>
251             </children>
252           </grid>
253           <grid id="f03f0" layout-manager="GridLayoutManager" row-count="1" column-count="2" same-size-
    horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
254             <margin top="0" left="0" bottom="0" right="0"/>
255             <constraints>
256               <grid row="4" column="2" row-span="1" col-span="5" vsize-policy="3" hsize-policy="3" anchor="0
    " fill="3" indent="0" use-parent-layout="false"/>
257             </constraints>
258             <properties/>
259             <border type="none"/>
260             <children>
261               <component id="4888e" class="javax.swing.JButton" binding="buttonDeselectFile">
262                 <constraints>
263                   <grid row="0" column="1" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3"
    anchor="0" fill="1" indent="0" use-parent-layout="false"/>
264                 </constraints>
265                 <properties>
266                   <text value="Deselect File"/>
267                 </properties>
268               </component>
269               <component id="db795" class="javax.swing.JButton" binding="buttonBrowse">
270                 <constraints>
271                   <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3"
    anchor="0" fill="1" indent="0" use-parent-layout="false"/>
272                 </constraints>
273                 <properties>
274                   <text value="Browse"/>
```

```
275                 </properties>
276               </component>
277             </children>
278           </grid>
279           <grid id="7a4f" layout-manager="GridLayoutManager" row-count="1" column-count="1" same-size-
    horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
280             <margin top="0" left="0" bottom="0" right="0"/>
281             <constraints>
282               <grid row="4" column="1" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0
    " fill="3" indent="0" use-parent-layout="false"/>
283             </constraints>
284             <properties/>
285             <border type="none"/>
286             <children>
287               <component id="4872a" class="javax.swing.JTextField" binding="textFieldCvImageUrl">
288                 <constraints>
289                   <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="6"
    anchor="8" fill="1" indent="0" use-parent-layout="false">
290                     <preferred-size width="150" height="-1"/>
291                   </grid>
292                 </constraints>
293                 <properties/>
294               </component>
295             </children>
296           </grid>
297         </children>
298       </grid>
299       <grid id="b6340" binding="panelTable" layout-manager="GridLayoutManager" row-count="2" column-count="1
    " same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
300         <margin top="0" left="0" bottom="0" right="0"/>
301         <constraints>
302           <grid row="0" column="0" row-span="1" col-span="2" vsize-policy="3" hsize-policy="3" anchor="0"
    fill="3" indent="0" use-parent-layout="false"/>
303         </constraints>
304         <properties/>
305         <border type="none"/>
306         <children>
307           <scrollpane id="4dd25" binding="scrollPaneTable">
308             <constraints>
```

```
309               <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="7" hsize-policy="7" anchor="0
    " fill="3" indent="0" use-parent-layout="false"/>
310             </constraints>
311             <properties/>
312             <border type="none"/>
313             <children>
314               <component id="27adb" class="javax.swing.JTable" binding="tableCv">
315                 <constraints/>
316                 <properties>
317                   <minimumSize width="30" height="50"/>
318                   <preferredSize width="150" height="340"/>
319                   <requestFocusEnabled value="false"/>
320                 </properties>
321               </component>
322             </children>
323           </scrollpane>
324           <grid id="65cbf" layout-manager="GridLayoutManager" row-count="1" column-count="1" same-size-
    horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
325             <margin top="0" left="0" bottom="0" right="0"/>
326             <constraints>
327               <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0
    " fill="3" indent="0" use-parent-layout="false"/>
328             </constraints>
329             <properties/>
330             <border type="none"/>
331             <children/>
332           </grid>
333         </children>
334       </grid>
335     </children>
336   </grid>
337 </form>
338
```

```java
import entities.*;
import localData.CacheData;
import org.apache.http.HttpEntity;
import org.apache.http.client.methods.CloseableHttpResponse;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.entity.ContentType;
import org.apache.http.entity.mime.MultipartEntityBuilder;
import org.apache.http.entity.mime.content.FileBody;
import org.apache.http.entity.mime.content.StringBody;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.impl.client.HttpClients;
import org.apache.http.util.EntityUtils;
import org.json.JSONArray;
import org.json.JSONObject;

import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.*;
import java.io.*;
import java.net.HttpURLConnection;
import java.net.URL;
import java.nio.charset.StandardCharsets;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class CandidateCv extends JFrame{
    private JPanel panelMain;
    private JTable tableCv;
    private JButton buttonCreateCv;
    private JButton buttonEditCv;
    private JButton buttonDeleteCv;
    private JTextField textFieldCvId;
    private JTextField textFieldLinkedinAdress;
    private JTextField textFieldGithubAdress;
    private JTextField textFieldCvImageUrl;
    private JLabel labelCvId;
    private JLabel labelCvGithubAdress;
```

```
40      private JLabel labelCvLinkedinAdress;
41      private JLabel labelCvCoveringLetter;
42      private JLabel labelCvImageURL;
43      private JPanel panelButtons;
44      private JPanel panelCvInformations;
45      private JScrollPane scrollPaneTable;
46      private JPanel panelTable;
47      private JButton buttonBrowse;
48      private JButton buttonDeselectFile;
49      private JTextField textFieldCoveringLetter;
50      private JPanel panelForeignLanguages;
51      private JPanel panelProgrammingTechnologies;
52      private JButton buttonDeleteTextFields;
53      private JButton buttonSchoolForSavedCvs;
54      private JButton buttonJobHistoryForSavedCvs;
55      private JButton buttonSchoolForUnsavedCvs;
56      private JButton buttonJobHistoryForUnsavedCvs;
57      private JFileChooser fileChooser;
58
59      public User user;
60
61      private List<School> schools = new ArrayList<>();
62      private List<Cv> cvs = new ArrayList<>();
63      private List<String> foreignLanguagesString = new ArrayList<>();
64      private List<ForeignLanguage> foreignLanguages = new ArrayList<>();
65      private List<ForeignLanguage> selectedForeignLanguages = new ArrayList<>();
66
67      private List<String> programmingTechnologiesString = new ArrayList<>();
68      private List<ProgrammingTechnology> programmingTechnologies = new ArrayList<>();
69      private List<ProgrammingTechnology> selectedProgrammingTechnologies = new ArrayList<>();
70
71
72      public CandidateCv(User user){
73          this.user = user;
74
75          this.add(panelMain);
76          this.setSize(1280,720);
77          this.setDefaultCloseOperation(EXIT_ON_CLOSE);
78          this.setResizable(true);
```

```
 79
 80          fileChooser = new JFileChooser();
 81          fileChooser.setVisible(false);
 82
 83          try {
 84              getForeignLanguagesFromApi();
 85              getProgrammingTechnologiesFromApi();
 86          } catch (Exception e) {
 87              JOptionPane.showMessageDialog(this,e.getMessage());
 88          }
 89
 90          assignTable();
 91
 92
 93          tableCv.addMouseListener(new MouseAdapter() {
 94              @Override
 95              public void mouseClicked(MouseEvent e) {
 96
 97                  deleteTextFieldsAndAssignCheckbox();
 98
 99                  DefaultTableModel model = (DefaultTableModel) tableCv.getModel();
100
101                  int selectedRowIndex = tableCv.getSelectedRow();
102                  textFieldCvId.setText(model.getValueAt(selectedRowIndex,0).toString());
103                  textFieldGithubAdress.setText(model.getValueAt(selectedRowIndex,1).toString());
104                  textFieldLinkedinAdress.setText(model.getValueAt(selectedRowIndex,2).toString());
105                  textFieldCoveringLetter.setText(model.getValueAt(selectedRowIndex,3).toString());
106                  textFieldCvImageUrl.setText(model.getValueAt(selectedRowIndex,4).toString());
107
108                  assignCheckboxStatusByCvId(Integer.parseInt(textFieldCvId.getText()));
109              }
110          });
111
112          buttonSchoolForSavedCvs.addActionListener(e -> {
113              getSchoolsForSavedCvsPage();
114          });
115
116          buttonJobHistoryForSavedCvs.addActionListener(e -> {
117              refreshJobHistories();
```

```
118         });
119
120         buttonJobHistoryForUnsavedCvs.addActionListener(e -> getJobHistoriesForUnsavedCvsPage());
121
122         buttonDeleteCv.addActionListener(e -> {
123             try {
124                 deleteCv();
125             } catch (Exception ex) {
126                 JOptionPane.showMessageDialog(this,ex.getMessage());
127             }
128         });
129
130         buttonCreateCv.addActionListener(e -> {
131             try {
132                 setForeignLanguageToCv();
133                 setProgrammingTechnologiesToCv();
134
135
136                 getCheckboxStatusAndFillLists();
137                 int databaseAddedCvId = addCv();
138
139                 if (CacheData.savedSchools.size() != 0)
140                     addSchoolsToCv(databaseAddedCvId);
141
142                 if (CacheData.savedJobHistories.size() != 0)
143                     addJobHistoriesToCv(databaseAddedCvId);
144
145                 updateImageToCv(databaseAddedCvId);
146
147                 assignTable();
148                 deleteTextFieldsAndAssignCheckbox();
149
150             } catch (Exception ex) {
151                 JOptionPane.showMessageDialog(this,ex.getMessage());
152             }
153         });
154
155         buttonBrowse.addActionListener(e -> {
156             fileChooser.setVisible(true);
```

```
157             int value = fileChooser.showSaveDialog(this);
158
159             if (value == JFileChooser.APPROVE_OPTION) {
160                 File selectedFile = fileChooser.getSelectedFile();
161                 textFieldCvImageUrl.setText(selectedFile.getAbsolutePath());
162             }
163         });
164
165         buttonDeselectFile.addActionListener(e -> {
166             textFieldCvImageUrl.setText("");
167         });
168
169         buttonEditCv.addActionListener(e -> {
170             try {
171                 getCheckboxStatusAndFillLists();
172
173                 int databaseAddedCvId = updateCv();
174                 updateAndDeleteImageToCv(databaseAddedCvId);
175                 deleteTextFieldsAndAssignCheckbox();
176                 assignTable();
177             } catch (Exception ex) {
178                 JOptionPane.showMessageDialog(this,ex.getMessage());
179             }
180         });
181
182         buttonSchoolForUnsavedCvs.addActionListener(e -> getSchoolsForUnsavedCvsPage());
183
184         buttonJobHistoryForSavedCvs.addActionListener(e -> getJobHistoriesForSavedCvsPage());
185
186         buttonDeleteTextFields.addActionListener(e -> deleteTextFieldsAndAssignCheckbox());
187
188     }
189
190     private void addJobHistoriesToCv(int databaseAddedCvId) throws Exception{
191         for (var jobHistory : CacheData.savedJobHistories) {
192             JSONObject postJobHistory = new JSONObject();
193             postJobHistory.put("companyName", jobHistory.companyName);
194
195             postJobHistory.put("startedDate", jobHistory.startedDate);
```

```
196             postJobHistory.put("finishedDate", jobHistory.finishedDate);
197             postJobHistory.put("isFinished", jobHistory.isFinished);
198 
199             JSONObject cv = new JSONObject();
200             cv.put("id",databaseAddedCvId);
201             JSONObject addedCandidate = new JSONObject();
202             addedCandidate.put("id",user.id);
203             cv.put("candidate",addedCandidate);
204 
205             JSONObject position = new JSONObject();
206             position.put("id",jobHistory.jobPosition.id);
207 
208             postJobHistory.put("jobPosition",position);
209             postJobHistory.put("cv",cv);
210 
211             addJobHistoryToApi(postJobHistory);
212         }
213     }
214 
215     private void addJobHistoryToApi(JSONObject jobHistory) throws Exception {
216         URL url = new URL("http://localhost:8080/api/jobHistories/addJobHistory");
217         HttpURLConnection http = (HttpURLConnection)url.openConnection();
218         http.setRequestMethod("POST");
219         http.setDoOutput(true);
220         http.setRequestProperty("Content-Type", "application/json");
221 
222 
223         String data = jobHistory.toString();
224 
225         byte[] out = data.getBytes(StandardCharsets.UTF_8);
226 
227         OutputStream stream = http.getOutputStream();
228         stream.write(out);
229 
230         InputStream inputStream;
231 
232         if (http.getResponseCode() != 200)
233             inputStream = http.getErrorStream();
234         else
```

```
235             inputStream = http.getInputStream();
236 
237         String response = "";
238         String line;
239         BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
240 
241         while ((line = br.readLine()) != null) {
242             response += line;
243         }
244 
245         JSONObject returnObject = new JSONObject(response);
246 
247         if (returnObject.get("success").equals(true)){
248             http.disconnect();
249             deleteTextFieldsAndAssignCheckbox();
250         }
251 
252         else{
253             http.disconnect();
254             JOptionPane.showMessageDialog(this,returnObject.getString("message"),"Error",JOptionPane.
    ERROR_MESSAGE);
255         }
256     }
257 
258     private void getJobHistoriesForUnsavedCvsPage() {
259         CandidateJobHistoriesForUnsavedCvs form = new CandidateJobHistoriesForUnsavedCvs(user);
260         form.setVisible(true);
261     }
262 
263     private void getJobHistoriesForSavedCvsPage() {
264         if (!textFieldCvId.getText().equals("")){
265             CandidateJobHistoriesForSavedCvs form = new CandidateJobHistoriesForSavedCvs(user,textFieldCvId.
    getText());
266             form.setVisible(true);
267         }
268         else
269             JOptionPane.showMessageDialog(this,"You didn't select any cv","Error",JOptionPane.ERROR_MESSAGE
    );
270 
```

```
271     }
272
273     private void addSchoolsToCv(int cvId) throws Exception {
274
275         for (var school : CacheData.savedSchools) {
276             JSONObject postSchool = new JSONObject();
277             postSchool.put("name",school.name);
278             postSchool.put("department",school.department);
279
280             postSchool.put("startedDate",school.startedDate);
281             postSchool.put("graduatedDate",school.graduatedDate);
282             postSchool.put("graduated",school.isGraduated);
283
284             JSONObject cv = new JSONObject();
285             cv.put("id",cvId);
286             JSONObject addedCandidate = new JSONObject();
287             addedCandidate.put("id",user.id);
288             cv.put("candidate",addedCandidate);
289
290             postSchool.put("cv",cv);
291
292             addSchoolsToApi(postSchool);
293         }
294     }
295
296     private void addSchoolsToApi(JSONObject obj) throws Exception{
297         URL url = new URL("http://localhost:8080/api/schools/addSchool");
298         HttpURLConnection http = (HttpURLConnection)url.openConnection();
299         http.setRequestMethod("POST");
300         http.setDoOutput(true);
301         http.setRequestProperty("Content-Type", "application/json");
302
303
304         String data = obj.toString();
305
306         byte[] out = data.getBytes(StandardCharsets.UTF_8);
307
308         OutputStream stream = http.getOutputStream();
309         stream.write(out);
```

```
310
311         InputStream inputStream;
312
313         if (http.getResponseCode() != 200)
314             inputStream = http.getErrorStream();
315         else
316             inputStream = http.getInputStream();
317
318         String response = "";
319         String line;
320         BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
321
322         while ((line = br.readLine()) != null) {
323             response += line;
324         }
325
326         JSONObject returnObject = new JSONObject(response);
327
328         if (returnObject.get("success").equals(true)){
329             http.disconnect();
330             deleteTextFieldsAndAssignCheckbox();
331         }
332
333         else{
334             http.disconnect();
335             JOptionPane.showMessageDialog(this,returnObject.getString("message"),"Error",JOptionPane.
    ERROR_MESSAGE);
336         }
337     }
338
339     private void getSchoolsForSavedCvsPage(){
340         if (!textFieldCvId.getText().equals("")){
341             CandidateSchoolsForSavedCvs form = new CandidateSchoolsForSavedCvs(user,textFieldCvId.getText
    ());
342             form.setVisible(true);
343         }
344         else
345             JOptionPane.showMessageDialog(this,"You didn't select any cv","Error",JOptionPane.ERROR_MESSAGE
    );
```

```
346
347
348     }
349
350     private void getSchoolsForUnsavedCvsPage(){
351         CandidateSchoolsForUnsavedCvs candidateSchoolsForUnsavedCvs = new CandidateSchoolsForUnsavedCvs(user
    );
352         candidateSchoolsForUnsavedCvs.setVisible(true);
353     }
354
355     private void refreshJobHistories(){
356
357         /*panelJobHistories.removeAll();
358
359         panelJobHistories.setLayout(new GridLayout(LocalCache.getSavedJobHistories().size(),0));
360
361         for (var jobHistory :  LocalCache.getSavedJobHistories()) {
362
363             String label = jobHistory.companyName + " : " + jobHistory.jobPosition.name;
364
365             JCheckBox checkbox = new JCheckBox(label, null, false);
366
367             panelJobHistories.add(checkbox);
368         }
369
370         panelJobHistories.revalidate();
371         panelJobHistories.repaint();*/
372     }
373
374     private void updateAndDeleteImageToCv(int cvId) throws Exception{
375
376
377         if (!textFieldCvImageUrl.getText().startsWith("http") && !textFieldCvImageUrl.getText().equals("Boş"
    ) ){
378
379             if (!textFieldCvImageUrl.getText().equals("")){
380                 deleteCvImageFromCv(cvId);
381                 uploadCvImage(textFieldCvImageUrl.getText(),cvId);
382             }
```

```
383         }
384     }
385 
386     private void updateImageToCv(int cvId) throws Exception{
387 
388 
389         if (!textFieldCvImageUrl.getText().startsWith("http") && !textFieldCvImageUrl.getText().equals("Boş"
    ) ){
390 
391             if (!textFieldCvImageUrl.getText().equals("")){
392                 uploadCvImage(textFieldCvImageUrl.getText(),cvId);
393             }
394         }
395     }
396 
397     private int updateCv() throws Exception{
398         if (textFieldCvId.getText().equals(""))
399             throw new NullPointerException("Herhangi bir CV işaretlemediniz");
400 
401         int databaseAddedCvId = Integer.parseInt(textFieldCvId.getText());
402 
403         JSONObject obj = new JSONObject();
404         obj.put("id", textFieldCvId.getText());
405         obj.put("githubAdress", textFieldGithubAdress.getText());
406         obj.put("coveringLetter", textFieldCoveringLetter.getText());
407         obj.put("linkedinAdress", textFieldLinkedinAdress.getText());
408         JSONObject candidate = new JSONObject();
409         candidate.put("id", user.id);
410         obj.put("candidate",candidate);
411 
412         JSONArray postForeignLanguages = new JSONArray();
413         JSONArray postProgrammingTechnologies = new JSONArray();
414 
415         for (int i = 0 ; i < this.selectedForeignLanguages.size(); i++) {
416 
417             JSONObject postForeignLanguage = new JSONObject();
418             postForeignLanguage.put("id",selectedForeignLanguages.get(i).id);
419             postForeignLanguage.put("name",selectedForeignLanguages.get(i).name);
420 
```

```
421             JSONObject candidateForeignLanguages = new JSONObject();
422             candidateForeignLanguages.put("foreignLanguage",postForeignLanguage);
423             candidateForeignLanguages.put("level",2);
424 
425             postForeignLanguages.put(candidateForeignLanguages);
426         }
427 
428         obj.put("candidateForeignLanguages",postForeignLanguages);
429 
430 
431         for (int i = 0 ; i < this.selectedProgrammingTechnologies.size(); i++) {
432 
433             JSONObject postProgrammingTechnology = new JSONObject();
434             postProgrammingTechnology.put("id",selectedProgrammingTechnologies.get(i).id);
435             postProgrammingTechnology.put("name",selectedProgrammingTechnologies.get(i).name);
436 
437             JSONObject candidateProgrammingTechnologies = new JSONObject();
438             candidateProgrammingTechnologies.put("programmingTechnology",postProgrammingTechnology);
439 
440             postProgrammingTechnologies.put(candidateProgrammingTechnologies);
441         }
442 
443         obj.put("candidateKnowledges",postProgrammingTechnologies);
444 
445         URL url = new URL("http://localhost:8080/api/cvs/updateCv");
446         HttpURLConnection http = (HttpURLConnection)url.openConnection();
447         http.setRequestMethod("PUT");
448         http.setDoOutput(true);
449         http.setRequestProperty("Content-Type", "application/json");
450 
451 
452         String data = obj.toString();
453 
454         byte[] out = data.getBytes(StandardCharsets.UTF_8);
455 
456         OutputStream stream = http.getOutputStream();
457         stream.write(out);
458 
459         InputStream inputStream;
```

```
460
461         if (http.getResponseCode() != 200)
462             inputStream = http.getErrorStream();
463         else
464             inputStream = http.getInputStream();
465
466         String response = "";
467         String line;
468         BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
469
470         while ((line = br.readLine()) != null) {
471             response += line;
472         }
473
474         JSONObject returnObject = new JSONObject(response);
475
476         if (returnObject.get("success").equals(true)){
477             JOptionPane.showMessageDialog(this,returnObject.get("message"));
478             http.disconnect();
479             assignTable();
480             programmingTechnologiesString.clear();
481             foreignLanguagesString.clear();
482             selectedForeignLanguages.clear();
483             selectedProgrammingTechnologies.clear();
484             return databaseAddedCvId;
485         }
486
487         else{
488             http.disconnect();
489             JOptionPane.showMessageDialog(this,returnObject.get("message"));
490             return 0;
491         }
492
493     }
494
495     private void deleteCvImageFromCv(int cvId) throws Exception{
496
497         URL url = new URL("http://localhost:8080/api/cvs/deleteImageToCv?cvId=" + cvId);
498         HttpURLConnection httpCon = (HttpURLConnection) url.openConnection();
```

```
499         httpCon.setDoOutput(true);
500         httpCon.setRequestMethod("DELETE");
501         httpCon.connect();
502 
503         InputStream inputStream;
504 
505         if (httpCon.getResponseCode() != 200)
506             inputStream = httpCon.getErrorStream();
507         else
508             inputStream = httpCon.getInputStream();
509 
510         String response = "";
511         String line;
512         BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
513 
514         while ((line = br.readLine()) != null) {
515             response += line;
516         }
517 
518         JSONObject returnObject = new JSONObject(response);
519 
520         if (!returnObject.getBoolean("success")){
521             JOptionPane.showMessageDialog(this,returnObject.getString("message"),"Error",JOptionPane.
    ERROR_MESSAGE);
522         }
523 
524     }
525 
526     private void uploadCvImage(String path,int cvId) throws Exception{
527 
528             CloseableHttpClient httpclient = HttpClients.createDefault();
529             HttpPost httppost = new HttpPost("http://localhost:8080/api/cvs/uploadImageToCv");
530 
531             FileBody bin = new FileBody(new File(path));
532 
533             StringBody cvIdBody = new StringBody(String.valueOf(cvId), ContentType.TEXT_PLAIN);
534 
535             HttpEntity reqEntity = MultipartEntityBuilder.create()
536                     .addPart("multipartFile", bin)
```

```
537                     .addPart("cvId", cvIdBody)
538                     .build();
539
540
541             httppost.setEntity(reqEntity);
542
543             CloseableHttpResponse response = httpclient.execute(httppost);
544
545             HttpEntity resEntity = response.getEntity();
546
547             String responseString = EntityUtils.toString(resEntity, "UTF-8");
548
549             JSONObject obj = new JSONObject(responseString);
550             if (!obj.getBoolean("success")){
551                 JOptionPane.showMessageDialog(this,obj.getString("message"));
552             }
553
554             response.close();
555             httpclient.close();
556
557     }
558
559     private void getCheckboxStatusAndFillLists(){
560
561         programmingTechnologiesString.clear();
562         foreignLanguagesString.clear();
563         selectedForeignLanguages.clear();
564         selectedProgrammingTechnologies.clear();
565
566         var programmingTechnologiesComponents = panelProgrammingTechnologies.getComponents();
567
568         for (Component component : programmingTechnologiesComponents){
569             JCheckBox checkbox = (JCheckBox) component;
570
571             if(checkbox.isSelected()){
572                 programmingTechnologiesString.add(checkbox.getText());
573             }
574         }
575
```

```
576
577         var foreignLanguagesComponents = panelForeignLanguages.getComponents();
578
579         for (Component component : foreignLanguagesComponents){
580             JCheckBox checkbox = (JCheckBox) component;
581
582             if(checkbox.isSelected()){
583                 foreignLanguagesString.add(checkbox.getText());
584             }
585         }
586
587
588         for (String foreignLanguageString : foreignLanguagesString){
589             for (ForeignLanguage foreignLanguage : foreignLanguages){
590                 if (foreignLanguage.name.equals(foreignLanguageString)){
591                     selectedForeignLanguages.add(foreignLanguage);
592                 }
593             }
594         }
595
596         for (String programmingTechnologyString : programmingTechnologiesString){
597             for (ProgrammingTechnology programmingTechnology : programmingTechnologies){
598                 if (programmingTechnology.name.equals(programmingTechnologyString)){
599                     selectedProgrammingTechnologies.add(programmingTechnology);
600                 }
601             }
602         }
603
604     }
605
606     private void assignCheckboxStatusByCvId(int cvId){
607         var programmingTechnologiesComponents = panelProgrammingTechnologies.getComponents();
608         var foundCv = new Cv();
609
610         for (var cv : cvs){
611             if (cv.id == cvId){
612                 foundCv = cv;
613                 break;
614             }
```

```
615         }
616
617         for (Component component : programmingTechnologiesComponents){
618             JCheckBox checkbox = (JCheckBox) component;
619
620             for(var tech : foundCv.programmingTechnologies){
621                 if(tech.name.equals(checkbox.getText())){
622                     checkbox.setSelected(true);
623                 }
624             }
625         }
626
627         var foreignLanguagesComponents = panelForeignLanguages.getComponents();
628         for (Component component : foreignLanguagesComponents){
629             JCheckBox checkbox = (JCheckBox) component;
630
631             for(var tech : foundCv.foreignLanguages){
632                 if(tech.name.equals(checkbox.getText())){
633                     checkbox.setSelected(true);
634                 }
635             }
636         }
637     }
638
639     private void setProgrammingTechnologiesToCv() {
640
641         for (int i = 0 ; i < programmingTechnologiesString.size(); i++){
642             for (int j = 0 ; j < programmingTechnologies.size(); j++){
643                 if (programmingTechnologiesString.get(i) == programmingTechnologies.get(j).name){
644                     selectedProgrammingTechnologies.add(programmingTechnologies.get(j));
645                     break;
646                 }
647             }
648         }
649
650     }
651
652     private void setForeignLanguageToCv(){
653         for (int i = 0 ; i < foreignLanguagesString.size(); i++){
```

```
654             for (int j = 0 ; j < foreignLanguages.size(); j++){
655                 if (foreignLanguagesString.get(i) == foreignLanguages.get(j).name){
656                     selectedForeignLanguages.add(foreignLanguages.get(j));
657                     break;
658                 }
659             }
660         }
661     }
662
663     private void deleteCv() throws Exception{
664
665         if (textFieldCvId.getText().equals(""))
666             JOptionPane.showMessageDialog(this,"You didn't select any CV");
667         else {
668
669             URL url = new URL("http://localhost:8080/api/cvs/deleteCv?cvId=" + textFieldCvId.getText());
670             HttpURLConnection httpCon = (HttpURLConnection) url.openConnection();
671             httpCon.setDoOutput(true);
672             httpCon.setRequestMethod("DELETE");
673             httpCon.connect();
674
675             InputStream inputStream;
676
677             if (httpCon.getResponseCode() != 200)
678                 inputStream = httpCon.getErrorStream();
679             else
680                 inputStream = httpCon.getInputStream();
681
682             String response = "";
683             String line;
684             BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
685
686             while ((line = br.readLine()) != null) {
687                 response += line;
688             }
689
690             JSONObject returnObject = new JSONObject(response);
691
692             if (returnObject.getBoolean("success")) {
```

```
693                 JOptionPane.showMessageDialog(this, returnObject.getString("message"), "Information",
    JOptionPane.INFORMATION_MESSAGE);
694                 assignTable();
695                 deleteTextFieldsAndAssignCheckbox();
696             } else
697                 JOptionPane.showMessageDialog(this, returnObject.getString("message"), "Error", JOptionPane.
    ERROR_MESSAGE);
698         }
699     }
700
701     private void getForeignLanguagesFromApi() throws Exception{
702
703         URL getForeignLanguages = new URL("http://localhost:8080/api/foreignLanguages/getAll");
704         HttpURLConnection conn = (HttpURLConnection) getForeignLanguages.openConnection();
705         conn.setRequestMethod("GET");
706         conn.connect();
707         int responseCode = conn.getResponseCode();
708
709         if (responseCode != 200) {
710             throw new RuntimeException("HttpResponseCode: " + responseCode);
711         } else {
712
713             String inline = "";
714             Scanner scanner = new Scanner(getForeignLanguages.openStream());
715
716             //Write all the JSON data into a string using a scanner
717             while (scanner.hasNext()) {
718                 inline += scanner.nextLine();
719             }
720
721             //Close the scanner
722             scanner.close();
723
724             String jsonString = inline;
725             JSONObject obj = new JSONObject(jsonString);
726
727             JSONArray arr = obj.getJSONArray("data");
728
729             panelForeignLanguages.setLayout(new GridLayout(arr.length(),0));
```

```
730
731             for (int i = 0; i < arr.length(); i++) {
732
733                 JSONObject foreignLanguage = arr.getJSONObject(i);
734
735                 ForeignLanguage f = new ForeignLanguage();
736                 f.id = foreignLanguage.getInt("id");
737                 f.name = foreignLanguage.getString("name");
738
739                 foreignLanguages.add(f);
740
741                 String label = foreignLanguage.getString("name");
742
743                 JCheckBox checkbox = new JCheckBox(label,null,false);
744
745                 checkbox.addActionListener(e -> {
746                     if (checkbox.isSelected()){
747                        foreignLanguagesString.add(checkbox.getText());
748                     }else{
749                         foreignLanguagesString.remove(checkbox.getText());
750                     }
751                 });
752
753                 panelForeignLanguages.add(checkbox);
754             }
755
756         }
757
758     }
759
760     private void getProgrammingTechnologiesFromApi() throws Exception{
761
762         URL getKnowledges = new URL("http://localhost:8080/api/programmingTechnologies/getAll");
763         HttpURLConnection conn = (HttpURLConnection) getKnowledges.openConnection();
764         conn.setRequestMethod("GET");
765         conn.connect();
766         int responseCode = conn.getResponseCode();
767
768         if (responseCode != 200) {
```

```
769             throw new RuntimeException("HttpResponseCode: " + responseCode);
770         } else {
771 
772             String inline = "";
773             Scanner scanner = new Scanner(getKnowledges.openStream());
774 
775             //Write all the JSON data into a string using a scanner
776             while (scanner.hasNext()) {
777                 inline += scanner.nextLine();
778             }
779 
780             //Close the scanner
781             scanner.close();
782 
783             String jsonString = inline;
784             JSONObject obj = new JSONObject(jsonString);
785 
786             JSONArray arr = obj.getJSONArray("data");
787 
788             panelProgrammingTechnologies.setLayout(new GridLayout(arr.length(), 1));
789 
790             for (int i = 0; i < arr.length(); i++) {
791 
792                 JSONObject knowledge = arr.getJSONObject(i);
793 
794                 ProgrammingTechnology f = new ProgrammingTechnology();
795                 f.id = knowledge.getInt("id");
796                 f.name = knowledge.getString("name");
797 
798                 programmingTechnologies.add(f);
799 
800                 String label = knowledge.getString("name");
801 
802                 JCheckBox checkbox = new JCheckBox(label,null,false);
803 
804                 checkbox.addActionListener(e -> {
805                     if (checkbox.isSelected()){
806                         programmingTechnologiesString.add(checkbox.getText());
807                     }else{
```

```
808                         programmingTechnologiesString.remove(checkbox.getText());
809                     }
810                 });
811 
812                 panelProgrammingTechnologies.add(checkbox);
813             }
814 
815 
816 
817         }
818 
819     }
820 
821     private int addCv() throws Exception{
822 
823         JSONObject obj = new JSONObject();
824         obj.put("githubAdress", textFieldGithubAdress.getText());
825         obj.put("coveringLetter", textFieldCoveringLetter.getText());
826         obj.put("linkedinAdress", textFieldLinkedinAdress.getText());
827         JSONObject candidate = new JSONObject();
828         candidate.put("id", user.id);
829         obj.put("candidate",candidate);
830 
831         JSONArray postForeignLanguages = new JSONArray();
832         JSONArray postProgrammingTechnologies = new JSONArray();
833 
834         for (int i = 0 ; i < this.selectedForeignLanguages.size(); i++) {
835 
836             JSONObject postForeignLanguage = new JSONObject();
837             postForeignLanguage.put("id",selectedForeignLanguages.get(i).id);
838             postForeignLanguage.put("name",selectedForeignLanguages.get(i).name);
839 
840             JSONObject candidateForeignLanguages = new JSONObject();
841             candidateForeignLanguages.put("foreignLanguage",postForeignLanguage);
842             candidateForeignLanguages.put("level",2);
843 
844             postForeignLanguages.put(candidateForeignLanguages);
845         }
846 
```

```
847         obj.put("candidateForeignLanguages",postForeignLanguages);
848
849
850         for (int i = 0 ; i < this.selectedProgrammingTechnologies.size(); i++) {
851
852             JSONObject postProgrammingTechnology = new JSONObject();
853             postProgrammingTechnology.put("id",selectedProgrammingTechnologies.get(i).id);
854             postProgrammingTechnology.put("name",selectedProgrammingTechnologies.get(i).name);
855
856             JSONObject candidateProgrammingTechnologies = new JSONObject();
857             candidateProgrammingTechnologies.put("programmingTechnology",postProgrammingTechnology);
858
859             postProgrammingTechnologies.put(candidateProgrammingTechnologies);
860         }
861
862         obj.put("candidateKnowledges",postProgrammingTechnologies);
863
864
865         URL url = new URL("http://localhost:8080/api/cvs/addCv");
866         HttpURLConnection http = (HttpURLConnection)url.openConnection();
867         http.setRequestMethod("POST");
868         http.setDoOutput(true);
869         http.setRequestProperty("Content-Type", "application/json");
870
871
872         String data = obj.toString();
873
874         byte[] out = data.getBytes(StandardCharsets.UTF_8);
875
876         OutputStream stream = http.getOutputStream();
877         stream.write(out);
878
879         InputStream inputStream;
880
881         if (http.getResponseCode() != 200)
882             inputStream = http.getErrorStream();
883         else
884             inputStream = http.getInputStream();
885
```

```
886         String response = "";
887         String line;
888         BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
889 
890         while ((line = br.readLine()) != null) {
891             response += line;
892         }
893 
894         JSONObject returnObject = new JSONObject(response);
895 
896         if (returnObject.get("success").equals(true)){
897             JOptionPane.showMessageDialog(this,returnObject.get("message"));
898             http.disconnect();
899             return returnObject.getJSONObject("data").getInt("id");
900         }
901 
902         else{
903             http.disconnect();
904             JOptionPane.showMessageDialog(this,returnObject.get("message"));
905             return 0;
906         }
907 
908     }
909 
910 
911 
912     public void deleteTextFieldsAndAssignCheckbox(){
913         textFieldCvId.setText("");
914         textFieldCoveringLetter.setText("");
915         textFieldCvImageUrl.setText("");
916         textFieldGithubAdress.setText("");
917         textFieldLinkedinAdress.setText("");
918 
919         programmingTechnologiesString.clear();
920         foreignLanguagesString.clear();
921         selectedForeignLanguages.clear();
922         selectedProgrammingTechnologies.clear();
923 
924         var programmingTechnologiesComponents = panelProgrammingTechnologies.getComponents();
```

```
925
926         for (Component component : programmingTechnologiesComponents){
927             JCheckBox checkbox = (JCheckBox) component;
928             checkbox.setSelected(false);
929         }
930
931         var foreignLanguagesComponents = panelForeignLanguages.getComponents();
932
933         for (Component component : foreignLanguagesComponents){
934             JCheckBox checkbox = (JCheckBox) component;
935             checkbox.setSelected(false);
936         }
937     }
938
939     private void assignTable(){
940
941         String[] columns = new String[] {
942                 "Id","Github Address", "LinkedIn Address", "Covering Letter" , "Image Link"
943         };
944
945         try {
946             cvs = getCvFromApi();
947         } catch (Exception e) {
948             JOptionPane.showMessageDialog(this,e.getMessage(),"Error",JOptionPane.ERROR_MESSAGE);
949         }
950
951         //actual data for the table in a 2d array
952         Object[][] data = new Object[cvs.size()][5];
953
954         for(var i = 0; i < cvs.size(); i++){
955             data[i][0] = cvs.get(i).id;
956             data[i][1] = cvs.get(i).githubAdress;
957             data[i][2] = cvs.get(i).linkedinAdress;
958             data[i][3] = cvs.get(i).coveringLetter;
959             data[i][4] = cvs.get(i).cvImage.url;
960         }
961
962         DefaultTableModel model = new DefaultTableModel(data,columns){
963             @Override
```

```
964             public boolean isCellEditable(int row, int column) {
965                 return false;
966             }
967         };
968 
969         tableCv.setModel(model);
970     }
971 
972     private List<Cv> getCvFromApi() throws Exception{
973         ArrayList<Cv> cvs = new ArrayList<>();
974 
975         URL getCvsByCandidateId = new URL("http://localhost:8080/api/cvs/getAllByCandidateId?candidateId="+
    user.id);
976         HttpURLConnection conn = (HttpURLConnection) getCvsByCandidateId.openConnection();
977         conn.setRequestMethod("GET");
978         conn.connect();
979         int responseCode = conn.getResponseCode();
980 
981         if (responseCode != 200) {
982             throw new RuntimeException("HttpResponseCode: " + responseCode);
983         } else {
984 
985             String inline = "";
986             Scanner scanner = new Scanner(getCvsByCandidateId.openStream());
987 
988             //Write all the JSON data into a string using a scanner
989             while (scanner.hasNext()) {
990                 inline += scanner.nextLine();
991             }
992 
993             //Close the scanner
994             scanner.close();
995 
996             String jsonString = inline ;
997             JSONObject obj = new JSONObject(jsonString);
998 
999             JSONArray arr = obj.getJSONArray("data");
1000            for (int i = 0; i < arr.length(); i++)
1001            {
```

```
1002                    Cv cv = new Cv();
1003                    cv.linkedinAdress = arr.getJSONObject(i).getString("linkedinAdress");
1004                    cv.coveringLetter = arr.getJSONObject(i).getString("coveringLetter");
1005                    cv.githubAdress = arr.getJSONObject(i).getString("githubAdress");
1006
1007                    if(arr.getJSONObject(i).has("image") && !arr.getJSONObject(i).isNull("image")){
1008                        cv.cvImage = new CvImage(
1009                                arr.getJSONObject(i).getJSONObject("image").getInt("id"),
1010                                arr.getJSONObject(i).getJSONObject("image").getString("url"),
1011                                arr.getJSONObject(i).getJSONObject("image").getString("publicImageId"),
1012                                arr.getJSONObject(i).getJSONObject("image").getString("createdDate"));
1013                    }
1014                    else{
1015                        cv.cvImage = new CvImage(0,"Boş","Boş","Boş");
1016                    }
1017
1018
1019                    var knowledges = arr.getJSONObject(i).getJSONArray("candidateKnowledges");
1020                    var foreignLanguages = arr.getJSONObject(i).getJSONArray("candidateForeignLanguages");
1021
1022                    for (int x = 0; x < knowledges.length(); x++){
1023                        ProgrammingTechnology pt = new ProgrammingTechnology();
1024                        pt.name = knowledges.getJSONObject(x).getJSONObject("programmingTechnology").getString(
      "name");
1025                        pt.id = knowledges.getJSONObject(x).getJSONObject("programmingTechnology").getInt("id"
      );
1026
1027                        cv.programmingTechnologies.add(pt);
1028                    }
1029
1030                    for (int x = 0; x < foreignLanguages.length(); x++){
1031                        ForeignLanguage fl = new ForeignLanguage();
1032                        fl.name = foreignLanguages.getJSONObject(x).getJSONObject("foreignLanguage").getString(
      "name");
1033                        fl.id = foreignLanguages.getJSONObject(x).getJSONObject("foreignLanguage").getInt("id"
      );
1034
1035                        cv.foreignLanguages.add(fl);
1036                    }
```

```
1037
1038
1039                 cv.id = arr.getJSONObject(i).getInt("id");
1040
1041                 cvs.add(cv);
1042             }
1043
1044             return cvs;
1045         }
1046     }
1047
1048     public void main(String[] args){
1049         try {
1050             UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
1051         } catch (ClassNotFoundException e) {
1052             e.printStackTrace();
1053         } catch (InstantiationException e) {
1054             e.printStackTrace();
1055         } catch (IllegalAccessException e) {
1056             e.printStackTrace();
1057         } catch (UnsupportedLookAndFeelException e) {
1058             e.printStackTrace();
1059         }
1060
1061         //Set UI Thread to another thread
1062         SwingUtilities.invokeLater(() -> {
1063             CandidateCv gui = new CandidateCv(user);
1064             gui.setVisible(true);
1065         });
1066
1067     }
1068
1069 }
1070
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<form xmlns="http://www.intellij.com/uidesigner/form/" version="1" bind-to-class="CandidateProfile">
  <grid id="27dc6" binding="panelMain" layout-manager="GridLayoutManager" row-count="2" column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
    <margin top="0" left="0" bottom="0" right="0"/>
    <constraints>
      <xy x="20" y="20" width="500" height="400"/>
    </constraints>
    <properties/>
    <border type="none"/>
    <children>
      <scrollpane id="76245">
        <constraints>
          <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="7" hsize-policy="7" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
        </constraints>
        <properties/>
        <border type="none"/>
        <children>
          <grid id="83b44" binding="panelOptions" layout-manager="GridLayoutManager" row-count="1" column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
            <margin top="0" left="0" bottom="0" right="0"/>
            <constraints/>
            <properties/>
            <border type="none"/>
            <children/>
          </grid>
        </children>
      </scrollpane>
      <grid id="2ae7a" layout-manager="GridLayoutManager" row-count="1" column-count="4" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
        <margin top="0" left="0" bottom="0" right="0"/>
        <constraints>
          <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
        </constraints>
        <properties/>
        <border type="none"/>
        <children>
```

```
35            <component id="5668d" class="javax.swing.JButton" binding="buttonCv">
36              <constraints>
37                <grid row="0" column="1" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor="0"
    fill="1" indent="0" use-parent-layout="false"/>
38              </constraints>
39              <properties>
40                <text value="CV Processes"/>
41              </properties>
42            </component>
43            <hspacer id="75687">
44              <constraints>
45                <grid row="0" column="3" row-span="1" col-span="1" vsize-policy="1" hsize-policy="6" anchor="0"
    fill="1" indent="0" use-parent-layout="false"/>
46              </constraints>
47            </hspacer>
48            <hspacer id="82fb7">
49              <constraints>
50                <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="1" hsize-policy="6" anchor="0"
    fill="1" indent="0" use-parent-layout="false"/>
51              </constraints>
52            </hspacer>
53            <component id="6550e" class="javax.swing.JButton" binding="buttonJobAdvert">
54              <constraints>
55                <grid row="0" column="2" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor="0"
    fill="1" indent="0" use-parent-layout="false"/>
56              </constraints>
57              <properties>
58                <enabled value="false"/>
59                <text value="See Job Adverts"/>
60              </properties>
61            </component>
62          </children>
63        </grid>
64      </children>
65    </grid>
66  </form>
67
```

```java
import entities.User;

import javax.swing.*;
import java.awt.*;

public class CandidateProfile extends JFrame {

    public User user;
    private JPanel panelMain;
    private JButton buttonCv;
    private JPanel panelOptions;
    private JButton buttonJobAdvert;

    public CandidateProfile(User user){
        this.user = user;
        this.add(panelMain);
        this.setSize(1280,720);

        buttonCv.addActionListener(e -> {
            panelOptions.removeAll();
            panelOptions.revalidate();
            panelOptions.repaint();
            panelOptions.setLayout(new GridLayout(1,0));
            panelOptions.add(new CandidateCv(user).getContentPane());
            panelOptions.repaint();
            panelOptions.revalidate();
        });

        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setResizable(false);
    }

    public void main(String[] args){
        try {
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        } catch (InstantiationException e) {
            e.printStackTrace();
```

```
40          } catch (IllegalAccessException e) {
41              e.printStackTrace();
42          } catch (UnsupportedLookAndFeelException e) {
43              e.printStackTrace();
44          }
45
46          //Set UI Thread to another thread
47          SwingUtilities.invokeLater(() -> {
48              CandidateProfile gui = new CandidateProfile(user);
49              gui.setVisible(true);
50          });
51
52      }
53  }
54
```

```
<?xml version="1.0" encoding="UTF-8"?>
<form xmlns="http://www.intellij.com/uidesigner/form/" version="1" bind-to-class="CandidateHomepage">
  <grid id="27dc6" binding="panelMain" layout-manager="GridLayoutManager" row-count="2" column-count="2" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
    <margin top="0" left="0" bottom="0" right="0"/>
    <constraints>
      <xy x="20" y="20" width="1280" height="768"/>
    </constraints>
    <properties>
      <minimumSize width="1280" height="720"/>
    </properties>
    <border type="none"/>
    <children>
      <grid id="7fc41" binding="panelSidebar" layout-manager="GridLayoutManager" row-count="4" column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
        <margin top="0" left="0" bottom="0" right="0"/>
        <constraints>
          <grid row="0" column="0" row-span="2" col-span="1" vsize-policy="3" hsize-policy="1" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
        </constraints>
        <properties>
          <autoscrolls value="true"/>
          <background color="-4473925"/>
          <foreground color="-1025"/>
        </properties>
        <border type="none"/>
        <children>
          <vspacer id="1759a">
            <constraints>
              <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="6" hsize-policy="1" anchor="0" fill="2" indent="0" use-parent-layout="false"/>
            </constraints>
          </vspacer>
          <vspacer id="745d9">
            <constraints>
              <grid row="3" column="0" row-span="1" col-span="1" vsize-policy="6" hsize-policy="1" anchor="0" fill="2" indent="0" use-parent-layout="false"/>
            </constraints>
          </vspacer>
```

```
35            <component id="bef37" class="javax.swing.JButton" binding="buttonSignOff">
36              <constraints>
37                <grid row="2" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor="0"
    fill="1" indent="0" use-parent-layout="false"/>
38              </constraints>
39              <properties>
40                <font size="15" style="1"/>
41                <text value="Sign Off"/>
42              </properties>
43            </component>
44            <component id="8aa8e" class="javax.swing.JButton" binding="buttonProfile">
45              <constraints>
46                <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor="0"
    fill="1" indent="0" use-parent-layout="false"/>
47              </constraints>
48              <properties>
49                <actionCommand value=""/>
50                <font size="15" style="1"/>
51                <text value="Profile"/>
52              </properties>
53            </component>
54          </children>
55        </grid>
56        <grid id="86961" binding="panelAnotherForms" layout-manager="GridLayoutManager" row-count="1" column-
    count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
57          <margin top="0" left="0" bottom="0" right="0"/>
58          <constraints>
59            <grid row="0" column="1" row-span="2" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0"
    fill="3" indent="0" use-parent-layout="false"/>
60          </constraints>
61          <properties/>
62          <border type="none"/>
63          <children/>
64        </grid>
65      </children>
66    </grid>
67  </form>
68
```

```java
import entities.User;

import javax.swing.*;
import java.awt.*;
import java.io.File;
import java.io.IOException;

public class CandidateHomepage extends JFrame{
    private JPanel panelMain;
    private JButton buttonProfile;
    private JPanel panelSidebar;
    private JButton buttonSignOff;
    private JPanel panelAnotherForms;

    public static User user;

    public CandidateHomepage(User user){
        this.user = user;
        this.add(panelMain);
        this.setSize(1280,720);
        this.setTitle("Candidate Homepage");
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setResizable(true);

        File currentDirFile = new File("");

        String customerPicture = "";
        String signOffPicture = "";

        try {
            customerPicture = currentDirFile.getCanonicalPath() + "\\src\\main\\java\\images\\customer.png";
            signOffPicture = currentDirFile.getCanonicalPath() + "\\src\\main\\java\\images\\logout.png";
        } catch (IOException e) {
            e.printStackTrace();
        }

        ImageIcon customerIcon = new ImageIcon(customerPicture);
        ImageIcon signOffIcon = new ImageIcon(signOffPicture);
        buttonProfile.setIcon(customerIcon);
```

```
40          buttonSignOff.setIcon(signOffIcon);
41
42          buttonSignOff.addActionListener(e -> {
43              this.setVisible(false);
44              Login form = new Login();
45              form.setVisible(true);
46          });
47
48          buttonProfile.addActionListener(e -> {
49              panelAnotherForms.removeAll();
50              panelAnotherForms.revalidate();
51              panelAnotherForms.repaint();
52              panelAnotherForms.setLayout(new GridLayout(1,0));
53              panelAnotherForms.add(new CandidateProfile(user).getContentPane());
54              panelAnotherForms.repaint();
55              panelAnotherForms.revalidate();
56          });
57
58
59      }
60
61
62      public static void main(String[] args){
63          try {
64              UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
65          } catch (ClassNotFoundException e) {
66              e.printStackTrace();
67          } catch (InstantiationException e) {
68              e.printStackTrace();
69          } catch (IllegalAccessException e) {
70              e.printStackTrace();
71          } catch (UnsupportedLookAndFeelException e) {
72              e.printStackTrace();
73          }
74
75          //Set UI Thread to another thread
76          SwingUtilities.invokeLater(() -> {
77              CandidateHomepage gui = new CandidateHomepage(user);
78              gui.setVisible(true);
```

```
79          });
80      }
81 }
82
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<form xmlns="http://www.intellij.com/uidesigner/form/" version="1" bind-to-class="RegisterCandidate">
  <grid id="27dc6" binding="panelMain" layout-manager="GridLayoutManager" row-count="16" column-count="9" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
    <margin top="20" left="20" bottom="20" right="20"/>
    <constraints>
      <xy x="20" y="20" width="539" height="607"/>
    </constraints>
    <properties>
      <enabled value="false"/>
    </properties>
    <border type="none"/>
    <children>
      <vspacer id="c0b3c">
        <constraints>
          <grid row="15" column="0" row-span="1" col-span="9" vsize-policy="6" hsize-policy="1" anchor="0" fill="2" indent="0" use-parent-layout="false"/>
        </constraints>
      </vspacer>
      <vspacer id="f429">
        <constraints>
          <grid row="1" column="0" row-span="1" col-span="9" vsize-policy="6" hsize-policy="1" anchor="0" fill="2" indent="0" use-parent-layout="false"/>
        </constraints>
      </vspacer>
      <component id="726eb" class="javax.swing.JLabel">
        <constraints>
          <grid row="2" column="0" row-span="1" col-span="6" vsize-policy="0" hsize-policy="0" anchor="8" fill="0" indent="0" use-parent-layout="false"/>
        </constraints>
        <properties>
          <font size="16"/>
          <text value="Email : "/>
        </properties>
      </component>
      <component id="30873" class="javax.swing.JTextField" binding="textFieldEmail">
        <constraints>
          <grid row="2" column="6" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8" fill="1" indent="0" use-parent-layout="false">
```

```
35              <preferred-size width="150" height="-1"/>
36            </grid>
37          </constraints>
38          <properties/>
39        </component>
40        <component id="bdee" class="javax.swing.JLabel">
41          <constraints>
42            <grid row="4" column="0" row-span="1" col-span="5" vsize-policy="0" hsize-policy="0" anchor="8"
   fill="0" indent="0" use-parent-layout="false"/>
43          </constraints>
44          <properties>
45            <font size="16"/>
46            <text value="First Name : "/>
47          </properties>
48        </component>
49        <component id="b9958" class="javax.swing.JLabel">
50          <constraints>
51            <grid row="6" column="0" row-span="1" col-span="4" vsize-policy="0" hsize-policy="0" anchor="8"
   fill="0" indent="0" use-parent-layout="false"/>
52          </constraints>
53          <properties>
54            <font size="16"/>
55            <text value="Last Name : "/>
56          </properties>
57        </component>
58        <component id="40f18" class="javax.swing.JLabel">
59          <constraints>
60            <grid row="8" column="0" row-span="1" col-span="3" vsize-policy="0" hsize-policy="0" anchor="8"
   fill="0" indent="0" use-parent-layout="false"/>
61          </constraints>
62          <properties>
63            <font size="16"/>
64            <text value="Identity Id : "/>
65          </properties>
66        </component>
67        <component id="53602" class="javax.swing.JLabel">
68          <constraints>
69            <grid row="10" column="0" row-span="1" col-span="2" vsize-policy="0" hsize-policy="0" anchor="8"
   fill="0" indent="0" use-parent-layout="false"/>
```

```
        </constraints>
        <properties>
          <font size="16"/>
          <text value="Birth Year : "/>
        </properties>
      </component>
      <component id="a7c56" class="javax.swing.JLabel">
        <constraints>
          <grid row="12" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8" fill="0" indent="0" use-parent-layout="false"/>
        </constraints>
        <properties>
          <font size="16"/>
          <text value="Password : "/>
        </properties>
      </component>
      <component id="f9dc3" class="javax.swing.JTextField" binding="textFieldFirstName">
        <constraints>
          <grid row="4" column="6" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8" fill="1" indent="0" use-parent-layout="false">
            <preferred-size width="150" height="-1"/>
          </grid>
        </constraints>
        <properties/>
      </component>
      <component id="29757" class="javax.swing.JTextField" binding="textFieldLastName">
        <constraints>
          <grid row="6" column="6" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8" fill="1" indent="0" use-parent-layout="false">
            <preferred-size width="150" height="-1"/>
          </grid>
        </constraints>
        <properties/>
      </component>
      <component id="77724" class="javax.swing.JTextField" binding="textFieldIdentityId">
        <constraints>
          <grid row="8" column="6" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8" fill="1" indent="0" use-parent-layout="false">
            <preferred-size width="150" height="-1"/>
```

```
105           </grid>
106         </constraints>
107         <properties/>
108       </component>
109       <component id="2c5c8" class="javax.swing.JTextField" binding="textFieldBirthYear">
110         <constraints>
111           <grid row="10" column="6" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8"
    fill="1" indent="0" use-parent-layout="false">
112             <preferred-size width="150" height="-1"/>
113           </grid>
114         </constraints>
115         <properties/>
116       </component>
117       <grid id="e2cc7" layout-manager="GridLayoutManager" row-count="1" column-count="1" same-size-
    horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
118         <margin top="0" left="0" bottom="0" right="0"/>
119         <constraints>
120           <grid row="14" column="6" row-span="1" col-span="3" vsize-policy="3" hsize-policy="3" anchor="0"
    fill="3" indent="0" use-parent-layout="false"/>
121         </constraints>
122         <properties/>
123         <border type="none"/>
124         <children>
125           <component id="ed615" class="javax.swing.JButton" binding="buttonRegister">
126             <constraints>
127               <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor="0
    " fill="1" indent="0" use-parent-layout="false"/>
128             </constraints>
129             <properties>
130               <font size="15"/>
131               <text value="Register"/>
132             </properties>
133           </component>
134         </children>
135       </grid>
136       <grid id="b1e3" layout-manager="GridLayoutManager" row-count="1" column-count="1" same-size-
    horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
137         <margin top="0" left="0" bottom="0" right="0"/>
138         <constraints>
```

```
          <grid row="14" column="0" row-span="1" col-span="6" vsize-policy="3" hsize-policy="3" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
        </constraints>
        <properties/>
        <border type="none"/>
        <children>
          <component id="33eee" class="javax.swing.JButton" binding="buttonGoBack">
            <constraints>
              <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor="0" fill="1" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties>
              <font size="15"/>
              <text value="Go Back To Login"/>
            </properties>
          </component>
        </children>
      </grid>
      <grid id="b67a7" layout-manager="GridLayoutManager" row-count="1" column-count="3" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
        <margin top="0" left="0" bottom="0" right="0"/>
        <constraints>
          <grid row="0" column="0" row-span="1" col-span="9" vsize-policy="3" hsize-policy="3" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
        </constraints>
        <properties/>
        <border type="none"/>
        <children>
          <component id="8a013" class="javax.swing.JLabel" binding="textFieldTittle">
            <constraints>
              <grid row="0" column="1" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8" fill="0" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties>
              <enabled value="true"/>
              <font size="25"/>
              <text value="Candidate Register"/>
            </properties>
          </component>
```

```
173           <hspacer id="ae3e">
174             <constraints>
175               <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="1" hsize-policy="6" anchor="0
    " fill="1" indent="0" use-parent-layout="false"/>
176             </constraints>
177           </hspacer>
178           <hspacer id="807df">
179             <constraints>
180               <grid row="0" column="2" row-span="1" col-span="1" vsize-policy="1" hsize-policy="6" anchor="0
    " fill="1" indent="0" use-parent-layout="false"/>
181             </constraints>
182           </hspacer>
183         </children>
184       </grid>
185       <component id="e6a7a" class="javax.swing.JLabel" binding="LabelEmailValidation">
186         <constraints>
187           <grid row="3" column="6" row-span="1" col-span="2" vsize-policy="0" hsize-policy="0" anchor="8"
    fill="0" indent="0" use-parent-layout="false"/>
188         </constraints>
189         <properties>
190           <foreground color="-9043957"/>
191           <text value=""/>
192         </properties>
193       </component>
194       <component id="91244" class="javax.swing.JLabel" binding="LabelFirstNameValidation">
195         <constraints>
196           <grid row="5" column="6" row-span="1" col-span="2" vsize-policy="0" hsize-policy="0" anchor="8"
    fill="0" indent="0" use-parent-layout="false"/>
197         </constraints>
198         <properties>
199           <foreground color="-9043957"/>
200           <text value=""/>
201         </properties>
202       </component>
203       <component id="639ee" class="javax.swing.JLabel" binding="LabelLastNameValidation">
204         <constraints>
205           <grid row="7" column="6" row-span="1" col-span="2" vsize-policy="0" hsize-policy="0" anchor="8"
    fill="0" indent="0" use-parent-layout="false"/>
206         </constraints>
```

```
207         <properties>
208           <foreground color="-9043957"/>
209           <text value=""/>
210         </properties>
211       </component>
212       <component id="9e0d0" class="javax.swing.JLabel" binding="LabelIdentityIdValidation">
213         <constraints>
214           <grid row="9" column="6" row-span="1" col-span="2" vsize-policy="0" hsize-policy="0" anchor="8"
    fill="0" indent="0" use-parent-layout="false"/>
215         </constraints>
216         <properties>
217           <background color="-9043957"/>
218           <foreground color="-9043957"/>
219           <text value=""/>
220         </properties>
221       </component>
222       <component id="f156d" class="javax.swing.JLabel" binding="LabelBirthYearValidation">
223         <constraints>
224           <grid row="11" column="6" row-span="1" col-span="2" vsize-policy="0" hsize-policy="0" anchor="8"
    fill="0" indent="0" use-parent-layout="false"/>
225         </constraints>
226         <properties>
227           <background color="-9043957"/>
228           <foreground color="-9043957"/>
229           <text value=""/>
230         </properties>
231       </component>
232       <component id="de829" class="javax.swing.JTextArea" binding="TextAreaPasswordValidation">
233         <constraints>
234           <grid row="13" column="6" row-span="1" col-span="3" vsize-policy="6" hsize-policy="6" anchor="0"
    fill="3" indent="0" use-parent-layout="false">
235             <preferred-size width="150" height="50"/>
236           </grid>
237         </constraints>
238         <properties>
239           <font name="Tahoma" size="11" style="0"/>
240           <foreground color="-9043957"/>
241           <lineWrap value="true"/>
242           <visible value="false"/>
```

```
        </properties>
      </component>
      <component id="4517b" class="javax.swing.JPasswordField" binding="passwordFieldPassword">
        <constraints>
          <grid row="12" column="6" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8" fill="1" indent="0" use-parent-layout="false">
            <preferred-size width="150" height="-1"/>
          </grid>
        </constraints>
        <properties/>
      </component>
    </children>
  </grid>
</form>
```

```java
import org.json.JSONObject;

import javax.swing.*;
import java.awt.*;
import java.io.*;
import java.net.HttpURLConnection;
import java.net.URL;
import java.nio.charset.StandardCharsets;

public class RegisterCandidate extends JFrame{
    private JTextField textFieldEmail;
    private JButton buttonRegister;
    private JButton buttonGoBack;
    private JTextField textFieldFirstName;
    private JTextField textFieldLastName;
    private JTextField textFieldIdentityId;
    private JTextField textFieldBirthYear;
    private JPanel panelMain;
    private JLabel LabelEmailValidation;
    private JLabel LabelFirstNameValidation;
    private JLabel LabelLastNameValidation;
    private JLabel LabelIdentityIdValidation;
    private JLabel LabelBirthYearValidation;
    private JLabel LabelPasswordValidation;
    private JLabel textFieldTittle;
    private JTextArea TextAreaPasswordValidation;
    private JPasswordField passwordFieldPassword;

    public RegisterCandidate(){
        this.add(panelMain);
        this.setSize(500,350);
        this.setTitle("Candidate Register");
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setResizable(false);

        File currentDirFile = new File("");

        String helper = "";

```

```
40          try {
41              helper = currentDirFile.getCanonicalPath() + "\\src\\main\\java\\images\\register.png";
42          } catch (IOException e) {
43              e.printStackTrace();
44          }
45
46          ImageIcon icon = new ImageIcon(helper);
47          textFieldTittle.setIcon(icon);
48
49          buttonGoBack.addActionListener(e -> {
50              this.setVisible(false);
51              new Login().setVisible(true);
52          });
53
54          buttonRegister.addActionListener(e -> {
55              try {
56                  Register();
57              } catch (Exception ex) {
58                  JOptionPane.showMessageDialog(this,ex.getMessage());
59              }
60          });
61
62          TestMethod();
63
64      }
65
66      private void TestMethod() {
67          textFieldEmail.setText("sahin.maral@hotmail.com");
68          textFieldFirstName.setText("Şahin");
69          textFieldLastName.setText("Maral");
70          textFieldIdentityId.setText("31241146608");
71          textFieldBirthYear.setText("2000");
72      }
73
74
75      private void Register() throws Exception{
76
77          JSONObject obj = new JSONObject();
78          obj.put("birthYear", textFieldBirthYear.getText());
```

```
 79          obj.put("firstName", textFieldFirstName.getText());
 80          obj.put("lastName", textFieldLastName.getText());
 81          obj.put("identityId", textFieldIdentityId.getText());
 82          obj.put("password", String.valueOf(passwordFieldPassword.getPassword()));
 83          obj.put("email", textFieldEmail.getText());
 84          obj.put("passwordRepeat",String.valueOf(passwordFieldPassword.getPassword()));
 85
 86          URL url = new URL("http://localhost:8080/api/candidates/register");
 87          HttpURLConnection http = (HttpURLConnection)url.openConnection();
 88          http.setRequestMethod("POST");
 89          http.setDoOutput(true);
 90          http.setRequestProperty("Content-Type", "application/json");
 91
 92
 93          String data = obj.toString();
 94
 95          byte[] out = data.getBytes(StandardCharsets.UTF_8);
 96
 97          OutputStream stream = http.getOutputStream();
 98          stream.write(out);
 99
100          InputStream inputStream;
101
102          if (http.getResponseCode() != 200)
103              inputStream = http.getErrorStream();
104          else
105              inputStream = http.getInputStream();
106
107          String response = "";
108          String line;
109          BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
110
111          while ((line = br.readLine()) != null) {
112              response += line;
113          }
114
115          JSONObject returnObject = new JSONObject(response);
116
117          if (returnObject.get("success").equals(true)){
```

```
118             JOptionPane.showMessageDialog(this,returnObject.get("message"));
119             http.disconnect();
120             this.setVisible(false);
121             new Login().setVisible(true);
122         }
123 
124         else if(!response.contains("data")){
125             JOptionPane.showMessageDialog(this,returnObject.get("message"));
126         }
127 
128         else {
129 
130             clearValidations();
131 
132             JSONObject validationErrors = returnObject.getJSONObject("data");
133             Dimension dimension = this.getSize();
134 
135             String content = validationErrors.toString();
136             if (content.contains("birthYear")){
137                 if (LabelBirthYearValidation.getText().length() == 0)
138                 dimension.height += 50;
139                 LabelBirthYearValidation.setText(validationErrors.get("birthYear").toString());
140             }
141 
142             if (content.contains("password")) {
143 
144                 if (TextAreaPasswordValidation.getText().equals(""))
145                     dimension.height += 50;
146                 TextAreaPasswordValidation.setVisible(true);
147                 TextAreaPasswordValidation.setText(validationErrors.get("password").toString());
148             }
149             if (content.contains("lastName")){
150                 if (LabelLastNameValidation.getText().equals(""))
151                     dimension.height += 50;
152                 LabelLastNameValidation.setText(validationErrors.get("lastName").toString());
153             }
154 
155             if (content.contains("identityId")){
156                 if (LabelIdentityIdValidation.getText().equals(""))
```

```
157                     dimension.height += 50;
158                 LabelIdentityIdValidation.setText(validationErrors.get("identityId").toString());
159             }
160 
161             if (content.contains("firstName")){
162                 if (LabelFirstNameValidation.getText().equals(""))
163                     dimension.height += 50;
164                 LabelFirstNameValidation.setText(validationErrors.get("firstName").toString());
165             }
166 
167             if (content.contains("email")){
168                 if (LabelEmailValidation.getText().equals(""))
169                     dimension.height += 50;
170                 LabelEmailValidation.setText(validationErrors.get("email").toString());
171             }
172 
173             this.setSize(dimension);
174 
175         }
176 
177 
178 
179     }
180 
181     private void clearValidations() {
182         TextAreaPasswordValidation.setText("");
183         LabelEmailValidation.setText("");
184         LabelBirthYearValidation.setText("");
185         LabelFirstNameValidation.setText("");
186         LabelLastNameValidation.setText("");
187         LabelIdentityIdValidation.setText("");
188     }
189 
190     public static void main(String[] args){
191         try {
192             UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
193         } catch (ClassNotFoundException e) {
194             e.printStackTrace();
195         } catch (InstantiationException e) {
```

```
196             e.printStackTrace();
197         } catch (IllegalAccessException e) {
198             e.printStackTrace();
199         } catch (UnsupportedLookAndFeelException e) {
200             e.printStackTrace();
201         }
202 
203         //Set UI Thread to another thread
204         SwingUtilities.invokeLater(() -> {
205             RegisterCandidate gui = new RegisterCandidate();
206             gui.setVisible(true);
207         });
208     }
209 }
210 
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<form xmlns="http://www.intellij.com/uidesigner/form/" version="1" bind-to-class="CandidateSchoolsForSavedCvs">
  <grid id="27dc6" binding="panelMain" layout-manager="GridLayoutManager" row-count="2" column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
    <margin top="10" left="10" bottom="10" right="10"/>
    <constraints>
      <xy x="20" y="20" width="1018" height="584"/>
    </constraints>
    <properties/>
    <border type="none"/>
    <children>
      <grid id="f4ea4" binding="panelTable" layout-manager="GridLayoutManager" row-count="3" column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
        <margin top="0" left="0" bottom="0" right="0"/>
        <constraints>
          <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
        </constraints>
        <properties/>
        <border type="none"/>
        <children>
          <scrollpane id="4ba54" binding="scrollPaneTable">
            <constraints>
              <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="7" hsize-policy="7" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties/>
            <border type="none"/>
            <children>
              <component id="c0ee0" class="javax.swing.JTable" binding="tableSchool">
                <constraints/>
                <properties>
                  <minimumSize width="30" height="50"/>
                  <preferredSize width="150" height="340"/>
                  <requestFocusEnabled value="false"/>
                </properties>
              </component>
            </children>
```

```
35            </scrollpane>
36            <grid id="72d19" layout-manager="GridLayoutManager" row-count="1" column-count="1" same-size-
  horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
37              <margin top="0" left="0" bottom="0" right="0"/>
38              <constraints>
39                <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0"
   fill="3" indent="0" use-parent-layout="false"/>
40              </constraints>
41              <properties/>
42              <border type="none"/>
43              <children/>
44            </grid>
45            <grid id="79adb" binding="panelButtons" layout-manager="GridLayoutManager" row-count="1" column-
  count="4" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
46              <margin top="0" left="0" bottom="0" right="0"/>
47              <constraints>
48                <grid row="2" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0"
   fill="3" indent="0" use-parent-layout="false"/>
49              </constraints>
50              <properties/>
51              <border type="none"/>
52              <children>
53                <component id="9e95c" class="javax.swing.JButton" binding="buttonCreateSchool">
54                  <constraints>
55                    <grid row="0" column="1" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor
  ="0" fill="1" indent="0" use-parent-layout="false"/>
56                  </constraints>
57                  <properties>
58                    <text value="Create School"/>
59                  </properties>
60                </component>
61                <component id="ef0c6" class="javax.swing.JButton" binding="buttonEditSchool">
62                  <constraints>
63                    <grid row="0" column="3" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor
  ="0" fill="1" indent="0" use-parent-layout="false"/>
64                  </constraints>
65                  <properties>
66                    <text value="Edit Selected School"/>
67                  </properties>
```

```
 68                 </component>
 69                 <component id="fe18a" class="javax.swing.JButton" binding="buttonDeleteTextFields">
 70                   <constraints>
 71                     <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3"
    anchor="0" fill="1" indent="0" use-parent-layout="false"/>
 72                   </constraints>
 73                   <properties>
 74                     <text value="Refresh Page"/>
 75                   </properties>
 76                 </component>
 77                 <component id="42521" class="javax.swing.JButton" binding="buttonDeleteSchool">
 78                   <constraints>
 79                     <grid row="0" column="2" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3"
    anchor="0" fill="1" indent="0" use-parent-layout="false"/>
 80                   </constraints>
 81                   <properties>
 82                     <text value="Delete Selected School"/>
 83                   </properties>
 84                 </component>
 85               </children>
 86             </grid>
 87           </children>
 88         </grid>
 89         <grid id="87999" binding="panelSchoolIsnformations" layout-manager="GridLayoutManager" row-count="5"
    column-count="5" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
 90           <margin top="0" left="0" bottom="0" right="0"/>
 91           <constraints>
 92             <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0"
    fill="3" indent="0" use-parent-layout="false"/>
 93           </constraints>
 94           <properties/>
 95           <border type="none"/>
 96           <children>
 97             <component id="60c9e" class="javax.swing.JTextField" binding="textFieldSchoolId">
 98               <constraints>
 99                 <grid row="0" column="2" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8
    " fill="1" indent="0" use-parent-layout="false">
100                   <preferred-size width="150" height="-1"/>
101                 </grid>
```

```
102             </constraints>
103             <properties>
104               <editable value="false"/>
105               <enabled value="false"/>
106             </properties>
107           </component>
108           <component id="309bd" class="javax.swing.JLabel" binding="labelCvId">
109             <constraints>
110               <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
111             </constraints>
112             <properties>
113               <text value="Id : "/>
114             </properties>
115           </component>
116           <component id="911f8" class="javax.swing.JTextField" binding="textFieldName">
117             <constraints>
118               <grid row="1" column="2" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8
    " fill="1" indent="0" use-parent-layout="false">
119                 <preferred-size width="150" height="-1"/>
120               </grid>
121             </constraints>
122             <properties/>
123           </component>
124           <component id="54e27" class="javax.swing.JTextField" binding="textFieldDepartment">
125             <constraints>
126               <grid row="2" column="2" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8
    " fill="1" indent="0" use-parent-layout="false">
127                 <preferred-size width="150" height="-1"/>
128               </grid>
129             </constraints>
130             <properties/>
131           </component>
132           <component id="743c3" class="javax.swing.JLabel" binding="labelCvLinkedinAdress">
133             <constraints>
134               <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
135             </constraints>
136             <properties>
```

```
137                 <text value="Name : "/>
138               </properties>
139             </component>
140             <hspacer id="fcd97">
141               <constraints>
142                 <grid row="2" column="1" row-span="1" col-span="1" vsize-policy="1" hsize-policy="6" anchor="0
    " fill="1" indent="0" use-parent-layout="false"/>
143               </constraints>
144             </hspacer>
145             <component id="d8a49" class="javax.swing.JLabel" binding="labelCvGithubAdress">
146               <constraints>
147                 <grid row="2" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
148               </constraints>
149               <properties>
150                 <text value="Department : "/>
151               </properties>
152             </component>
153             <component id="8f76a" class="javax.swing.JLabel" binding="labelCvCoveringLetter">
154               <constraints>
155                 <grid row="3" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
156               </constraints>
157               <properties>
158                 <text value="Started Date : "/>
159               </properties>
160             </component>
161             <component id="5b70e" class="javax.swing.JLabel">
162               <constraints>
163                 <grid row="4" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
164               </constraints>
165               <properties>
166                 <text value="Graduated Date : "/>
167               </properties>
168             </component>
169             <grid id="67a8a" binding="panelStartedDate" layout-manager="GridLayoutManager" row-count="1"
    column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
170               <margin top="0" left="0" bottom="0" right="0"/>
```

```
171             <constraints>
172               <grid row="3" column="2" row-span="1" col-span="3" vsize-policy="3" hsize-policy="3" anchor="0
    " fill="3" indent="0" use-parent-layout="false"/>
173             </constraints>
174             <properties/>
175             <border type="none"/>
176             <children/>
177           </grid>
178           <grid id="24820" binding="panelGraduatedDate" layout-manager="GridLayoutManager" row-count="1"
    column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
179             <margin top="0" left="0" bottom="0" right="0"/>
180             <constraints>
181               <grid row="4" column="2" row-span="1" col-span="3" vsize-policy="3" hsize-policy="3" anchor="0
    " fill="3" indent="0" use-parent-layout="false"/>
182             </constraints>
183             <properties/>
184             <border type="none"/>
185             <children/>
186           </grid>
187         </children>
188       </grid>
189     </children>
190   </grid>
191 </form>
192
```

```java
import com.toedter.calendar.JDateChooser;
import entities.*;

import org.json.JSONArray;
import org.json.JSONObject;

import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.event.WindowEvent;
import java.io.*;
import java.net.HttpURLConnection;
import java.net.URL;
import java.nio.charset.StandardCharsets;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.time.LocalDateTime;
import java.time.ZoneId;
import java.util.*;
import java.util.List;

public class CandidateSchoolsForSavedCvs extends JFrame {
    private JPanel panelMain;
    private JPanel panelTable;
    private JScrollPane scrollPaneTable;
    private JTable tableSchool;
    private JPanel panelButtons;
    private JButton buttonCreateSchool;
    private JButton buttonEditSchool;
    private JButton buttonDeleteTextFields;
    private JPanel panelSchoolIsnformations;
    private JTextField textFieldSchoolId;
    private JLabel labelCvId;
    private JTextField textFieldName;
    private JTextField textFieldDepartment;
    private JLabel labelCvLinkedinAdress;
    private JLabel labelCvGithubAdress;
```

```
40      private JLabel labelCvCoveringLetter;
41      private JPanel panelStartedDate;
42      private JPanel panelGraduatedDate;
43      private JButton buttonDeleteSchool;
44      private JDateChooser chooserGraduatedDate;
45      private JDateChooser chooserStartedDate;
46
47      public User user;
48      public String cvId;
49      List<School> schools = new ArrayList<>();
50
51      public CandidateSchoolsForSavedCvs(User user, String cvId) {
52          this.user = user;
53          this.cvId = cvId;
54
55          this.add(panelMain);
56          this.setSize(720, 600);
57          this.dispatchEvent(new WindowEvent(this, WindowEvent.WINDOW_CLOSING));
58
59          this.setResizable(false);
60          this.setTitle("Schools for saved cvs");
61
62          chooserStartedDate = new JDateChooser();
63          panelStartedDate.setLayout(new GridLayout(1,0));
64          chooserStartedDate.setDateFormatString("yyyy-MM-dd");
65          panelStartedDate.add(chooserStartedDate);
66
67          chooserGraduatedDate = new JDateChooser();
68          panelGraduatedDate.setLayout(new GridLayout(1,0));
69          chooserGraduatedDate.setDateFormatString("yyyy-MM-dd");
70
71          panelGraduatedDate.add(chooserGraduatedDate);
72
73          tableSchool.addMouseListener(new MouseAdapter() {
74              @Override
75              public void mouseClicked(MouseEvent e) {
76
77                  deleteTextFields();
78
```

```
 79                 DefaultTableModel model = (DefaultTableModel) tableSchool.getModel();
 80
 81                 int selectedRowIndex = tableSchool.getSelectedRow();
 82                 textFieldSchoolId.setText(model.getValueAt(selectedRowIndex,0).toString());
 83                 textFieldName.setText(model.getValueAt(selectedRowIndex,1).toString());
 84                 textFieldDepartment.setText(model.getValueAt(selectedRowIndex,2).toString());
 85
 86                 Date startedDate= null;
 87                 try {
 88                     startedDate = new SimpleDateFormat("yyyy-MM-dd").parse((String) model.getValueAt(
    selectedRowIndex,3));
 89                 } catch (ParseException ex) {
 90                     ex.printStackTrace();
 91                 }
 92
 93                 chooserStartedDate.setDate(startedDate);
 94
 95
 96                 if (model.getValueAt(selectedRowIndex,4)!=null){
 97                     Date graduatedDate= null;
 98                     try {
 99                         graduatedDate = new SimpleDateFormat("yyyy-MM-dd").parse((String) model.getValueAt(
    selectedRowIndex,4));
100                     } catch (ParseException ex) {
101                         ex.printStackTrace();
102                     }
103                     chooserGraduatedDate.setDate(graduatedDate);
104                 }
105
106
107             }
108         });
109
110         if (!cvId.equals("")){
111             try {
112                 getSchoolsFromApi();
113                 assignTable();
114             } catch (Exception e) {
115                 JOptionPane.showMessageDialog(this,e.getMessage());
```

```
116             }
117         }
118 
119         buttonCreateSchool.addActionListener(e -> {
120             try {
121                 addSchool();
122             } catch (Exception ex) {
123                 JOptionPane.showMessageDialog(this, ex.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
124             }
125         });
126 
127         buttonDeleteSchool.addActionListener(e -> {
128             try {
129                 deleteSchool();
130             } catch (Exception ex) {
131                 JOptionPane.showMessageDialog(this, ex.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
132             }
133         });
134 
135         buttonEditSchool.addActionListener(e -> {
136             try {
137                 updateSchool();
138             } catch (Exception ex) {
139                 JOptionPane.showMessageDialog(this, ex.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
140             }
141         });
142 
143         buttonDeleteTextFields.addActionListener(e -> deleteTextFields());
144     }
145 
146     private void addSchool() throws Exception{
147         JSONObject obj = new JSONObject();
148         obj.put("id", textFieldSchoolId.getText());
149         obj.put("department", textFieldDepartment.getText());
150         obj.put("name", textFieldName.getText());
151 
152         JSONObject cvJson = new JSONObject();
153         cvJson.put("id",cvId);
154 
```

```
155         obj.put("cv",cvJson);
156
157         JSONObject candidate = new JSONObject();
158         candidate.put("id", user.id);
159         cvJson.put("candidate",candidate);
160
161         var startedDate = convertToLocalDateTimeViaInstant(chooserStartedDate.getDate());
162         obj.put("startedDate",startedDate);
163
164         if (chooserGraduatedDate.getDate()!=null){
165             var graduatedDate = convertToLocalDateTimeViaInstant(chooserGraduatedDate.getDate());
166             obj.put("graduatedDate",graduatedDate);
167         }
168         else{
169             obj.put("graduatedDate",JSONObject.NULL);
170         }
171
172
173
174         URL url = new URL("http://localhost:8080/api/schools/addSchool");
175         HttpURLConnection http = (HttpURLConnection)url.openConnection();
176         http.setRequestMethod("POST");
177         http.setDoOutput(true);
178         http.setRequestProperty("Content-Type", "application/json");
179
180
181         String data = obj.toString();
182
183         byte[] out = data.getBytes(StandardCharsets.UTF_8);
184
185         OutputStream stream = http.getOutputStream();
186         stream.write(out);
187
188         InputStream inputStream;
189
190         if (http.getResponseCode() != 200)
191             inputStream = http.getErrorStream();
192         else
193             inputStream = http.getInputStream();
```

```
194
195         String response = "";
196         String line;
197         BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
198
199         while ((line = br.readLine()) != null) {
200             response += line;
201         }
202
203         JSONObject returnObject = new JSONObject(response);
204
205         if (returnObject.get("success").equals(true)){
206             JOptionPane.showMessageDialog(this,returnObject.get("message"));
207             http.disconnect();
208             assignTable();
209             deleteTextFields();
210         }
211
212         else{
213             http.disconnect();
214             JOptionPane.showMessageDialog(this,returnObject.get("message"));
215         }
216
217     }
218
219     public LocalDateTime convertToLocalDateTimeViaInstant(Date dateToConvert) {
220         return dateToConvert.toInstant()
221                 .atZone(ZoneId.systemDefault())
222                 .toLocalDateTime();
223     }
224
225     private void updateSchool() throws Exception {
226         if (textFieldSchoolId.getText().equals(""))
227             throw new NullPointerException("Herhangi bir okul işaretlemediniz");
228
229         JSONObject obj = new JSONObject();
230         obj.put("id", textFieldSchoolId.getText());
231         obj.put("department", textFieldDepartment.getText());
232         obj.put("name", textFieldName.getText());
```

```
233
234         JSONObject cvJson = new JSONObject();
235         cvJson.put("id",cvId);
236
237         obj.put("cv",cvJson);
238
239         JSONObject candidate = new JSONObject();
240         candidate.put("id", user.id);
241         cvJson.put("candidate",candidate);
242
243         var startedDate = convertToLocalDateTimeViaInstant(chooserStartedDate.getDate());
244         obj.put("startedDate",startedDate);
245
246         if (chooserGraduatedDate.getDate()!=null){
247             var graduatedDate = convertToLocalDateTimeViaInstant(chooserGraduatedDate.getDate());
248             obj.put("graduatedDate",graduatedDate);
249         }
250         else{
251             obj.put("graduatedDate",JSONObject.NULL);
252         }
253
254
255
256         URL url = new URL("http://localhost:8080/api/schools/updateSchool");
257         HttpURLConnection http = (HttpURLConnection)url.openConnection();
258         http.setRequestMethod("PUT");
259         http.setDoOutput(true);
260         http.setRequestProperty("Content-Type", "application/json");
261
262
263         String data = obj.toString();
264
265         byte[] out = data.getBytes(StandardCharsets.UTF_8);
266
267         OutputStream stream = http.getOutputStream();
268         stream.write(out);
269
270         InputStream inputStream;
271
```

```
272         if (http.getResponseCode() != 200)
273             inputStream = http.getErrorStream();
274         else
275             inputStream = http.getInputStream();
276 
277         String response = "";
278         String line;
279         BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
280 
281         while ((line = br.readLine()) != null) {
282             response += line;
283         }
284 
285         JSONObject returnObject = new JSONObject(response);
286 
287         if (returnObject.get("success").equals(true)){
288             JOptionPane.showMessageDialog(this,returnObject.get("message"));
289             http.disconnect();
290             assignTable();
291         }
292 
293         else{
294             http.disconnect();
295             JOptionPane.showMessageDialog(this,returnObject.get("message"));
296         }
297 
298     }
299 
300     private void assignTable(){
301 
302         String[] columns = new String[] {
303                 "Id","Name", "Department", "Started Date" , "Graduated Date" , "Is Graduated"
304         };
305 
306         try {
307             schools = getSchoolsFromApi();
308         } catch (Exception e) {
309             JOptionPane.showMessageDialog(this,e.getMessage());
310         }
```

```
311
312         //actual data for the table in a 2d array
313         Object[][] data = new Object[schools.size()][6];
314
315         for(var i = 0; i < schools.size(); i++){
316             data[i][0] = schools.get(i).id;
317             data[i][1] = schools.get(i).name;
318             data[i][2] = schools.get(i).department;
319             data[i][3] = schools.get(i).startedDate;
320             if (schools.get(i).graduatedDate!=null){
321                 data[i][4] = schools.get(i).graduatedDate;
322                 data[i][5] = "Mezun olmuş";
323             }
324             else{
325                 data[i][5] = "Mezun olmamış";
326             }
327
328         }
329
330         DefaultTableModel model = new DefaultTableModel(data,columns){
331             @Override
332             public boolean isCellEditable(int row, int column) {
333                 return false;
334             }
335         };
336
337         tableSchool.setModel(model);
338     }
339
340     private void deleteSchool() throws Exception {
341
342         if (textFieldSchoolId.getText().equals(""))
343             JOptionPane.showMessageDialog(this, "You didn't select any school");
344         else {
345
346             URL url = new URL("http://localhost:8080/api/schools/deleteSchool?schoolId=" + textFieldSchoolId
    .getText());
347             HttpURLConnection httpCon = (HttpURLConnection) url.openConnection();
348             httpCon.setDoOutput(true);
```

```
349             httpCon.setRequestMethod("DELETE");
350             httpCon.connect();
351 
352             InputStream inputStream;
353 
354             if (httpCon.getResponseCode() != 200)
355                 inputStream = httpCon.getErrorStream();
356             else
357                 inputStream = httpCon.getInputStream();
358 
359             String response = "";
360             String line;
361             BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
362 
363             while ((line = br.readLine()) != null) {
364                 response += line;
365             }
366 
367             JSONObject returnObject = new JSONObject(response);
368 
369             if (returnObject.getBoolean("success")) {
370                 JOptionPane.showMessageDialog(this, returnObject.getString("message"), "Information",
    JOptionPane.INFORMATION_MESSAGE);
371                 assignTable();
372                 deleteTextFields();
373             } else
374                 JOptionPane.showMessageDialog(this, returnObject.getString("message"), "Error", JOptionPane.
    ERROR_MESSAGE);
375         }
376     }
377 
378     public void deleteTextFields(){
379         textFieldSchoolId.setText("");
380         textFieldName.setText("");
381         textFieldDepartment.setText("");
382 
383         Date dateNow = null;
384         String currentDate = new SimpleDateFormat("yyyy-mm-dd").format(Calendar.getInstance().getTime());
385         try {
```

```
386             dateNow=new SimpleDateFormat("yyyy-mm-dd").parse(currentDate);
387         } catch (ParseException e) {
388             e.printStackTrace();
389         }
390         chooserStartedDate.setDate(dateNow);
391         chooserGraduatedDate.setDate(dateNow);
392 
393     }
394 
395     private List<School> getSchoolsFromApi() throws Exception{
396 
397         List<School> schools = new ArrayList<>();
398 
399         URL getSchoolsByCandidateId = new URL("http://localhost:8080/api/schools/getAllCvId?cvId="+Integer.
    parseInt(cvId));
400         HttpURLConnection conn = (HttpURLConnection) getSchoolsByCandidateId.openConnection();
401         conn.setRequestMethod("GET");
402         conn.connect();
403         int responseCode = conn.getResponseCode();
404 
405         if (responseCode != 200) {
406             throw new RuntimeException("HttpResponseCode: " + responseCode);
407         } else {
408 
409             String inline = "";
410             Scanner scanner = new Scanner(getSchoolsByCandidateId.openStream());
411 
412             //Write all the JSON data into a string using a scanner
413             while (scanner.hasNext()) {
414                 inline += scanner.nextLine();
415             }
416 
417             //Close the scanner
418             scanner.close();
419 
420             String jsonString = inline ;
421             JSONObject obj = new JSONObject(jsonString);
422 
423             JSONArray arr = obj.getJSONArray("data");
```

```
424             for (int i = 0; i < arr.length(); i++)
425             {
426                 School school = new School();
427                 school.id = arr.getJSONObject(i).getInt("id");
428                 school.department = arr.getJSONObject(i).getString("department");
429                 school.name = arr.getJSONObject(i).getString("name");
430                 school.startedDate = arr.getJSONObject(i).getString("startedDate");
431                 if (!arr.getJSONObject(i).isNull("graduatedDate"))
432                     school.graduatedDate = arr.getJSONObject(i).getString("graduatedDate");
433                 JSONObject cvJson = arr.getJSONObject(i).getJSONObject("cv");
434 
435                 Cv cv = new Cv();
436                 cv.id = cvJson.getInt("id");
437 
438                 Candidate candidate = new Candidate();
439                 candidate.id = cvJson.getJSONObject("candidate").getInt("id");
440                 cv.candidate = candidate;
441 
442                 schools.add(school);
443 
444             }
445 
446             return schools;
447 
448         }
449     }
450 
451     public void main(String[] args){
452         try {
453             UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
454         } catch (ClassNotFoundException e) {
455             e.printStackTrace();
456         } catch (InstantiationException e) {
457             e.printStackTrace();
458         } catch (IllegalAccessException e) {
459             e.printStackTrace();
460         } catch (UnsupportedLookAndFeelException e) {
461             e.printStackTrace();
462         }
```

```
463
464         //Set UI Thread to another thread
465         SwingUtilities.invokeLater(() -> {
466             CandidateSchoolsForSavedCvs gui = new CandidateSchoolsForSavedCvs(user,cvId);
467             gui.setVisible(true);
468         });
469
470     }
471
472
473 }
474
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<form xmlns="http://www.intellij.com/uidesigner/form/" version="1" bind-to-class="CandidateSchoolsForUnsavedCvs">
  <grid id="27dc6" binding="panelMain" layout-manager="GridLayoutManager" row-count="2" column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
    <margin top="10" left="10" bottom="10" right="10"/>
    <constraints>
      <xy x="20" y="20" width="663" height="400"/>
    </constraints>
    <properties/>
    <border type="none"/>
    <children>
      <grid id="40183" binding="panelTable" layout-manager="GridLayoutManager" row-count="3" column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
        <margin top="0" left="0" bottom="0" right="0"/>
        <constraints>
          <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
        </constraints>
        <properties/>
        <border type="none"/>
        <children>
          <scrollpane id="dcf75" binding="scrollPaneTable">
            <constraints>
              <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="7" hsize-policy="7" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties/>
            <border type="none"/>
            <children>
              <component id="bc556" class="javax.swing.JTable" binding="tableSchool">
                <constraints/>
                <properties>
                  <minimumSize width="30" height="50"/>
                  <preferredSize width="150" height="340"/>
                  <requestFocusEnabled value="false"/>
                </properties>
              </component>
            </children>
```

```
35            </scrollpane>
36            <grid id="a1192" layout-manager="GridLayoutManager" row-count="1" column-count="1" same-size-
  horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
37              <margin top="0" left="0" bottom="0" right="0"/>
38              <constraints>
39                <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0"
   fill="3" indent="0" use-parent-layout="false"/>
40              </constraints>
41              <properties/>
42              <border type="none"/>
43              <children/>
44            </grid>
45            <grid id="94620" binding="panelButtons" layout-manager="GridLayoutManager" row-count="1" column-
  count="4" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
46              <margin top="0" left="0" bottom="0" right="0"/>
47              <constraints>
48                <grid row="2" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0"
   fill="3" indent="0" use-parent-layout="false"/>
49              </constraints>
50              <properties/>
51              <border type="none"/>
52              <children>
53                <component id="84f80" class="javax.swing.JButton" binding="buttonCreateSchool">
54                  <constraints>
55                    <grid row="0" column="1" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor
  ="0" fill="1" indent="0" use-parent-layout="false"/>
56                  </constraints>
57                  <properties>
58                    <text value="Create School"/>
59                  </properties>
60                </component>
61                <component id="206e" class="javax.swing.JButton" binding="buttonEditSchool">
62                  <constraints>
63                    <grid row="0" column="3" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor
  ="0" fill="1" indent="0" use-parent-layout="false"/>
64                  </constraints>
65                  <properties>
66                    <text value="Edit Selected School"/>
67                  </properties>
```

```
 68                </component>
 69                <component id="f0f46" class="javax.swing.JButton" binding="buttonDeleteTextFields">
 70                  <constraints>
 71                    <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3"
    anchor="0" fill="1" indent="0" use-parent-layout="false"/>
 72                  </constraints>
 73                  <properties>
 74                    <text value="Refresh Page"/>
 75                  </properties>
 76                </component>
 77                <component id="e0f5b" class="javax.swing.JButton" binding="buttonDeleteSchool">
 78                  <constraints>
 79                    <grid row="0" column="2" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3"
    anchor="0" fill="1" indent="0" use-parent-layout="false"/>
 80                  </constraints>
 81                  <properties>
 82                    <text value="Delete Selected School"/>
 83                  </properties>
 84                </component>
 85              </children>
 86            </grid>
 87          </children>
 88        </grid>
 89        <grid id="dafae" binding="panelSchoolIsnformations" layout-manager="GridLayoutManager" row-count="5"
    column-count="5" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
 90          <margin top="0" left="0" bottom="0" right="0"/>
 91          <constraints>
 92            <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0"
    fill="3" indent="0" use-parent-layout="false"/>
 93          </constraints>
 94          <properties/>
 95          <border type="none"/>
 96          <children>
 97            <component id="7cdcb" class="javax.swing.JTextField" binding="textFieldSchoolId">
 98              <constraints>
 99                <grid row="0" column="2" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8
    " fill="1" indent="0" use-parent-layout="false">
100                  <preferred-size width="150" height="-1"/>
101                </grid>
```

```
102             </constraints>
103             <properties>
104               <editable value="false"/>
105               <enabled value="false"/>
106             </properties>
107           </component>
108           <component id="bb62a" class="javax.swing.JLabel" binding="labelCvId">
109             <constraints>
110               <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
111             </constraints>
112             <properties>
113               <text value="Id : "/>
114             </properties>
115           </component>
116           <component id="32857" class="javax.swing.JTextField" binding="textFieldName">
117             <constraints>
118               <grid row="1" column="2" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8
    " fill="1" indent="0" use-parent-layout="false">
119                 <preferred-size width="150" height="-1"/>
120               </grid>
121             </constraints>
122             <properties/>
123           </component>
124           <component id="9603b" class="javax.swing.JTextField" binding="textFieldDepartment">
125             <constraints>
126               <grid row="2" column="2" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8
    " fill="1" indent="0" use-parent-layout="false">
127                 <preferred-size width="150" height="-1"/>
128               </grid>
129             </constraints>
130             <properties/>
131           </component>
132           <component id="68e3e" class="javax.swing.JLabel" binding="labelCvLinkedinAdress">
133             <constraints>
134               <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
135             </constraints>
136             <properties>
```

```
137               <text value="Name : "/>
138             </properties>
139           </component>
140           <hspacer id="136b8">
141             <constraints>
142               <grid row="2" column="1" row-span="1" col-span="1" vsize-policy="1" hsize-policy="6" anchor="0
    " fill="1" indent="0" use-parent-layout="false"/>
143             </constraints>
144           </hspacer>
145           <component id="a9145" class="javax.swing.JLabel" binding="labelCvGithubAdress">
146             <constraints>
147               <grid row="2" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
148             </constraints>
149             <properties>
150               <text value="Department : "/>
151             </properties>
152           </component>
153           <component id="b486b" class="javax.swing.JLabel" binding="labelCvCoveringLetter">
154             <constraints>
155               <grid row="3" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
156             </constraints>
157             <properties>
158               <text value="Started Date : "/>
159             </properties>
160           </component>
161           <component id="ffa25" class="javax.swing.JLabel">
162             <constraints>
163               <grid row="4" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
164             </constraints>
165             <properties>
166               <text value="Graduated Date : "/>
167             </properties>
168           </component>
169           <grid id="866e8" binding="panelStartedDate" layout-manager="GridLayoutManager" row-count="1"
    column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
170             <margin top="0" left="0" bottom="0" right="0"/>
```

```
171             <constraints>
172               <grid row="3" column="2" row-span="1" col-span="3" vsize-policy="3" hsize-policy="3" anchor="0
    " fill="3" indent="0" use-parent-layout="false"/>
173             </constraints>
174             <properties/>
175             <border type="none"/>
176             <children/>
177           </grid>
178           <grid id="a652f" binding="panelGraduatedDate" layout-manager="GridLayoutManager" row-count="1"
    column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
179             <margin top="0" left="0" bottom="0" right="0"/>
180             <constraints>
181               <grid row="4" column="2" row-span="1" col-span="3" vsize-policy="3" hsize-policy="3" anchor="0
    " fill="3" indent="0" use-parent-layout="false"/>
182             </constraints>
183             <properties/>
184             <border type="none"/>
185             <children/>
186           </grid>
187         </children>
188       </grid>
189     </children>
190   </grid>
191 </form>
192
```

```java
import com.toedter.calendar.JDateChooser;
import entities.*;

import org.json.JSONArray;
import org.json.JSONObject;

import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.event.WindowEvent;
import java.io.*;
import java.net.HttpURLConnection;
import java.net.URL;
import java.nio.charset.StandardCharsets;
import java.text.DateFormat;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.time.LocalDateTime;
import java.time.ZoneId;
import java.util.*;
import java.util.List;

public class CandidateSchoolsForUnsavedCvs extends JFrame {
    private JPanel panelMain;
    private JPanel panelTable;
    private JScrollPane scrollPaneTable;
    private JTable tableSchool;
    private JPanel panelButtons;
    private JButton buttonCreateSchool;
    private JButton buttonEditSchool;
    private JButton buttonDeleteTextFields;
    private JPanel panelSchoolIsnformations;
    private JTextField textFieldSchoolId;
    private JLabel labelCvId;
    private JTextField textFieldName;
    private JTextField textFieldDepartment;
    private JLabel labelCvLinkedinAdress;
```

```
40      private JLabel labelCvGithubAdress;
41      private JLabel labelCvCoveringLetter;
42      private JPanel panelStartedDate;
43      private JPanel panelGraduatedDate;
44      private JButton buttonDeleteSchool;
45      private JDateChooser chooserGraduatedDate;
46      private JDateChooser chooserStartedDate;
47
48      public User user;
49      public List<School> savedSchools = new ArrayList<>();
50
51      public CandidateSchoolsForUnsavedCvs(User user) {
52          this.user = user;
53
54          this.add(panelMain);
55          this.setSize(720, 600);
56          this.dispatchEvent(new WindowEvent(this, WindowEvent.WINDOW_CLOSING));
57
58          this.setResizable(false);
59          this.setTitle("Schools for unsaved cvs");
60
61          chooserStartedDate = new JDateChooser();
62          panelStartedDate.setLayout(new GridLayout(1,0));
63          chooserStartedDate.setDateFormatString("yyyy-MM-dd");
64          panelStartedDate.add(chooserStartedDate);
65
66          chooserGraduatedDate = new JDateChooser();
67          panelGraduatedDate.setLayout(new GridLayout(1,0));
68          chooserGraduatedDate.setDateFormatString("yyyy-MM-dd");
69
70          panelGraduatedDate.add(chooserGraduatedDate);
71
72          tableSchool.addMouseListener(new MouseAdapter() {
73              @Override
74              public void mouseClicked(MouseEvent e) {
75
76                  deleteTextFields();
77
78                  DefaultTableModel model = (DefaultTableModel) tableSchool.getModel();
```

```
 79
 80                 int selectedRowIndex = tableSchool.getSelectedRow();
 81                 textFieldSchoolId.setText(model.getValueAt(selectedRowIndex,0).toString());
 82                 textFieldName.setText(model.getValueAt(selectedRowIndex,1).toString());
 83                 textFieldDepartment.setText(model.getValueAt(selectedRowIndex,2).toString());
 84
 85                 Date startedDate= null;
 86                 try {
 87                     startedDate = new SimpleDateFormat("yyyy-MM-dd").parse((String) model.getValueAt(
    selectedRowIndex,3));
 88                 } catch (ParseException ex) {
 89                     ex.printStackTrace();
 90                 }
 91
 92                 chooserStartedDate.setDate(startedDate);
 93
 94
 95                 if (model.getValueAt(selectedRowIndex,4)!=null){
 96                     Date graduatedDate= null;
 97                     try {
 98                         graduatedDate = new SimpleDateFormat("yyyy-MM-dd").parse((String) model.getValueAt(
    selectedRowIndex,4));
 99                     } catch (ParseException ex) {
100                         ex.printStackTrace();
101                     }
102                     chooserGraduatedDate.setDate(graduatedDate);
103                 }
104
105
106             }
107         });
108
109         buttonCreateSchool.addActionListener(e -> {
110                 addSchool();
111         });
112
113         buttonDeleteSchool.addActionListener(e -> {
114             try {
115                 deleteSchool();
```

```
116             } catch (Exception ex) {
117                 JOptionPane.showMessageDialog(this, ex.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
118             }
119         });
120 
121         buttonEditSchool.addActionListener(e -> {
122             try {
123                 updateSchool();
124             } catch (Exception ex) {
125                 JOptionPane.showMessageDialog(this, ex.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
126             }
127         });
128 
129         buttonDeleteTextFields.addActionListener(e -> deleteTextFields());
130     }
131 
132     private void addSchool(){
133         School school = new School();
134 
135         if (savedSchools.size() == 0)
136             school.id = 1;
137         else
138             school.id = savedSchools.get(savedSchools.size()-1).id + 1;
139 
140         school.name = textFieldName.getText();
141         school.department = textFieldDepartment.getText();
142 
143         DateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");
144 
145         String strDate = dateFormat.format(chooserStartedDate.getDate());
146         school.startedDate = strDate;
147 
148         if (chooserGraduatedDate.getDate() != null){
149             String graduatedDate = dateFormat.format(chooserGraduatedDate.getDate());
150             school.graduatedDate = graduatedDate;
151         }
152 
153 
154         if (chooserGraduatedDate.getDate() != null)
```

```
155             school.isGraduated = true;
156         else
157             school.isGraduated = false;
158 
159         savedSchools.add(school);
160 
161         assignTable();
162         deleteTextFields();
163     }
164 
165 
166     private void updateSchool(){
167         if (textFieldSchoolId.getText().equals(""))
168             JOptionPane.showMessageDialog(this,"You didn't select any school","Error",JOptionPane.
    ERROR_MESSAGE);
169         else{
170 
171             School foundSchool = new School();
172 
173             for (var school : savedSchools){
174                 if (school.id == Integer.parseInt(textFieldSchoolId.getText()))
175                     foundSchool = school;
176             }
177 
178 
179             foundSchool.name = textFieldName.getText();
180             foundSchool.department = textFieldDepartment.getText();
181 
182             DateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");
183 
184             String strDate = dateFormat.format(chooserStartedDate.getDate());
185             foundSchool.startedDate = strDate;
186 
187             if (chooserGraduatedDate.getDate() != null && foundSchool.graduatedDate == null){
188                 String graduatedDate = dateFormat.format(chooserGraduatedDate.getDate());
189                 foundSchool.graduatedDate = graduatedDate;
190             }
191 
192 
```

```
193             if (foundSchool.graduatedDate != null)
194                 foundSchool.isGraduated = true;
195             else
196                 foundSchool.isGraduated = false;
197 
198             assignTable();
199             deleteTextFields();
200         }
201     }
202 
203     private void assignTable(){
204 
205         String[] columns = new String[] {
206                 "Id","Name", "Department", "Started Date" , "Graduated Date" , "Is Graduated"
207         };
208 
209         Object[][] data = new Object[savedSchools.size()][6];
210 
211         for(var i = 0; i < savedSchools.size(); i++){
212             data[i][0] = savedSchools.get(i).id;
213             data[i][1] = savedSchools.get(i).name;
214             data[i][2] = savedSchools.get(i).department;
215             data[i][3] = savedSchools.get(i).startedDate;
216             if (savedSchools.get(i).isGraduated){
217                 data[i][4] = savedSchools.get(i).graduatedDate;
218                 data[i][5] = "Mezun olmuş";
219             }
220             else
221             data[i][5] = "Mezun olmamış";
222         }
223 
224         DefaultTableModel model = new DefaultTableModel(data,columns){
225             @Override
226             public boolean isCellEditable(int row, int column) {
227                 return false;
228             }
229         };
230 
231         tableSchool.setModel(model);
```

```
232     }
233
234     private void deleteSchool() {
235         if (textFieldSchoolId.getText().equals(""))
236             JOptionPane.showMessageDialog(this,"You didn't select any school","Error",JOptionPane.
    ERROR_MESSAGE);
237         else{
238             savedSchools.removeIf(school -> school.id == Integer.parseInt(textFieldSchoolId.getText()));
239             assignTable();
240             deleteTextFields();
241         }
242     }
243
244     public void deleteTextFields(){
245         textFieldSchoolId.setText("");
246         textFieldName.setText("");
247         textFieldDepartment.setText("");
248
249         Date dateNow = null;
250         String currentDate = new SimpleDateFormat("yyyy-mm-dd").format(Calendar.getInstance().getTime());
251         try {
252             dateNow=new SimpleDateFormat("yyyy-mm-dd").parse(currentDate);
253         } catch (ParseException e) {
254             e.printStackTrace();
255         }
256         chooserStartedDate.setDate(dateNow);
257         chooserGraduatedDate.setDate(dateNow);
258
259     }
260
261
262     public void main(String[] args){
263         try {
264             UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
265         } catch (ClassNotFoundException e) {
266             e.printStackTrace();
267         } catch (InstantiationException e) {
268             e.printStackTrace();
269         } catch (IllegalAccessException e) {
```

```
            e.printStackTrace();
        } catch (UnsupportedLookAndFeelException e) {
            e.printStackTrace();
        }

        //Set UI Thread to another thread
        SwingUtilities.invokeLater(() -> {
            CandidateSchoolsForUnsavedCvs gui = new CandidateSchoolsForUnsavedCvs(user);
            gui.setVisible(true);
        });

    }


}

```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<form xmlns="http://www.intellij.com/uidesigner/form/" version="1" bind-to-class="CandidateJobHistoriesForSavedCvs">
  <grid id="27dc6" binding="panelMain" layout-manager="GridLayoutManager" row-count="2" column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
    <margin top="10" left="10" bottom="10" right="10"/>
    <constraints>
      <xy x="20" y="20" width="719" height="400"/>
    </constraints>
    <properties/>
    <border type="none"/>
    <children>
      <grid id="ca8db" binding="panelTable" layout-manager="GridLayoutManager" row-count="3" column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
        <margin top="0" left="0" bottom="0" right="0"/>
        <constraints>
          <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
        </constraints>
        <properties/>
        <border type="none"/>
        <children>
          <scrollpane id="93bd8" binding="scrollPaneTable">
            <constraints>
              <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="7" hsize-policy="7" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties/>
            <border type="none"/>
            <children>
              <component id="711a1" class="javax.swing.JTable" binding="tableJobHistories">
                <constraints/>
                <properties>
                  <minimumSize width="30" height="50"/>
                  <preferredSize width="150" height="340"/>
                  <requestFocusEnabled value="false"/>
                </properties>
              </component>
            </children>
```

```
35            </scrollpane>
36            <grid id="18167" layout-manager="GridLayoutManager" row-count="1" column-count="1" same-size-
  horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
37              <margin top="0" left="0" bottom="0" right="0"/>
38              <constraints>
39                <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0"
   fill="3" indent="0" use-parent-layout="false"/>
40              </constraints>
41              <properties/>
42              <border type="none"/>
43              <children/>
44            </grid>
45            <grid id="c8102" binding="panelButtons" layout-manager="GridLayoutManager" row-count="1" column-
  count="4" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
46              <margin top="0" left="0" bottom="0" right="0"/>
47              <constraints>
48                <grid row="2" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0"
   fill="3" indent="0" use-parent-layout="false"/>
49              </constraints>
50              <properties/>
51              <border type="none"/>
52              <children>
53                <component id="17185" class="javax.swing.JButton" binding="buttonCreateJobHistories">
54                  <constraints>
55                    <grid row="0" column="1" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor
  ="0" fill="1" indent="0" use-parent-layout="false"/>
56                  </constraints>
57                  <properties>
58                    <text value="Create Job Histories"/>
59                  </properties>
60                </component>
61                <component id="36dc6" class="javax.swing.JButton" binding="buttonEditJobHistories">
62                  <constraints>
63                    <grid row="0" column="2" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor
  ="0" fill="1" indent="0" use-parent-layout="false"/>
64                  </constraints>
65                  <properties>
66                    <text value="Edit Job Histories"/>
67                  </properties>
```

```
68                </component>
69                <component id="8c130" class="javax.swing.JButton" binding="buttonDeleteSelectedJobHistories">
70                  <constraints>
71                    <grid row="0" column="3" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3"
   anchor="0" fill="1" indent="0" use-parent-layout="false"/>
72                  </constraints>
73                  <properties>
74                    <text value="Delete Selected Job Histories"/>
75                  </properties>
76                </component>
77                <component id="47656" class="javax.swing.JButton" binding="buttonDeleteTextFields">
78                  <constraints>
79                    <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3"
   anchor="0" fill="1" indent="0" use-parent-layout="false"/>
80                  </constraints>
81                  <properties>
82                    <text value="Refresh Page"/>
83                  </properties>
84                </component>
85              </children>
86            </grid>
87          </children>
88        </grid>
89        <grid id="d8b7b" binding="panelJobSchoolsInformations" layout-manager="GridLayoutManager" row-count="5
   " column-count="5" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
90          <margin top="0" left="0" bottom="0" right="0"/>
91          <constraints>
92            <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0"
   fill="3" indent="0" use-parent-layout="false"/>
93          </constraints>
94          <properties/>
95          <border type="none"/>
96          <children>
97            <component id="90581" class="javax.swing.JTextField" binding="textFieldJobHistoryId">
98              <constraints>
99                <grid row="0" column="2" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8
   " fill="1" indent="0" use-parent-layout="false">
100               <preferred-size width="150" height="-1"/>
101             </grid>
```

```
102             </constraints>
103             <properties>
104               <editable value="false"/>
105               <enabled value="false"/>
106             </properties>
107           </component>
108           <component id="68064" class="javax.swing.JLabel" binding="labelCvId">
109             <constraints>
110               <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
111             </constraints>
112             <properties>
113               <text value="Id : "/>
114             </properties>
115           </component>
116           <component id="d73f7" class="javax.swing.JTextField" binding="textFieldCompanyName">
117             <constraints>
118               <grid row="1" column="2" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8
    " fill="1" indent="0" use-parent-layout="false">
119                 <preferred-size width="150" height="-1"/>
120               </grid>
121             </constraints>
122             <properties/>
123           </component>
124           <component id="b5105" class="javax.swing.JLabel">
125             <constraints>
126               <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
127             </constraints>
128             <properties>
129               <text value="Company Name : "/>
130             </properties>
131           </component>
132           <hspacer id="e2082">
133             <constraints>
134               <grid row="2" column="1" row-span="1" col-span="1" vsize-policy="1" hsize-policy="6" anchor="0
    " fill="1" indent="0" use-parent-layout="false"/>
135             </constraints>
136           </hspacer>
```

```
137           <component id="47ae3" class="javax.swing.JLabel">
138             <constraints>
139               <grid row="2" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
140             </constraints>
141             <properties>
142               <text value="Position : "/>
143             </properties>
144           </component>
145           <component id="fc5cf" class="javax.swing.JLabel">
146             <constraints>
147               <grid row="3" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
148             </constraints>
149             <properties>
150               <text value="Started Date : "/>
151             </properties>
152           </component>
153           <component id="32736" class="javax.swing.JLabel">
154             <constraints>
155               <grid row="4" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
156             </constraints>
157             <properties>
158               <text value="Finished Date : "/>
159             </properties>
160           </component>
161           <grid id="464ba" binding="panelStartedDate" layout-manager="GridLayoutManager" row-count="1"
    column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
162             <margin top="0" left="0" bottom="0" right="0"/>
163             <constraints>
164               <grid row="3" column="2" row-span="1" col-span="3" vsize-policy="3" hsize-policy="3" anchor="0
    " fill="3" indent="0" use-parent-layout="false"/>
165             </constraints>
166             <properties/>
167             <border type="none"/>
168             <children/>
169           </grid>
170           <grid id="b529f" binding="panelFinishedDate" layout-manager="GridLayoutManager" row-count="1"
```

```
column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
            <margin top="0" left="0" bottom="0" right="0"/>
            <constraints>
              <grid row="4" column="2" row-span="1" col-span="3" vsize-policy="3" hsize-policy="3" anchor="0
" fill="3" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties/>
            <border type="none"/>
            <children/>
          </grid>
          <component id="bad35" class="javax.swing.JComboBox" binding="comboBoxPositions">
            <constraints>
              <grid row="2" column="2" row-span="1" col-span="3" vsize-policy="0" hsize-policy="2" anchor="8
" fill="1" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties/>
          </component>
        </children>
      </grid>
    </children>
  </grid>
</form>
```

```java
import com.toedter.calendar.JDateChooser;
import entities.*;
import org.json.JSONArray;
import org.json.JSONObject;

import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.event.WindowEvent;
import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.nio.charset.StandardCharsets;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.ZoneId;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.Scanner;

public class CandidateJobHistoriesForSavedCvs extends JFrame {

    private JPanel panelMain;
    private JPanel panelTable;
    private JScrollPane scrollPaneTable;
    private JTable tableJobHistories;
    private JPanel panelButtons;
    private JButton buttonCreateJobHistories;
    private JButton buttonEditJobHistories;
    private JButton buttonDeleteSelectedJobHistories;
    private JButton buttonDeleteTextFields;
```

```
40      private JPanel panelJobSchoolsInformations;
41      private JTextField textFieldJobHistoryId;
42      private JLabel labelCvId;
43      private JTextField textFieldCompanyName;
44      private JPanel panelStartedDate;
45      private JPanel panelFinishedDate;
46      private JComboBox comboBoxPositions;
47      private JDateChooser chooserStartedDate;
48      private JDateChooser chooserFinishedDate;
49
50      private List<JobPosition> jobPositions = new ArrayList<>();
51      private List<JobHistory> jobHistories = new ArrayList<>();
52
53      public User user;
54      public String cvId;
55
56      public CandidateJobHistoriesForSavedCvs(User user,String cvId){
57
58          this.user = user;
59          this.cvId = cvId;
60
61          this.add(panelMain);
62          this.setSize(720, 600);
63          this.dispatchEvent(new WindowEvent(this, WindowEvent.WINDOW_CLOSING));
64          this.setResizable(false);
65
66          this.setTitle("Job histories for saved cvs");
67
68          panelStartedDate.setLayout(new GridLayout(1,0));
69          chooserStartedDate = new JDateChooser();
70          chooserStartedDate.setDateFormatString("yyyy-MM-dd");
71          chooserStartedDate.setName("chooserStartedDate");
72          panelStartedDate.add(chooserStartedDate);
73
74          panelFinishedDate.setLayout(new GridLayout(1,0));
75          chooserFinishedDate = new JDateChooser();
76          chooserFinishedDate.setDateFormatString("yyyy-MM-dd");
77          chooserFinishedDate.setName("chooserFinishedDate");
78          panelFinishedDate.add(chooserFinishedDate);
```

```
 79
 80          buttonCreateJobHistories.addActionListener(e-> {
 81              try {
 82                  addJobHistory();
 83              }
 84              catch (Exception exception){
 85                  JOptionPane.showMessageDialog(this, exception.getMessage());
 86              }
 87          });
 88
 89          buttonEditJobHistories.addActionListener(e-> {
 90              try {
 91                  updateJobHistory();
 92              }
 93              catch (Exception exception){
 94                  JOptionPane.showMessageDialog(this, exception.getMessage());
 95              }
 96          });
 97
 98          buttonDeleteSelectedJobHistories.addActionListener(e-> {
 99              try {
100                  deleteJobHistory();
101              }
102              catch (Exception exception){
103                  JOptionPane.showMessageDialog(this, exception.getMessage());
104              }
105          });
106
107          try {
108              jobPositions = getJobPositionsFromApi();
109
110              DefaultComboBoxModel model = new DefaultComboBoxModel();
111              for (var jobPosition : jobPositions){
112                  model.addElement(new ComboKeyValue(jobPosition.name, String.valueOf(jobPosition.id)));
113              }
114
115              comboBoxPositions.setModel(model);
116
117          } catch (Exception e) {
```

```
118             JOptionPane.showMessageDialog(this,e.getMessage());
119         }
120 
121         tableJobHistories.addMouseListener(new MouseAdapter() {
122             @Override
123             public void mouseClicked(MouseEvent e) {
124 
125                 deleteTextFields();
126 
127                 DefaultTableModel model = (DefaultTableModel) tableJobHistories.getModel();
128 
129                 int selectedRowIndex = tableJobHistories.getSelectedRow();
130                 textFieldJobHistoryId.setText(model.getValueAt(selectedRowIndex,0).toString());
131                 textFieldCompanyName.setText(model.getValueAt(selectedRowIndex,1).toString());
132 
133 
134                 for (var jobPosition : jobPositions){
135                     if (jobPosition.name.equals(model.getValueAt(selectedRowIndex,2).toString())){
136                         comboBoxPositions.setSelectedIndex(jobPosition.id-1);
137                     }
138                 }
139 
140                 Date startedDate= null;
141                 try {
142                     startedDate = new SimpleDateFormat("yyyy-MM-dd").parse((String) model.getValueAt(
    selectedRowIndex,3));
143                 } catch (ParseException ex) {
144                     ex.printStackTrace();
145                 }
146 
147                 chooserStartedDate.setDate(startedDate);
148 
149 
150                 if (model.getValueAt(selectedRowIndex,4)!=null){
151                     Date finishedDate= null;
152                     try {
153                         finishedDate = new SimpleDateFormat("yyyy-MM-dd").parse((String) model.getValueAt(
    selectedRowIndex,4));
154                     } catch (ParseException ex) {
```

```
155                         ex.printStackTrace();
156                     }
157                     chooserFinishedDate.setDate(finishedDate);
158                 }
159 
160 
161             }
162         });
163 
164         buttonDeleteTextFields.addActionListener(e -> deleteTextFields());
165 
166         assignTable();
167     }
168 
169     private void deleteJobHistory() throws Exception {
170         if (textFieldJobHistoryId.getText().equals(""))
171             JOptionPane.showMessageDialog(this, "You didn't select any job history");
172         else {
173 
174             URL url = new URL("http://localhost:8080/api/jobHistories/deleteJobHistory?jobHistoryId=" +
    textFieldJobHistoryId.getText());
175             HttpURLConnection httpCon = (HttpURLConnection) url.openConnection();
176             httpCon.setDoOutput(true);
177             httpCon.setRequestMethod("DELETE");
178             httpCon.connect();
179 
180             InputStream inputStream;
181 
182             if (httpCon.getResponseCode() != 200)
183                 inputStream = httpCon.getErrorStream();
184             else
185                 inputStream = httpCon.getInputStream();
186 
187             String response = "";
188             String line;
189             BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
190 
191             while ((line = br.readLine()) != null) {
192                 response += line;
```

```
193             }
194 
195             JSONObject returnObject = new JSONObject(response);
196 
197             if (returnObject.getBoolean("success")) {
198                 JOptionPane.showMessageDialog(this, returnObject.getString("message"), "Information",
    JOptionPane.INFORMATION_MESSAGE);
199                 assignTable();
200                 deleteTextFields();
201             } else
202                 JOptionPane.showMessageDialog(this, returnObject.getString("message"), "Error", JOptionPane.
    ERROR_MESSAGE);
203         }
204 
205     }
206 
207     private void assignTable(){
208 
209         String[] columns = new String[] {
210                 "Id","Company Name", "Position Name", "Started Date" , "Finished Date" , "Is Finished"
211         };
212 
213         List<JobHistory> jobHistories = new ArrayList<>();
214 
215         try {
216             jobHistories = getJobHistoriesFromApi();
217         } catch (Exception e) {
218             JOptionPane.showMessageDialog(this,e.getMessage());
219         }
220 
221         //actual data for the table in a 2d array
222         Object[][] data = new Object[jobHistories.size()][6];
223 
224         for(var i = 0; i < jobHistories.size(); i++){
225             data[i][0] = jobHistories.get(i).id;
226             data[i][1] = jobHistories.get(i).companyName;
227             data[i][2] = jobHistories.get(i).jobPosition.name;
228             data[i][3] = jobHistories.get(i).startedDate;
229             if (jobHistories.get(i).finishedDate!=null)
```

```
230                 data[i][4] = jobHistories.get(i).finishedDate;
231             if (jobHistories.get(i).isFinished)
232                 data[i][5] = "İşten ayrılmış";
233             else
234                 data[i][5] = "İşten ayrılmamış";
235         }
236 
237         DefaultTableModel model = new DefaultTableModel(data,columns){
238             @Override
239             public boolean isCellEditable(int row, int column) {
240                 return false;
241             }
242         };
243 
244         tableJobHistories.setModel(model);
245     }
246 
247     private void deleteTextFields() {
248         textFieldCompanyName.setText("");
249         textFieldJobHistoryId.setText("");
250 
251 
252         ZoneId defaultZoneId = ZoneId.systemDefault();
253         SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd");
254         Date date = Date.from(LocalDate.now().atStartOfDay(defaultZoneId).toInstant());
255         formatter.format(date);
256 
257         chooserStartedDate.setDate(date);
258         chooserFinishedDate.setDate(date);
259     }
260 
261     private void updateJobHistory() throws Exception{
262         if (textFieldJobHistoryId.getText().equals(""))
263             throw new NullPointerException("Herhangi bir okul işaretlemediniz");
264 
265         JSONObject obj = new JSONObject();
266         obj.put("id", textFieldJobHistoryId.getText());
267         obj.put("companyName", textFieldCompanyName.getText());
268 
```

```
269         JSONObject cvJson = new JSONObject();
270         cvJson.put("id",cvId);
271 
272         obj.put("cv",cvJson);
273 
274         for (var jobPosition : jobPositions){
275             if (jobPosition.name == comboBoxPositions.getSelectedItem().toString()){
276                 JSONObject jobPositionJson = new JSONObject();
277                 jobPositionJson.put("id",jobPosition.id);
278                 obj.put("jobPosition",jobPositionJson);
279             }
280         }
281 
282         JSONObject candidate = new JSONObject();
283         candidate.put("id", user.id);
284         cvJson.put("candidate",candidate);
285 
286         var startedDate = convertToLocalDateTimeViaInstant(chooserStartedDate.getDate());
287         obj.put("startedDate",startedDate);
288 
289         if (chooserFinishedDate.getDate()!=null){
290             var finishedDate = convertToLocalDateTimeViaInstant(chooserFinishedDate.getDate());
291             obj.put("finishedDate",finishedDate);
292         }
293         else{
294             obj.put("finishedDate",JSONObject.NULL);
295         }
296 
297 
298 
299         URL url = new URL("http://localhost:8080/api/jobHistories/updateJobHistory");
300         HttpURLConnection http = (HttpURLConnection)url.openConnection();
301         http.setRequestMethod("PUT");
302         http.setDoOutput(true);
303         http.setRequestProperty("Content-Type", "application/json");
304 
305 
306         String data = obj.toString();
307 
```

```
308         byte[] out = data.getBytes(StandardCharsets.UTF_8);
309
310         OutputStream stream = http.getOutputStream();
311         stream.write(out);
312
313         InputStream inputStream;
314
315         if (http.getResponseCode() != 200)
316             inputStream = http.getErrorStream();
317         else
318             inputStream = http.getInputStream();
319
320         String response = "";
321         String line;
322         BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
323
324         while ((line = br.readLine()) != null) {
325             response += line;
326         }
327
328         JSONObject returnObject = new JSONObject(response);
329
330         if (returnObject.get("success").equals(true)){
331             JOptionPane.showMessageDialog(this,returnObject.get("message"));
332             http.disconnect();
333             assignTable();
334         }
335
336         else{
337             http.disconnect();
338             JOptionPane.showMessageDialog(this,returnObject.get("message"));
339         }
340     }
341
342     public LocalDateTime convertToLocalDateTimeViaInstant(Date dateToConvert) {
343         return dateToConvert.toInstant()
344                 .atZone(ZoneId.systemDefault())
345                 .toLocalDateTime();
346     }
```

```
347
348     private void addJobHistory() throws Exception{
349         JSONObject obj = new JSONObject();
350         obj.put("id", textFieldJobHistoryId.getText());
351         obj.put("companyName", textFieldCompanyName.getText());
352
353         JSONObject cvJson = new JSONObject();
354         cvJson.put("id",cvId);
355
356         obj.put("cv",cvJson);
357
358         for (var jobPosition : jobPositions){
359             if (jobPosition.name == comboBoxPositions.getSelectedItem().toString()){
360                 JSONObject jobPositionJson = new JSONObject();
361                 jobPositionJson.put("id",jobPosition.id);
362                 obj.put("jobPosition",jobPositionJson);
363             }
364         }
365
366         JSONObject candidate = new JSONObject();
367         candidate.put("id", user.id);
368         cvJson.put("candidate",candidate);
369
370         var startedDate = convertToLocalDateTimeViaInstant(chooserStartedDate.getDate());
371         obj.put("startedDate",startedDate);
372
373         if (chooserFinishedDate.getDate()!=null){
374             var finishedDate = convertToLocalDateTimeViaInstant(chooserFinishedDate.getDate());
375             obj.put("finishedDate",finishedDate);
376         }
377
378         obj.put("finishedDate",JSONObject.NULL);
379
380
381         URL url = new URL("http://localhost:8080/api/jobHistories/addJobHistory");
382         HttpURLConnection http = (HttpURLConnection)url.openConnection();
383         http.setRequestMethod("POST");
384         http.setDoOutput(true);
385         http.setRequestProperty("Content-Type", "application/json");
```

```
386
387
388         String data = obj.toString();
389
390         byte[] out = data.getBytes(StandardCharsets.UTF_8);
391
392         OutputStream stream = http.getOutputStream();
393         stream.write(out);
394
395         InputStream inputStream;
396
397         if (http.getResponseCode() != 200)
398             inputStream = http.getErrorStream();
399         else
400             inputStream = http.getInputStream();
401
402         String response = "";
403         String line;
404         BufferedReader br = new BufferedReader(new InputStreamReader(inputStream));
405
406         while ((line = br.readLine()) != null) {
407             response += line;
408         }
409
410         JSONObject returnObject = new JSONObject(response);
411
412         if (returnObject.get("success").equals(true)){
413             JOptionPane.showMessageDialog(this,returnObject.get("message"));
414             http.disconnect();
415             assignTable();
416             deleteTextFields();
417         }
418
419         else{
420             http.disconnect();
421             JOptionPane.showMessageDialog(this,returnObject.get("message"));
422         }
423     }
424
```

```
425     private List<JobHistory> getJobHistoriesFromApi() throws Exception{
426         ArrayList<JobHistory> jobHistories = new ArrayList<>();
427 
428         URL getJobPositions = new URL("http://localhost:8080/api/jobHistories/getAllByCvId?cvId="+cvId);
429         HttpURLConnection conn = (HttpURLConnection) getJobPositions.openConnection();
430         conn.setRequestMethod("GET");
431         conn.connect();
432         int responseCode = conn.getResponseCode();
433 
434         if (responseCode != 200) {
435             throw new RuntimeException("HttpResponseCode: " + responseCode);
436         } else {
437 
438             String inline = "";
439             Scanner scanner = new Scanner(getJobPositions.openStream());
440 
441             //Write all the JSON data into a string using a scanner
442             while (scanner.hasNext()) {
443                 inline += scanner.nextLine();
444             }
445 
446             //Close the scanner
447             scanner.close();
448 
449             String jsonString = inline ;
450             JSONObject obj = new JSONObject(jsonString);
451 
452             JSONArray arr = obj.getJSONArray("data");
453             for (int i = 0; i < arr.length(); i++)
454             {
455                 JobHistory jobHistory = new JobHistory();
456                 jobHistory.companyName = arr.getJSONObject(i).getString("companyName");
457                 jobHistory.id = arr.getJSONObject(i).getInt("id");
458 
459                 JobPosition jobPosition = new JobPosition();
460                 jobPosition.name = arr.getJSONObject(i).getJSONObject("jobPosition").getString("title");
461                 jobPosition.id = arr.getJSONObject(i).getJSONObject("jobPosition").getInt("id");
462                 jobHistory.jobPosition = jobPosition;
463 
```

```
464                 jobHistory.startedDate = arr.getJSONObject(i).getString("startedDate");
465
466                 if (arr.getJSONObject(i).isNull("finishedDate"))
467                     jobHistory.isFinished = false;
468
469                 else
470                 {
471                     jobHistory.finishedDate = arr.getJSONObject(i).getString("finishedDate");
472                     jobHistory.isFinished = true;
473                 }
474
475
476
477
478                 jobHistories.add(jobHistory);
479             }
480
481             return jobHistories;
482         }
483     }
484
485     private List<JobPosition> getJobPositionsFromApi() throws Exception{
486         ArrayList<JobPosition> jobPositions = new ArrayList<>();
487
488         URL getJobPositions = new URL("http://localhost:8080/api/jobPositions/getAll");
489         HttpURLConnection conn = (HttpURLConnection) getJobPositions.openConnection();
490         conn.setRequestMethod("GET");
491         conn.connect();
492         int responseCode = conn.getResponseCode();
493
494         if (responseCode != 200) {
495             throw new RuntimeException("HttpResponseCode: " + responseCode);
496         } else {
497
498             String inline = "";
499             Scanner scanner = new Scanner(getJobPositions.openStream());
500
501             //Write all the JSON data into a string using a scanner
502             while (scanner.hasNext()) {
```

```
503                 inline += scanner.nextLine();
504             }
505 
506             //Close the scanner
507             scanner.close();
508 
509             String jsonString = inline ;
510             JSONObject obj = new JSONObject(jsonString);
511 
512             JSONArray arr = obj.getJSONArray("data");
513             for (int i = 0; i < arr.length(); i++)
514             {
515                 JobPosition jobPosition = new JobPosition();
516                 jobPosition.name = arr.getJSONObject(i).getString("title");
517                 jobPosition.id = arr.getJSONObject(i).getInt("id");
518 
519                 jobPositions.add(jobPosition);
520             }
521 
522             return jobPositions;
523         }
524     }
525 
526     public void main(String[] args){
527         try {
528             UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
529         } catch (ClassNotFoundException e) {
530             e.printStackTrace();
531         } catch (InstantiationException e) {
532             e.printStackTrace();
533         } catch (IllegalAccessException e) {
534             e.printStackTrace();
535         } catch (UnsupportedLookAndFeelException e) {
536             e.printStackTrace();
537         }
538 
539         //Set UI Thread to another thread
540         SwingUtilities.invokeLater(() -> {
541             CandidateJobHistoriesForSavedCvs gui = new CandidateJobHistoriesForSavedCvs(user,cvId);
```

```
542                 gui.setVisible(true);
543             });
544 
545         }
546 
547 
548 }
549 
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<form xmlns="http://www.intellij.com/uidesigner/form/" version="1" bind-to-class="CandidateJobHistoriesForUnsavedCvs">
  <grid id="27dc6" binding="panelMain" layout-manager="GridLayoutManager" row-count="2" column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
    <margin top="10" left="10" bottom="10" right="10"/>
    <constraints>
      <xy x="20" y="20" width="723" height="400"/>
    </constraints>
    <properties/>
    <border type="none"/>
    <children>
      <grid id="cfa79" binding="panelTable" layout-manager="GridLayoutManager" row-count="3" column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
        <margin top="0" left="0" bottom="0" right="0"/>
        <constraints>
          <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
        </constraints>
        <properties/>
        <border type="none"/>
        <children>
          <scrollpane id="56c68" binding="scrollPaneTable">
            <constraints>
              <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="7" hsize-policy="7" anchor="0" fill="3" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties/>
            <border type="none"/>
            <children>
              <component id="8d207" class="javax.swing.JTable" binding="tableJobHistories">
                <constraints/>
                <properties>
                  <minimumSize width="30" height="50"/>
                  <preferredSize width="150" height="340"/>
                  <requestFocusEnabled value="false"/>
                </properties>
              </component>
            </children>
```

```
35            </scrollpane>
36            <grid id="49bc3" layout-manager="GridLayoutManager" row-count="1" column-count="1" same-size-
  horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
37              <margin top="0" left="0" bottom="0" right="0"/>
38              <constraints>
39                <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0"
   fill="3" indent="0" use-parent-layout="false"/>
40              </constraints>
41              <properties/>
42              <border type="none"/>
43              <children/>
44            </grid>
45            <grid id="91af7" binding="panelButtons" layout-manager="GridLayoutManager" row-count="1" column-
  count="4" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
46              <margin top="0" left="0" bottom="0" right="0"/>
47              <constraints>
48                <grid row="2" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0"
   fill="3" indent="0" use-parent-layout="false"/>
49              </constraints>
50              <properties/>
51              <border type="none"/>
52              <children>
53                <component id="fcf1f" class="javax.swing.JButton" binding="buttonCreateJobHistories">
54                  <constraints>
55                    <grid row="0" column="1" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor
  ="0" fill="1" indent="0" use-parent-layout="false"/>
56                  </constraints>
57                  <properties>
58                    <text value="Create Job Histories"/>
59                  </properties>
60                </component>
61                <component id="e2564" class="javax.swing.JButton" binding="buttonEditJobHistories">
62                  <constraints>
63                    <grid row="0" column="2" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3" anchor
  ="0" fill="1" indent="0" use-parent-layout="false"/>
64                  </constraints>
65                  <properties>
66                    <text value="Edit Job Histories"/>
67                  </properties>
```

```
68                </component>
69                <component id="4765d" class="javax.swing.JButton" binding="buttonDeleteSelectedJobHistories">
70                  <constraints>
71                    <grid row="0" column="3" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3"
   anchor="0" fill="1" indent="0" use-parent-layout="false"/>
72                  </constraints>
73                  <properties>
74                    <text value="Delete Selected Job Histories"/>
75                  </properties>
76                </component>
77                <component id="f9d57" class="javax.swing.JButton" binding="buttonDeleteTextFields">
78                  <constraints>
79                    <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="3"
   anchor="0" fill="1" indent="0" use-parent-layout="false"/>
80                  </constraints>
81                  <properties>
82                    <text value="Refresh Page"/>
83                  </properties>
84                </component>
85              </children>
86            </grid>
87          </children>
88        </grid>
89        <grid id="72e0" binding="panelJobSchoolsInformations" layout-manager="GridLayoutManager" row-count="5"
    column-count="5" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
90          <margin top="0" left="0" bottom="0" right="0"/>
91          <constraints>
92            <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="3" hsize-policy="3" anchor="0"
   fill="3" indent="0" use-parent-layout="false"/>
93          </constraints>
94          <properties/>
95          <border type="none"/>
96          <children>
97            <component id="74708" class="javax.swing.JTextField" binding="textFieldJobHistoryId">
98              <constraints>
99                <grid row="0" column="2" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8
   " fill="1" indent="0" use-parent-layout="false">
100               <preferred-size width="150" height="-1"/>
101             </grid>
```

```
102             </constraints>
103             <properties>
104               <editable value="false"/>
105               <enabled value="false"/>
106             </properties>
107           </component>
108           <component id="74e3d" class="javax.swing.JLabel" binding="labelCvId">
109             <constraints>
110               <grid row="0" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
111             </constraints>
112             <properties>
113               <text value="Id : "/>
114             </properties>
115           </component>
116           <component id="82d3" class="javax.swing.JTextField" binding="textFieldCompanyName">
117             <constraints>
118               <grid row="1" column="2" row-span="1" col-span="3" vsize-policy="0" hsize-policy="6" anchor="8
    " fill="1" indent="0" use-parent-layout="false">
119                 <preferred-size width="150" height="-1"/>
120               </grid>
121             </constraints>
122             <properties/>
123           </component>
124           <component id="c064f" class="javax.swing.JLabel">
125             <constraints>
126               <grid row="1" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
127             </constraints>
128             <properties>
129               <text value="Company Name : "/>
130             </properties>
131           </component>
132           <hspacer id="cde5b">
133             <constraints>
134               <grid row="2" column="1" row-span="1" col-span="1" vsize-policy="1" hsize-policy="6" anchor="0
    " fill="1" indent="0" use-parent-layout="false"/>
135             </constraints>
136           </hspacer>
```

```
137           <component id="a75f4" class="javax.swing.JLabel">
138             <constraints>
139               <grid row="2" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
140             </constraints>
141             <properties>
142               <text value="Position : "/>
143             </properties>
144           </component>
145           <component id="3734a" class="javax.swing.JLabel">
146             <constraints>
147               <grid row="3" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
148             </constraints>
149             <properties>
150               <text value="Started Date : "/>
151             </properties>
152           </component>
153           <component id="8fe27" class="javax.swing.JLabel">
154             <constraints>
155               <grid row="4" column="0" row-span="1" col-span="1" vsize-policy="0" hsize-policy="0" anchor="8
    " fill="0" indent="0" use-parent-layout="false"/>
156             </constraints>
157             <properties>
158               <text value="Finished Date : "/>
159             </properties>
160           </component>
161           <grid id="47b7d" binding="panelStartedDate" layout-manager="GridLayoutManager" row-count="1"
    column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
162             <margin top="0" left="0" bottom="0" right="0"/>
163             <constraints>
164               <grid row="3" column="2" row-span="1" col-span="3" vsize-policy="3" hsize-policy="3" anchor="0
    " fill="3" indent="0" use-parent-layout="false"/>
165             </constraints>
166             <properties/>
167             <border type="none"/>
168             <children/>
169           </grid>
170           <grid id="3564f" binding="panelFinishedDate" layout-manager="GridLayoutManager" row-count="1"
```

```
column-count="1" same-size-horizontally="false" same-size-vertically="false" hgap="-1" vgap="-1">
            <margin top="0" left="0" bottom="0" right="0"/>
            <constraints>
              <grid row="4" column="2" row-span="1" col-span="3" vsize-policy="3" hsize-policy="3" anchor="0
" fill="3" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties/>
            <border type="none"/>
            <children/>
          </grid>
          <component id="ad9c2" class="javax.swing.JComboBox" binding="comboBoxPositions">
            <constraints>
              <grid row="2" column="2" row-span="1" col-span="3" vsize-policy="0" hsize-policy="2" anchor="8
" fill="1" indent="0" use-parent-layout="false"/>
            </constraints>
            <properties/>
          </component>
        </children>
      </grid>
    </children>
  </grid>
</form>
```

```java
import com.toedter.calendar.JDateChooser;
import entities.*;
import localData.CacheData;
import org.json.JSONArray;
import org.json.JSONObject;

import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.event.WindowEvent;
import java.net.HttpURLConnection;
import java.net.URL;
import java.text.DateFormat;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.ZoneId;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.Scanner;

public class CandidateJobHistoriesForUnsavedCvs extends JFrame {

    private JPanel panelMain;
    private JPanel panelTable;
    private JScrollPane scrollPaneTable;
    private JTable tableJobHistories;
    private JPanel panelButtons;
    private JButton buttonCreateJobHistories;
    private JButton buttonEditJobHistories;
    private JButton buttonDeleteSelectedJobHistories;
    private JButton buttonDeleteTextFields;
    private JPanel panelJobSchoolsInformations;
    private JTextField textFieldJobHistoryId;
    private JLabel labelCvId;
```

```
40      private JTextField textFieldCompanyName;
41      private JPanel panelStartedDate;
42      private JPanel panelFinishedDate;
43      private JComboBox comboBoxPositions;
44      private JDateChooser chooserStartedDate;
45      private JDateChooser chooserFinishedDate;
46
47      private List<JobPosition> jobPositions = new ArrayList<>();
48
49      public User user;
50
51      public CandidateJobHistoriesForUnsavedCvs(User user){
52
53          this.user = user;
54
55          this.add(panelMain);
56          this.setSize(720, 600);
57          this.dispatchEvent(new WindowEvent(this, WindowEvent.WINDOW_CLOSING));
58          this.setResizable(false);
59
60          this.setTitle("Job histories for unsaved cvs");
61
62          panelStartedDate.setLayout(new GridLayout(1,0));
63          chooserStartedDate = new JDateChooser();
64          chooserStartedDate.setDateFormatString("yyyy-MM-dd");
65          chooserStartedDate.setName("chooserStartedDate");
66          panelStartedDate.add(chooserStartedDate);
67
68          panelFinishedDate.setLayout(new GridLayout(1,0));
69          chooserFinishedDate = new JDateChooser();
70          chooserFinishedDate.setDateFormatString("yyyy-MM-dd");
71          chooserFinishedDate.setName("chooserFinishedDate");
72          panelFinishedDate.add(chooserFinishedDate);
73
74          buttonCreateJobHistories.addActionListener(e-> {
75              try {
76                  addJobHistory();
77              }
78              catch (Exception exception){
```

```
79                  JOptionPane.showMessageDialog(this, exception.getMessage());
80              }
81          });
82
83          buttonEditJobHistories.addActionListener(e-> {
84              try {
85                  updateJobHistory();
86              }
87              catch (Exception exception){
88                  JOptionPane.showMessageDialog(this, exception.getMessage(), "Error", JOptionPane.
    ERROR_MESSAGE);
89              }
90          });
91
92          buttonDeleteSelectedJobHistories.addActionListener(e-> {
93              try {
94                  deleteJobHistory();
95              }
96              catch (Exception exception){
97                  JOptionPane.showMessageDialog(this, exception.getMessage());
98              }
99          });
100
101         try {
102             jobPositions = getJobPositionsFromApi();
103
104             DefaultComboBoxModel model = new DefaultComboBoxModel();
105             for (var jobPosition : jobPositions){
106                 model.addElement(new ComboKeyValue(jobPosition.name, String.valueOf(jobPosition.id)));
107             }
108
109             comboBoxPositions.setModel(model);
110
111         } catch (Exception e) {
112             JOptionPane.showMessageDialog(this,e.getMessage());
113         }
114
115         tableJobHistories.addMouseListener(new MouseAdapter() {
116             @Override
```

```
117             public void mouseClicked(MouseEvent e) {
118
119                 deleteTextFields();
120
121                 DefaultTableModel model = (DefaultTableModel) tableJobHistories.getModel();
122
123                 int selectedRowIndex = tableJobHistories.getSelectedRow();
124                 textFieldJobHistoryId.setText(model.getValueAt(selectedRowIndex,0).toString());
125                 textFieldCompanyName.setText(model.getValueAt(selectedRowIndex,1).toString());
126
127
128                 for (var jobPosition : jobPositions){
129                     if (jobPosition.name.equals(model.getValueAt(selectedRowIndex,2).toString())){
130                         comboBoxPositions.setSelectedIndex(jobPosition.id-1);
131                     }
132                 }
133
134                 Date startedDate= null;
135                 try {
136                     startedDate = new SimpleDateFormat("yyyy-MM-dd").parse((String) model.getValueAt(
    selectedRowIndex,3));
137                 } catch (ParseException ex) {
138                     ex.printStackTrace();
139                 }
140
141                 chooserStartedDate.setDate(startedDate);
142
143
144                 if (model.getValueAt(selectedRowIndex,4)!=null){
145                     Date finishedDate= null;
146                     try {
147                         finishedDate = new SimpleDateFormat("yyyy-MM-dd").parse((String) model.getValueAt(
    selectedRowIndex,4));
148                     } catch (ParseException ex) {
149                         ex.printStackTrace();
150                     }
151                     chooserFinishedDate.setDate(finishedDate);
152                 }
153
```

```
154
155             }
156         });
157
158         buttonDeleteTextFields.addActionListener(e -> deleteTextFields());
159
160         assignTable();
161     }
162
163     private void deleteJobHistory() throws Exception {
164         if (textFieldJobHistoryId.getText().equals(""))
165             throw new NullPointerException("You didn't select any job history");
166         else {
167             CacheData.savedJobHistories.removeIf(jobHistory -> jobHistory.id == Integer.parseInt(
    textFieldJobHistoryId.getText()));
168             assignTable();
169             deleteTextFields();
170         }
171
172     }
173
174     private void assignTable(){
175
176         String[] columns = new String[] {
177                 "Id","Company Name", "Position Name", "Started Date" , "Finished Date" , "Is Finished"
178         };
179
180         //actual data for the table in a 2d array
181         Object[][] data = new Object[CacheData.savedJobHistories.size()][6];
182
183         for(var i = 0; i < CacheData.savedJobHistories.size(); i++){
184             data[i][0] = CacheData.savedJobHistories.get(i).id;
185             data[i][1] = CacheData.savedJobHistories.get(i).companyName;
186             data[i][2] = CacheData.savedJobHistories.get(i).jobPosition.name;
187             data[i][3] = CacheData.savedJobHistories.get(i).startedDate;
188             data[i][4] = CacheData.savedJobHistories.get(i).finishedDate;
189             if (CacheData.savedJobHistories.get(i).isFinished)
190                 data[i][5] = "İşten ayrılmış";
191             else
```

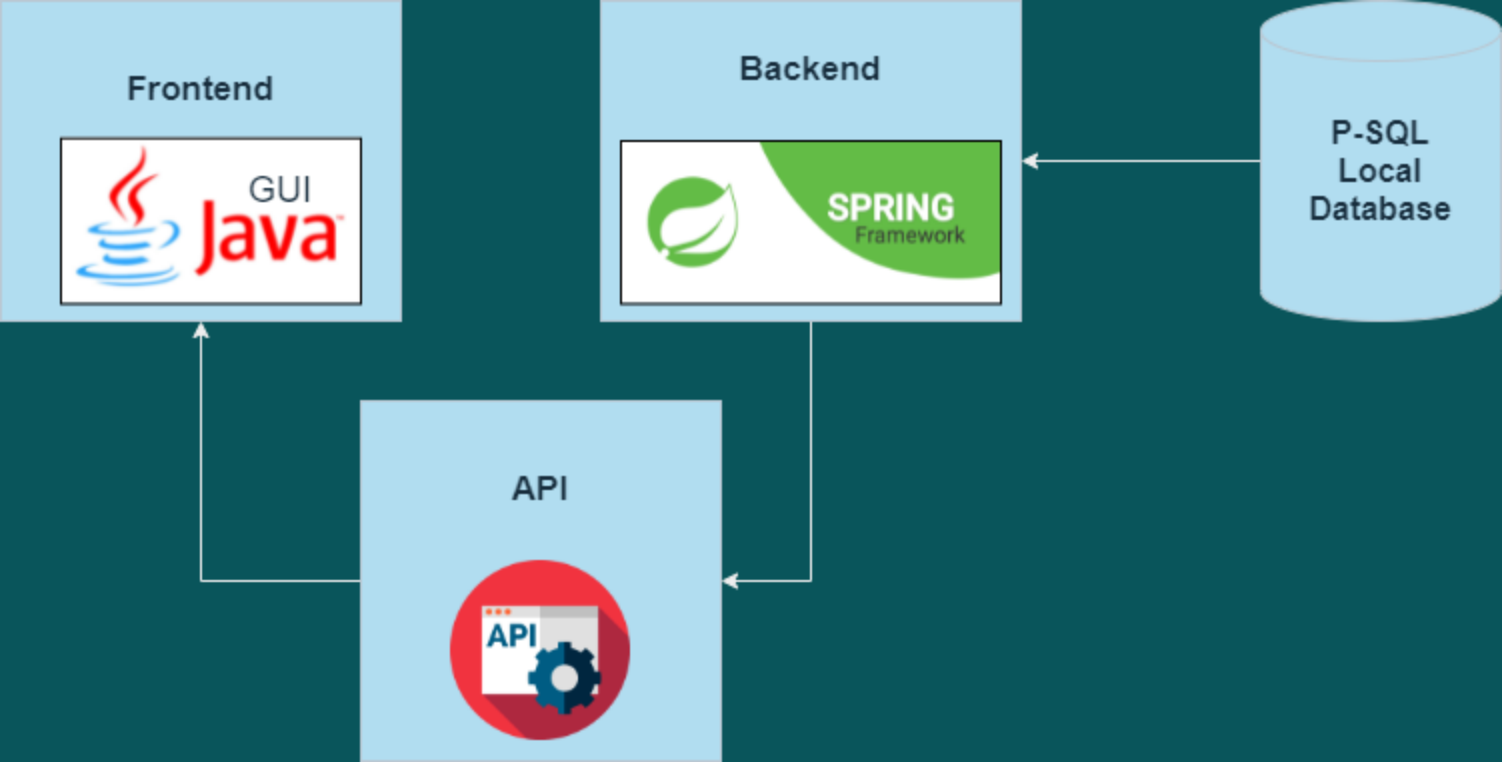
```
192                 data[i][5] = "İşten ayrılmamış";
193         }
194 
195         DefaultTableModel model = new DefaultTableModel(data,columns){
196             @Override
197             public boolean isCellEditable(int row, int column) {
198                 return false;
199             }
200         };
201 
202         tableJobHistories.setModel(model);
203     }
204 
205     private void deleteTextFields() {
206         textFieldCompanyName.setText("");
207         textFieldJobHistoryId.setText("");
208 
209 
210         ZoneId defaultZoneId = ZoneId.systemDefault();
211         SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd");
212         Date date = Date.from(LocalDate.now().atStartOfDay(defaultZoneId).toInstant());
213         formatter.format(date);
214 
215         chooserStartedDate.setDate(date);
216         chooserFinishedDate.setDate(date);
217     }
218 
219     private void updateJobHistory() throws Exception{
220         if (textFieldJobHistoryId.getText().equals(""))
221             throw new NullPointerException("Herhangi bir iş geçmişi işaretlemediniz");
222 
223         JobHistory foundJobHistory = new JobHistory();
224 
225         for (var jobHistory : CacheData.savedJobHistories){
226             if (jobHistory.id == Integer.parseInt(textFieldJobHistoryId.getText()))
227                 foundJobHistory = jobHistory;
228         }
229 
230 
```

```
231         foundJobHistory.companyName = textFieldCompanyName.getText();
232
233         DateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");
234
235         String strDate = dateFormat.format(chooserStartedDate.getDate());
236         foundJobHistory.startedDate = strDate;
237
238         if (chooserFinishedDate.getDate() != null && foundJobHistory.finishedDate == null){
239             String finishedDate = dateFormat.format(chooserFinishedDate.getDate());
240             foundJobHistory.finishedDate = finishedDate;
241         }
242
243
244         if (foundJobHistory.finishedDate != null)
245             foundJobHistory.isFinished = true;
246         else
247             foundJobHistory.isFinished = false;
248
249         assignTable();
250         deleteTextFields();
251
252     }
253
254     public LocalDateTime convertToLocalDateTimeViaInstant(Date dateToConvert) {
255         return dateToConvert.toInstant()
256                 .atZone(ZoneId.systemDefault())
257                 .toLocalDateTime();
258     }
259
260     private void addJobHistory(){
261         JobHistory jobHistory = new JobHistory();
262         jobHistory.companyName = textFieldCompanyName.getText();
263
264         if (CacheData.savedJobHistories.size() == 0)
265             jobHistory.id = 1;
266         else
267             jobHistory.id = CacheData.savedJobHistories.get(CacheData.savedJobHistories.size()-1).id + 1;
268
269         DateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");
```

```
270
271         String strDate = dateFormat.format(chooserStartedDate.getDate());
272         jobHistory.startedDate = strDate;
273
274         if (chooserFinishedDate.getDate() != null){
275             String finishedDate = dateFormat.format(chooserFinishedDate.getDate());
276             jobHistory.finishedDate = finishedDate;
277         }
278
279         if (chooserFinishedDate.getDate() != null)
280             jobHistory.isFinished = true;
281         else
282             jobHistory.isFinished = false;
283
284         for (var jobPosition : jobPositions){
285             if (jobPosition.name.equals(comboBoxPositions.getSelectedItem().toString()))
286                 jobHistory.jobPosition = jobPosition;
287         }
288
289         CacheData.savedJobHistories.add(jobHistory);
290         assignTable();
291         deleteTextFields();
292     }
293
294     private List<JobPosition> getJobPositionsFromApi() throws Exception{
295         ArrayList<JobPosition> jobPositions = new ArrayList<>();
296
297         URL getJobPositions = new URL("http://localhost:8080/api/jobPositions/getAll");
298         HttpURLConnection conn = (HttpURLConnection) getJobPositions.openConnection();
299         conn.setRequestMethod("GET");
300         conn.connect();
301         int responseCode = conn.getResponseCode();
302
303         if (responseCode != 200) {
304             throw new RuntimeException("HttpResponseCode: " + responseCode);
305         } else {
306
307             String inline = "";
308             Scanner scanner = new Scanner(getJobPositions.openStream());
```

```
309
310             //Write all the JSON data into a string using a scanner
311             while (scanner.hasNext()) {
312                 inline += scanner.nextLine();
313             }
314
315             //Close the scanner
316             scanner.close();
317
318             String jsonString = inline ;
319             JSONObject obj = new JSONObject(jsonString);
320
321             JSONArray arr = obj.getJSONArray("data");
322             for (int i = 0; i < arr.length(); i++)
323             {
324                 JobPosition jobPosition = new JobPosition();
325                 jobPosition.name = arr.getJSONObject(i).getString("title");
326                 jobPosition.id = arr.getJSONObject(i).getInt("id");
327
328                 jobPositions.add(jobPosition);
329             }
330
331             return jobPositions;
332         }
333     }
334
335     public void main(String[] args){
336         try {
337             UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
338         } catch (ClassNotFoundException e) {
339             e.printStackTrace();
340         } catch (InstantiationException e) {
341             e.printStackTrace();
342         } catch (IllegalAccessException e) {
343             e.printStackTrace();
344         } catch (UnsupportedLookAndFeelException e) {
345             e.printStackTrace();
346         }
347
```

```
348          //Set UI Thread to another thread
349          SwingUtilities.invokeLater(() -> {
350              CandidateJobHistoriesForUnsavedCvs gui = new CandidateJobHistoriesForUnsavedCvs(user);
351              gui.setVisible(true);
352          });
353
354      }
355
356
357  }
358
```


Frontend
GUI
Java
Backend
SPRING
Framework
P-SQL
Local
Database
API
API