SCHOOL OF COMPUTING

UNIVERSITI UTARA MALAYSIA

STIA1123 PROGRAMMING 2

Project Title: PIBG Election for SMK Tinggi Klang

FIRST SEMESTER SESSION 2022/2023 (A221)

Prepared By:

NAME	MATRIC NUMBER
VINCENT BEH HUA EIK	279018
POON WAI KIT	279021

Submitted to: Madam Norida bt. Muhd. Darus

Submission Date: 20th January 2023

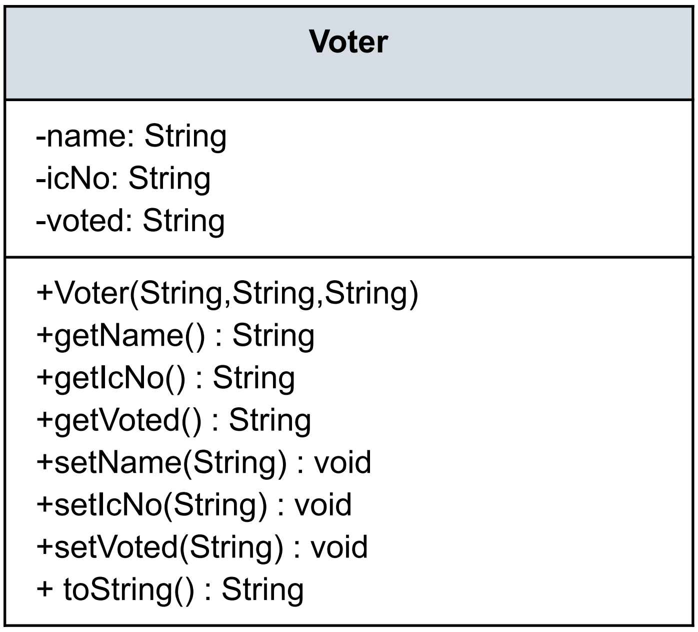
Background of the Project

The purpose of this project is to create a Guided User-Interface (GUI) application for the election process for the Parent-Teacher Association of a school or better known as Persatuan Ibu-Bapa dan Guru (PIBG) in Malaysia. This project will be specifically for the PIBG of SMK Tinggi Klang. For this application, there will be 4 main sections to the application which is candidate and voter registration, a voting section and an administrative section to allow admin actions.

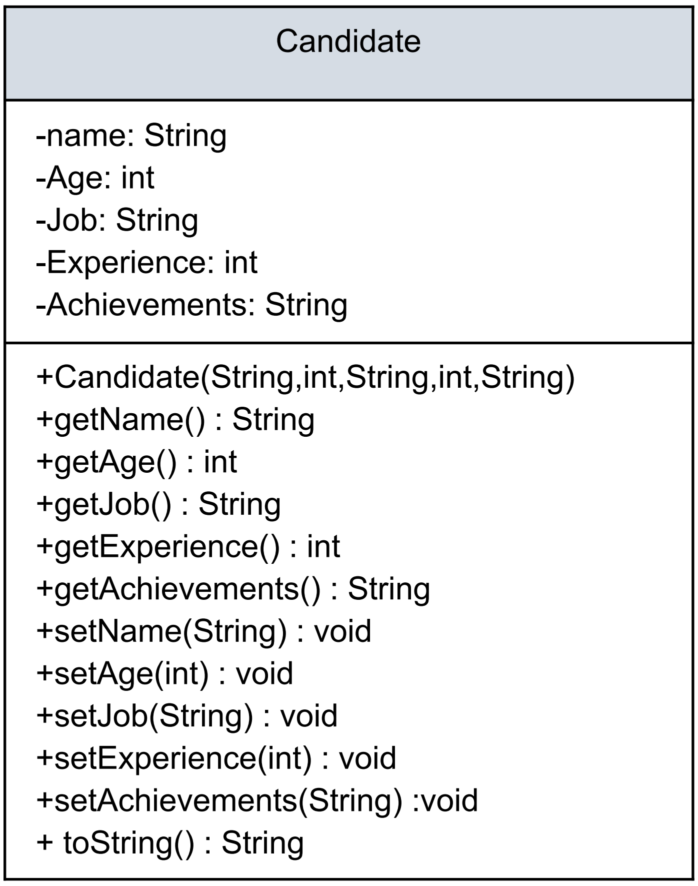
For the candidate registration section, the section will allow the competing candidates to enter their information such as their name, age, occupation, years of experience and their best related experience for the PIBG. The voter registration section allow participating voters to register their information. The voting section will be the section that allows the voters to choose their preferred candidate. The administrative section will only be for admins to make some changes during the registration period. They can also access and view the election results.

The end-goal of the project is to have the application to allow the voters to walk in and vote before leaving as the voters will be busy parents who are working. The application can also be modified to allow other schools to run the same elections while there will be a reset button to allow schools to reuse the program whenever they will need to run an election again.

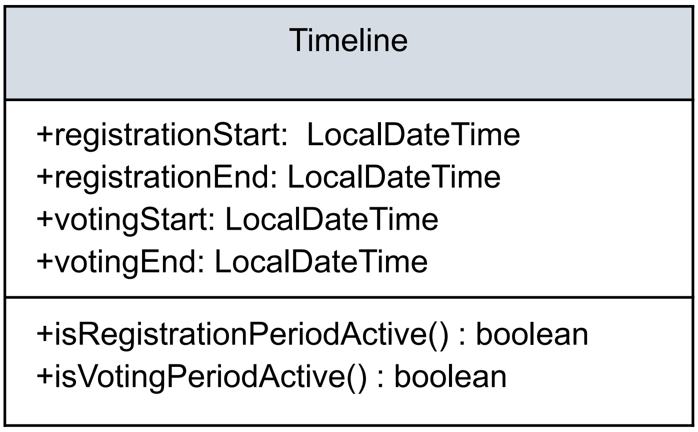
UML Class Diagrams



```
Voter
-name: String
-icNo: String
-voted: String
+Voter(String,String,String)
+getName() : String
+getIcNo() : String
+getVoted() : String
+setName(String) : void
+setIcNo(String) : void
+setVoted(String) : void
+ toString() : String
```



```
Candidate
-name: String
-Age: int
-Job: String
-Experience: int
-Achievements: String
+Candidate(String,int,String,int,String)
+getName() : String
+getAge() : int
+getJob() : String
+getExperience() : int
+getAchievements() : String
+setName(String) : void
+setAge(int) : void
+setJob(String) : void
+setExperience(int) : void
+setAchievements(String) :void
+ toString() : String
```


Timeline
+registrationStart: LocalDateTime
+registrationEnd: LocalDateTime
+votingStart: LocalDateTime
+votingEnd: LocalDateTime
+isRegistrationPeriodActive() : boolean
+isVotingPeriodActive() : boolean

Program Description

User Defined Class

The User-defined classes present in this project is according to the UML Classes above.

The Voter class contains information about a voter, including their name (as a string), IC number (also as a string), and whether or not they have voted (also as a string). The class has a constructor that takes in three strings (name, IC number, and voted status) and sets the corresponding class variables. There are also getter and setter methods for each variable, as well as a toString method that returns a string representation of the voter's information.

The Candidate class contains information about a candidate, including their name (string), age (int), job (string), experience (int), and achievements (string). The class has a constructor that takes in five parameters (name, age, job, experience, and achievements) and sets the corresponding class variables. There are also getter and setter methods for each variable, as well as a toString method that returns a string representation of the candidate's information.

The Timeline class contains four LocalDateTime variables, registrationStart, registrationEnd, votingStart and votingEnd which is set before the overall process of the election begins. The class also has two methods, isRegistrationPeriodActive() and isVotingPeriodActive(), which return a boolean indicating if the respective periods are currently active.

Array of Objects

For our application, there will be 2 array of objects initialised by using the *ArrayList* Class in a class for each of the arrays. The arrays are also static to allow access from other jFrame Forms. The reason they are inside separate classes called holders is to make it easier to manage and encapsulate the data and methods. It is also to ensure the data for both the arraylists are consistent across all the forms we are using for the program. Below is the snippet of code showing the array of objects initialised.

```
public class CandidateListHolder {

     public static ArrayList<Candidate> list = new ArrayList<>();
 }

 public class VoterListHolder {

     public static ArrayList<Voter> voterlist = new ArrayList<>();
 }
```

<u>Exception Handling</u>

There are a few instances of exception handling methods that we have applied in the program. One is setting up try-catch blocks within the code of accepting code in textfields which should be entered as integers to catch any number format exceptions that occur. We also have catch blocks for IOExceptions to catch any IO Exceptions occurring. Below are examples of said try-catch blocks.

<u>Example 1 from CandidateRegSaveActionPerformed in CandidateReg.java</u>

```
try {
     //action code from line 380-415
            } catch (NumberFormatException ex) {
                Object[] options = {"OK"};
                JOptionPane.showOptionDialog(this, "Please enter a valid
number for Age and Experience", "Error", JOptionPane.DEFAULT_OPTION,
JOptionPane.ERROR_MESSAGE, null, options, options[0]);
                txtAge.setText("");;
                txtExp.setText("");
            }
```

<u>Example 2 from cfnBtnActionPerformedin CandidateReg.java</u>

```
try {
            writer = new FileWriter("Candidate.txt", true);
            writer.append(nameText + "," + age + "," + occText + "," + exp
+ "," + bestText + "\n");
            writer.close();
            // create new candidate object
            Candidate candidate = new Candidate(nameText, age, occText,
exp, bestText);
            list.add(candidate);
            jDialog1.setVisible(false);
            new MainMenuGUI().setVisible(true);

        } catch (IOException ex) {

Logger.getLogger(CandidateReg.class.getName()).log(Level.SEVERE, null,
ex);
        }
```

File I/O

There are a few ways we have implemented the concept of File I/O in our GUI program. Mainly we have 2 ways of implementing said concept. Namely, the first method is reading information from a text file and inserting the read information to create the objects and adding them into the array of objects. Then, we apply the concept by writing the information obtained from the program into the text file.

Method 1 (Example is from MainMenuGUI.java)

```
private void loadVoterData() {
        try {
            voterlist.clear();
            File file = new File("Voter.txt");
            FileReader ff = new FileReader(file);
            BufferedReader bb = new BufferedReader(ff);
            String line;
            while ((line = bb.readLine()) != null) {
                String[] parts = line.split(",");
                String name = parts[0];
                String icNo = parts[1];
                String voted = parts[2];

                Voter voter = new Voter(name, icNo, voted);
                voterlist.add(voter);
            }
            bb.close();
            ff.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

Method 2 (Example is from CandidateReg.java cfnBtnActionPerformed)

```
FileWriter writer;
        try {
            writer = new FileWriter("Candidate.txt", true);
            writer.append(nameText + "," + age + "," + occText + "," + exp
+ "," + bestText + "\n");
            writer.close();
            // create new candidate object
            Candidate candidate = new Candidate(nameText, age, occText,
exp, bestText);
            list.add(candidate);
            jDialog1.setVisible(false);
            new MainMenuGUI().setVisible(true);

        } catch (IOException ex) {
```

```
Logger.getLogger(CandidateReg.class.getName()).log(Level.SEVERE, null, 
ex);
        }
```

<u>GUI containers and components</u>

Overall for this program, there were 5 main JFrame Forms used which are MainMenuGUI, CandidateReg, VoterRegister, VoteGUI and adminGUI. There are multiple types of containers and components used such as the Panel, labels, buttons, text areas, combo boxes, password fields, and dialogs. Some of them are within the JFrame Forms themselves while some are popup Dialog windows which will only show up if certain actions are performed.

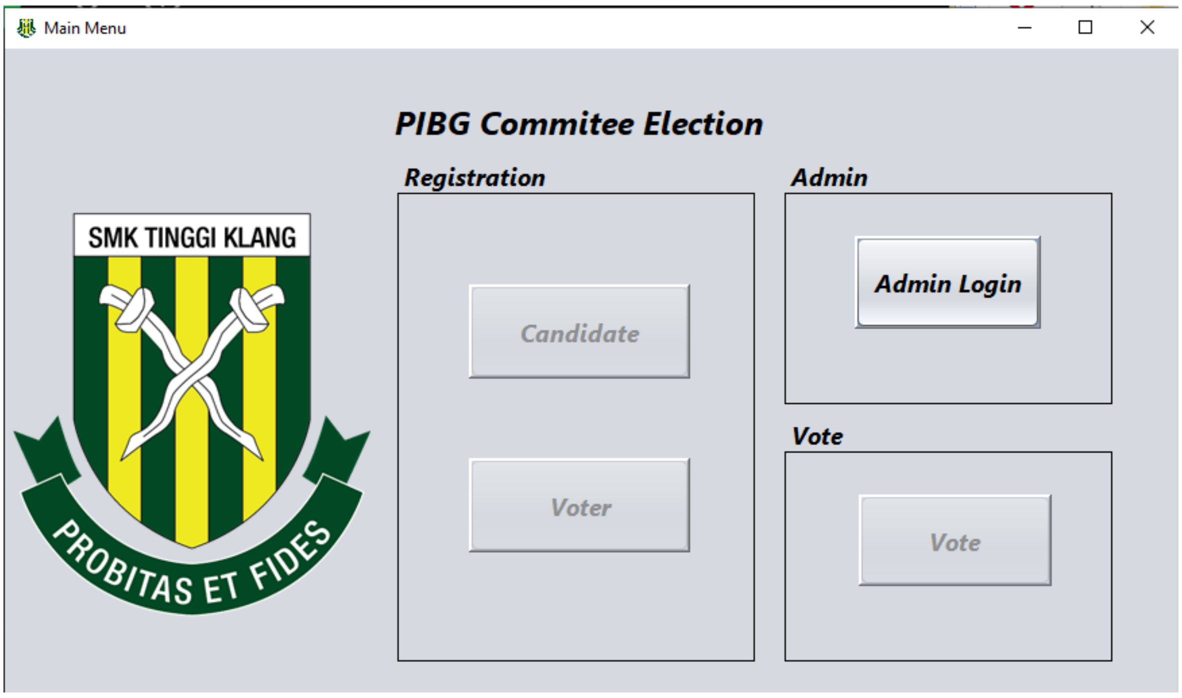


Figure 1: The MainMenuGUI showing labels, buttons and panels.

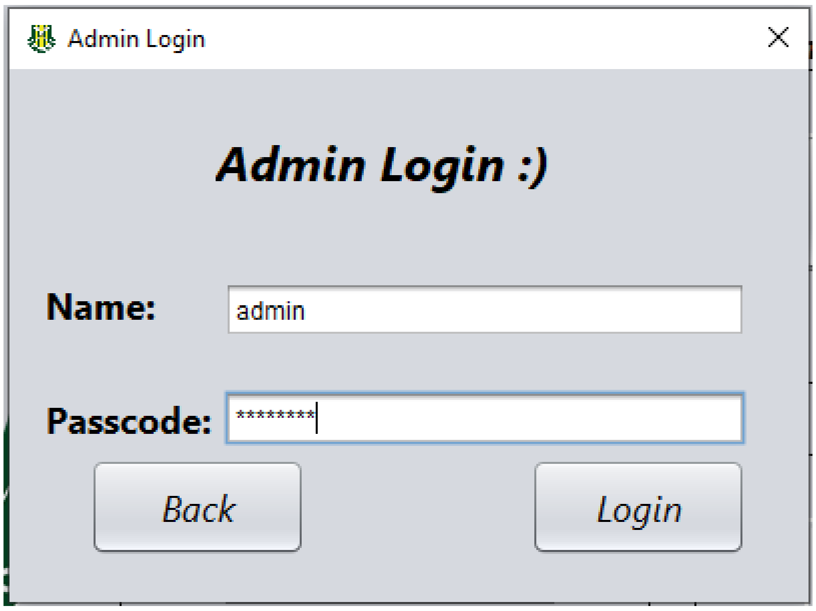


Figure 2: The adminLogin dialog showing labels, buttons, text and password fields.

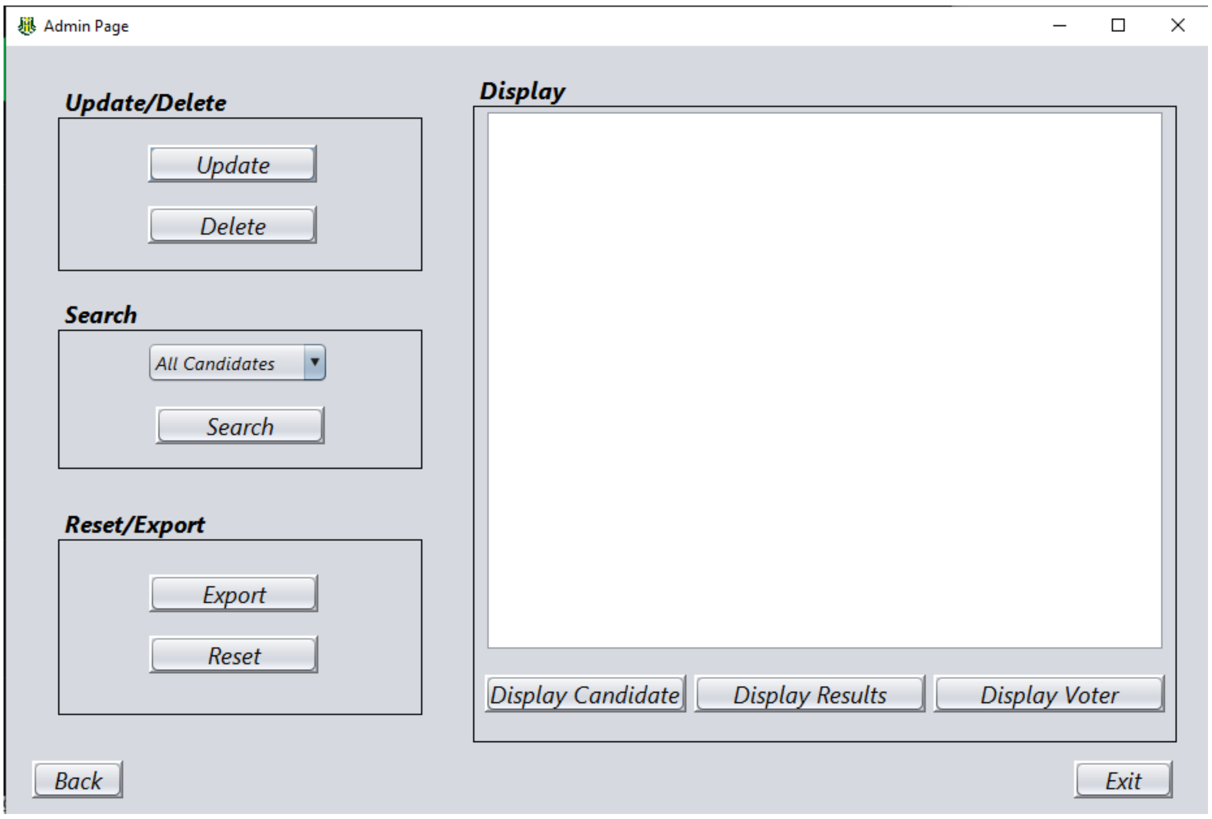


Figure 3: The adminGUI showing combo box and text area.

Code

Voter Class

```
public class Voter {

    private String name;
    private String icNo;
    private String voted;

    public Voter(String name, String icNo, String voted) {
        this.name = name;
        this.icNo = icNo;
        this.voted = voted;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getIcNo() {
        return icNo;
    }

    public void setIcNo(String icNo) {
        this.icNo = icNo;
    }

    public String getVoted() {
        return voted;
    }

    public void setVoted(String voted) {
        this.voted = voted;
    }

    @Override
    public String toString() {
        return "Parent{" + "name=" + name + ", icNo=" + icNo + ", voted="
+ voted + '}';
    }

}
```

Candidate Class

```
public class Candidate {

    private String name;
    private int Age;
    private String Job;
    private int Experience;
    private String Achievements;

    public Candidate(String name, int Age, String Job, int Experience, 
String Achievements) {
        this.name = name;
        this.Age = Age;
        this.Job = Job;
        this.Experience = Experience;
        this.Achievements = Achievements;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getAge() {
        return Age;
    }

    public void setAge(int Age) {
        this.Age = Age;
    }

    public String getJob() {
        return Job;
    }

    public void setJob(String Job) {
        this.Job = Job;
    }

    public int getExperience() {
        return Experience;
    }
    public void setExperience(int Experience) {
        this.Experience = Experience;
    }

    public String getAchievements() {
        return Achievements;
    }

    public void setAchievements(String Achievements) {
        this.Achievements = Achievements;
    }

    @Override
    public String toString() {
        return "Candidate{" + "name=" + name + ", Age=" + Age + ", Job=" +
Job + ", Experience=" + Experience + ", Achievements=" + Achievements +
'}';
    }

}
```

<u>Timeline Class</u>

```
import java.time.LocalDateTime;

class Timeline {

    public static LocalDateTime registrationStart = LocalDateTime.of(2023,
1, 16, 16, 0);
    public static LocalDateTime registrationEnd = LocalDateTime.of(2023,
1, 22, 14, 0);
    public static LocalDateTime votingStart = LocalDateTime.of(2023, 1,
16, 22, 43);
    public static LocalDateTime votingEnd = LocalDateTime.of(2023, 1, 16,
18, 59);

    public static boolean isRegistrationPeriodActive() {
        LocalDateTime now = LocalDateTime.now();
        return now.isAfter(registrationStart) &&
now.isBefore(registrationEnd);
    }

    public static boolean isVotingPeriodActive() {
        LocalDateTime now = LocalDateTime.now();
        return now.isAfter(votingStart) && now.isBefore(votingEnd);
    }
}
```

MainMenuGUI

```
import java.awt.Image;
import java.awt.Toolkit;
import java.awt.event.KeyEvent;
import java.awt.event.WindowEvent;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;
import javax.swing.JOptionPane;
import java.time.LocalDateTime;
import javax.swing.ImageIcon;
import static my.PibgVoteApp.MainMenuGUI.CandidateListHolder.list;
import static my.PibgVoteApp.MainMenuGUI.VoterListHolder.voterlist;

public class MainMenuGUI extends javax.swing.JFrame {

    private static MainMenuGUI mainMenuInstance;

    public class CandidateListHolder {

        public static ArrayList<Candidate> list = new ArrayList<>();
    }

    public class VoterListHolder {

        public static ArrayList<Voter> voterlist = new ArrayList<>();
    }


public MainMenuGUI() {
        initComponents();
        setImage();
        mainMenuInstance = this;
        loadCanData();
        loadVoterData();
        checkTime();
    }
```

```
@Override
    protected void processWindowEvent(WindowEvent e) {
        if (e.getID() == WindowEvent.WINDOW_CLOSING) {
            // prevent the window from being closed
            Object[] options = {"OK"};
            JOptionPane.showOptionDialog(this, "You are not allowed to
exit!", "NO EXIT",
                    JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE,
                    null, options, options[0]);
            return;
        }
        super.processWindowEvent(e);
    }


private void VoteBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        voteLogin.setVisible(true);
        voteLogin.setSize(415, 340);
        voteLogin.setLocationRelativeTo(null);
    }

    private void CanReBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        MainMenuGUI.closeMainMenu();
        new CandidateReg().setVisible(true);
    }

    private void VoterReBtnActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        MainMenuGUI.closeMainMenu();
        new VoterRegister().setVisible(true);
    }

private void adminBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        adminLogin.setVisible(true);
        adminLogin.setSize(415, 305);
        adminLogin.setLocationRelativeTo(null);
    }

    private void adminBackBtnActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
        adminLogin.setVisible(false);     }
```

```
private void adminLoginBtnActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        String name = txtAdmin.getText();
        String pass = txtAdminPass.getText();

        if (name.equals("admin") && pass.equals("stia1123")) {
            adminLogin.setVisible(false);
            MainMenuGUI.closeMainMenu();
            new adminGUI().setVisible(true);
        } else {
            JOptionPane.showMessageDialog(null, "Wrong Password", "Error",
JOptionPane.ERROR_MESSAGE);
            txtAdmin.setText("");
            txtAdminPass.setText("");
        }
    }

private void voterLoginBackBtnActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
        voteLogin.setVisible(false);
    }


private void voterLoginCfmBtnActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
        if (voteLoginName.getText().isEmpty() ||
voteLoginIcNo.getText().isEmpty()) {
            Object[] options = {"OK"};
            JOptionPane.showOptionDialog(this, "Please fill in all
fields", "Error",
                    JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE,
                    null, options, options[0]);

        } else {
            String nameText =
voteLoginName.getText().trim().toLowerCase();
            String icText = voteLoginIcNo.getText().trim().toLowerCase();

            String nameTextOrig = voteLoginName.getText();
            String icTextOrig = voteLoginIcNo.getText();

            boolean found = false;
            boolean registered = false;
            for (Voter v : voterlist) {
                if (v.getName().toLowerCase().trim().equals(nameText) &&
v.getIcNo().toLowerCase().trim().equals(icText)) {
                    registered = true;
                    if (v.getVoted().equals("-")) {
                        found = true;
                        break;
                    }
                }
            }

            if (found) {
                voteLogin.setVisible(false);
                MainMenuGUI.closeMainMenu();
                new VoteGUI(nameTextOrig, icTextOrig).setVisible(true);
            } else if (registered) {
                Object[] options = {"OK"};
                JOptionPane.showOptionDialog(this, "You have already voted
before!", "Error",
                        JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE,
                        null, options, options[0]);
            } else {
                Object[] options = {"OK"};
                JOptionPane.showOptionDialog(this, "You have not
registered as a voter!", "Error",
                        JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE,
                        null, options, options[0]);
            }
        }
    }
private void voteLoginIcNoKeyPressed(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        char ch = evt.getKeyChar();
        if (Character.isDigit(ch) || evt.getKeyCode() ==
KeyEvent.VK_BACK_SPACE || evt.getKeyCode() == KeyEvent.VK_DELETE) {
            voteLoginIcNo.setEditable(true);
            icNoLabel.setText("");
        } else {
            voteLoginIcNo.setEditable(false);
            icNoLabel.setText("Please Enter Numbers Only!");
        }}

public static void closeMainMenu() {
        mainMenuInstance.setVisible(false);
    }
```

```
private void loadCanData() {
        try {
            list.clear();
            File file = new File("Candidate.txt");
            FileReader fr = new FileReader(file);
            BufferedReader br = new BufferedReader(fr);
            String line;
            while ((line = br.readLine()) != null) {
                String[] parts = line.split(",");
                String name = parts[0];
                int age = Integer.parseInt(parts[1]);
                String job = parts[2];
                int experience = Integer.parseInt(parts[3]);
                String achievements = parts[4];
                Candidate candidate = new Candidate(name, age, job,
experience, achievements);
                list.add(candidate);
            }
            br.close();
            fr.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    private void loadVoterData() {
        try {
            voterlist.clear();
            File file = new File("Voter.txt");
            FileReader ff = new FileReader(file);
            BufferedReader bb = new BufferedReader(ff);
            String line;
            while ((line = bb.readLine()) != null) {
                String[] parts = line.split(",");
                String name = parts[0];
                String icNo = parts[1];
                String voted = parts[2];

                Voter voter = new Voter(name, icNo, voted);
                voterlist.add(voter);
            }
            bb.close();
            ff.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

```
public void checkTime() {
        LocalDateTime now = LocalDateTime.now();
        if (Timeline.isRegistrationPeriodActive()) {
            CanReBtn.setEnabled(true);
            VoterReBtn.setEnabled(true);
            VoteBtn.setEnabled(false);
        } else {
            CanReBtn.setEnabled(false);
            VoterReBtn.setEnabled(false);

            if (Timeline.isVotingPeriodActive()) {
                VoteBtn.setEnabled(true);
            } else {
                VoteBtn.setEnabled(false);
            }
        }
    }

    public void setImage() {
        Image image =
Toolkit.getDefaultToolkit().getImage(getClass().getResource("sekolah-tingg
i-klang-logo-DBDC58156C-seeklogo.com.png"));
        ImageIcon icon = new ImageIcon(image);
        setIconImage(icon.getImage());
        adminLogin.setIconImage(icon.getImage());
        voteLogin.setIconImage(icon.getImage());
    }
```

<u>CandidateReg</u>

```
import java.awt.Image;
import java.awt.Toolkit;
import java.awt.event.KeyEvent;
import java.awt.event.WindowEvent;
import java.io.FileWriter;
import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.ImageIcon;
import javax.swing.JOptionPane;
import static my.PibgVoteApp.MainMenuGUI.CandidateListHolder.list;

public class CandidateReg extends javax.swing.JFrame {

    private static CandidateReg candidateRegInstance;

public CandidateReg() {
        initComponents();
        setImage();
        candidateRegInstance = this;
    }

    @Override
    protected void processWindowEvent(WindowEvent e) {
        if (e.getID() == WindowEvent.WINDOW_CLOSING) {
            // prevent the window from being closed
            Object[] options = {"OK"};
            JOptionPane.showOptionDialog(this, "You are not allowed to 
exit!", "NO EXIT",
                    JOptionPane.DEFAULT_OPTION, 
JOptionPane.INFORMATION_MESSAGE,
                    null, options, options[0]);
            return;
        }
        super.processWindowEvent(e);
    }

private void CandidateRegClearActionPerformed(java.awt.event.ActionEvent 
evt) {
        // TODO add your handling code here:
        txtName.setText("");
        txtAge.setText("");
        txtJob.setText("");
        txtExp.setText("");
        txtBest.setText("");
    }
```

```
private void CandidateRegSaveActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
        if (txtName.getText().isEmpty() || txtAge.getText().isEmpty() ||
txtJob.getText().isEmpty() || txtExp.getText().isEmpty() ||
txtBest.getText().isEmpty()) {
            Object[] options = {"OK"};
            JOptionPane.showOptionDialog(this, "Please fill in all
fields", "Error",
                    JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE,
                    null, options, options[0]);
        } else {
            try {
                String nameText = txtName.getText();
                String ageText = txtAge.getText();
                int age = Integer.parseInt(ageText);
                String occText = txtJob.getText();
                String expText = txtExp.getText();
                int exp = Integer.parseInt(expText);
                String bestText = txtBest.getText();

                String nameTextModified = nameText.trim().toLowerCase();
                boolean duplicate = false;
                for (Candidate c : list) {
                    if
(c.getName().trim().toLowerCase().equals(nameTextModified) && c.getAge()
== age) {
                        duplicate = true;
                        break;
                    }
                }
                if (duplicate) {
                // display option pane saying there is already a record of
candidate registered
                    Object[] options = {"OK"};
                    JOptionPane.showOptionDialog(this, "Candidate already
registered", "Duplicate", JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE, null, options, options[0]);
                    txtName.setText("");
                    txtAge.setText("");
                    txtJob.setText("");
                    txtExp.setText("");
                    txtBest.setText("");
                } else {
                    nameLabel.setText(nameText);
                    ageLabel.setText(ageText);
                    occLabel.setText(occText);
                    expLabel.setText(expText);
                    bestLabel.setText(bestText);
                    CandidateReg.closeCandidateReg();
                    jDialog1.setVisible(true);
                    jDialog1.setSize(740, 380);
                    jDialog1.setLocationRelativeTo(null);
                }
            } catch (NumberFormatException ex) {
                Object[] options = {"OK"};
                JOptionPane.showOptionDialog(this, "Please enter a valid
number for Age and Experience", "Error", JOptionPane.DEFAULT_OPTION,
JOptionPane.ERROR_MESSAGE, null, options, options[0]);
                txtAge.setText("");;
                txtExp.setText("");
            }
        }
    }

private void CandidateRegBackActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
        //back
        CandidateReg.closeCandidateReg();
        new MainMenuGUI().setVisible(true);
    }

private void canBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        jDialog1.setVisible(false);
        new CandidateReg().setVisible(true);
    }

private void cfmBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String nameText = txtName.getText();
        String ageText = txtAge.getText();
        int age = Integer.parseInt(ageText);
        String occText = txtJob.getText();
        String expText = txtExp.getText();
        int exp = Integer.parseInt(expText);
        String bestText = txtBest.getText();

        FileWriter writer;
        try {
            writer = new FileWriter("Candidate.txt", true);
            writer.append(nameText + "," + age + "," + occText + "," + exp
+ "," + bestText + "\n");
            writer.close();
            // create new candidate object
            Candidate candidate = new Candidate(nameText, age, occText,
exp, bestText);
            list.add(candidate);
            jDialog1.setVisible(false);
            new MainMenuGUI().setVisible(true);

        } catch (IOException ex) {

Logger.getLogger(CandidateReg.class.getName()).log(Level.SEVERE, null,
ex);
        }
    }

private void txtAgeKeyPressed(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        char c = evt.getKeyChar();
        if (Character.isDigit(c) || evt.getKeyCode() ==
KeyEvent.VK_BACK_SPACE || evt.getKeyCode() == KeyEvent.VK_DELETE) {
            txtAge.setEditable(true);
            numLabel.setText("");
        } else {
            numLabel.setText("Please Enter Numbers Only!");
        }
    }

    private void txtExpKeyPressed(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        char ch = evt.getKeyChar();
        if (Character.isDigit(ch) || evt.getKeyCode() ==
KeyEvent.VK_BACK_SPACE || evt.getKeyCode() == KeyEvent.VK_DELETE) {
            txtExp.setEditable(true);
            ageLbl.setText("");
        } else {
            ageLbl.setText("Please Enter Numbers Only!");
        }
    }
```

```
public void setImage() {
        Image image =
Toolkit.getDefaultToolkit().getImage(getClass().getResource("sekolah-tingg
i-klang-logo-DBDC58156C-seeklogo.com.png"));
        ImageIcon icon = new ImageIcon(image);
        setIconImage(icon.getImage());
        jDialog1.setIconImage(icon.getImage());
    }


public static void closeCandidateReg() {
        candidateRegInstance.setVisible(false);
    }
```

VoterRegister

```
import java.awt.Image;
import java.awt.Toolkit;
import java.awt.event.KeyEvent;
import java.awt.event.WindowEvent;
import java.io.FileWriter;
import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.ImageIcon;
import javax.swing.JOptionPane;
import static my.PibgVoteApp.MainMenuGUI.VoterListHolder.voterlist;

public class VoterRegister extends javax.swing.JFrame {

    private static VoterRegister voterRegisterInstance;

public VoterRegister() {
        initComponents();
        setImage();
        voterRegisterInstance = this;
    }

@Override
    protected void processWindowEvent(WindowEvent e) {
        if (e.getID() == WindowEvent.WINDOW_CLOSING) {
            // prevent the window from being closed
            Object[] options = {"OK"};
            JOptionPane.showOptionDialog(this, "You are not allowed to
exit!", "NO EXIT",
                    JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE,
                    null, options, options[0]);
            return;
        }
        super.processWindowEvent(e);
    }

private void voteRBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        voteRCfmSaved.setVisible(false);
        new MainMenuGUI().setVisible(true);
    }
```

```
private void voteRBackBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        VoterRegister.closeVoterRegister();
        new MainMenuGUI().setVisible(true);
    }

private void voteCfmBackBtnActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        voteRCfm.setVisible(false);
        new VoterRegister().setVisible(true);
    }

private void voteRSaveBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        if (txtNameRVoter.getText().isEmpty() ||
txtICRvoter.getText().isEmpty()) {
            Object[] options = {"OK"};
            JOptionPane.showOptionDialog(this, "Please fill in all
fields", "Error",
                    JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE,
                    null, options, options[0]);

        } else {
            String nameText = txtNameRVoter.getText();
            String icText = txtICRvoter.getText();
            String votedText = "-";
            String nameTextModified = nameText.trim().toLowerCase();
            String icTextModified = icText.trim().toLowerCase();

            // check if a voter with the same name and IC already exists
in the list
            boolean duplicate = false;
            for (Voter v : voterlist) {
                if
(v.getName().toLowerCase().trim().equals(nameTextModified) &&
v.getIcNo().toLowerCase().trim().equals(icTextModified)) {
                    duplicate = true;
                    break;
                }
            }

            if (duplicate) {
                // display option pane saying there is already a record of
voter registered
                Object[] options = {"OK"};
                JOptionPane.showOptionDialog(this, "Voter already
registered", "Duplicate", JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE,
                        null, options, options[0]);
                txtNameRVoter.setText("");
                txtICRvoter.setText("");
            } else {
                txtNameVoter.setText(nameText);
                textICvoter.setText(icText);
                VoterRegister.closeVoterRegister();
                voteRCfm.setVisible(true);
                voteRCfm.setSize(415, 300);
                voteRCfm.setLocationRelativeTo(null);
            }
        }
    }

private void voteCfmSaveBtnActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        String nameText = txtNameRVoter.getText();
        String icText = txtICRvoter.getText();
        String votedText = "-";

        FileWriter writer;
        try {
            writer = new FileWriter("Voter.txt", true);
            writer.append(nameText + "," + icText + "," + votedText +
"\n");
            writer.close();
            // create new candidate object
            Voter voter = new Voter(nameText, icText, votedText);
            voterlist.add(voter);
            VoterRegister.closeVoterRegister();

        } catch (IOException ex) {

Logger.getLogger(CandidateReg.class.getName()).log(Level.SEVERE, null,
ex);
        }
        voteRCfm.setVisible(false);
        voteRCfmSaved.setVisible(true);
        voteRCfmSaved.setSize(341, 163);
        voteRCfmSaved.setLocationRelativeTo(null);
    }
```

```
private void txtICRvoterKeyPressed(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        char ch = evt.getKeyChar();
        if (Character.isDigit(ch) || evt.getKeyCode() ==
KeyEvent.VK_BACK_SPACE || evt.getKeyCode() == KeyEvent.VK_DELETE) {
            txtICRvoter.setEditable(true);
            icLbl.setText("");
        } else {
            txtICRvoter.setEditable(false);
            icLbl.setText("Please Enter Numbers Only!");
        }
    }

    public static void closeVoterRegister() {
        voterRegisterInstance.setVisible(false);
    }

    public void setImage() {
        Image image =
Toolkit.getDefaultToolkit().getImage(getClass().getResource("sekolah-tingg
i-klang-logo-DBDC58156C-seeklogo.com.png"));
        ImageIcon icon = new ImageIcon(image);
        setIconImage(icon.getImage());
        voteRCfmSaved.setIconImage(icon.getImage());
        voteRCfm.setIconImage(icon.getImage());
    }
```

adminGUI

```
import java.awt.Image;
import java.awt.Toolkit;
import java.awt.event.KeyEvent;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.time.LocalDateTime;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.ImageIcon;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import static my.PibgVoteApp.MainMenuGUI.CandidateListHolder.list;
import static my.PibgVoteApp.MainMenuGUI.VoterListHolder.voterlist;

public class adminGUI extends javax.swing.JFrame {

    private static adminGUI adminInstance;
    int candidateIndex;

public adminGUI() {
        initComponents();
        setImage();
        adminInstance = this;
        insertCandidate();
        checkUpDelTime();
    }
private void deleteBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        DelIndex.setVisible(true);
        DelIndex.setSize(400, 300);
        DelIndex.setLocationRelativeTo(null);
    }

    private void adminBackBtnActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
        adminGUI.closeAdmin();
        new MainMenuGUI().setVisible(true);
    }

    private void adminExitBtnActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
        System.exit(0);       }
```

```
private void searchBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String selectedName = (String) jComboBox1.getSelectedItem();
        String text = "Voters:\n======================\n";
        if (selectedName.equals("All Candidates")) {
            for (Voter v : voterlist) {
                if (!v.getVoted().equals("-")) {
                    text += "Name: " + v.getName() + "\nIcNo: " +
v.getIcNo() + "\nVoted for: " + v.getVoted() +
"\n======================\n";
                }
            }
        } else {
            for (Voter v : voterlist) {
                if (v.getVoted().equals(selectedName)) {
                    text += "Name: " + v.getName() + "\nIcNo: " +
v.getIcNo() + "\nVoted for: " + v.getVoted() +
"\n======================\n";
                }
            }
        }
        txtOutput.setText(text);
        txtOutput.setCaretPosition(0);
    }

private void displayVoteBtnActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        String text = "Voters:\n======================\n";
        for (Voter v : voterlist) {
            text += "Name: " + v.getName() + "\nIcNo: " + v.getIcNo() +
"\nVoted for: " + v.getVoted() + "\n======================\n";
        }
        txtOutput.setText(text);
        txtOutput.setCaretPosition(0);
    }

 private void CancelBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        CandUpdate.setVisible(false);
        new adminGUI().setVisible(true);
    }
```

```
private void displayCanBtnActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        txtOutput.setText("Candidates:\n========================\n");
        int candidateNum = 1;
        for (Candidate c : list) {
            String row = String.format("Candidate No: %-5d\nName: 
%-15s\nAge: %-5d\nJob: %-25s\nExperience: %-5d\nAchievements: %s\n", 
candidateNum, c.getName(), c.getAge(), c.getJob(), c.getExperience(), 
c.getAchievements());
            txtOutput.append(row);
            txtOutput.append("========================\n");
            candidateNum++;
            txtOutput.setCaretPosition(0);
        }}

private void UpdateConBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        list.set(candidateIndex, new Candidate(nameUpdate.getText(), 
Integer.parseInt(ageUpdate.getText()), jobUpdate.getText(), 
Integer.parseInt(expUpdate.getText()), bestUpdate.getText()));

        FileWriter writer;
        try {
            writer = new FileWriter("Candidate.txt");
            for (Candidate c : list) {
                String candidateInfo = c.getName() + "," + c.getAge() + 
"," + c.getJob() + "," + c.getExperience() + "," + c.getAchievements() + 
"\n";
                writer.write(candidateInfo);
            }
            writer.close();
            Object[] options = {"OK"};
            JOptionPane.showOptionDialog(this, "The Candidate Information 
has been updated!", "Information Update",
                    JOptionPane.DEFAULT_OPTION, 
JOptionPane.INFORMATION_MESSAGE,
                    null, options, options[0]);
            CandUpdate.setVisible(false);
            new adminGUI().setVisible(true);

        } catch (IOException ex) {
            
Logger.getLogger(CandidateReg.class.getName()).log(Level.SEVERE, null, 
ex);
        }
    }
```

```
private void CancelBtn1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        UpdateIndex.setVisible(false);
    }

    private void ConBtn1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        try {
            int index = Integer.parseInt(txtUpdate.getText()) - 1;
            if (index >= 0 && index < list.size()) {
                Candidate c = list.get(index);
                int candidateIndex = list.indexOf(c);
                setCandidateDetails(c, candidateIndex);
                UpdateIndex.setVisible(false);
                adminGUI.closeAdmin();
                CandUpdate.setVisible(true);
                CandUpdate.setSize(740, 430);
                CandUpdate.setLocationRelativeTo(null);
            } else {
                Object[] options = {"OK"};
                JOptionPane.showOptionDialog(this, "Candidate not found,
Please enter again!", "Error",
                        JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE,
                        null, options, options[0]);

                txtUpdate.setText("");
            }
        } catch (NumberFormatException e) {

            Object[] options = {"OK"};
            JOptionPane.showOptionDialog(this, "Please enter a valid
candidate number!", "Error",
                    JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE,
                    null, options, options[0]);

            txtUpdate.setText("");
        }
    }
private void updateBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:

        UpdateIndex.setVisible(true);
        UpdateIndex.setSize(400, 300);
        UpdateIndex.setLocationRelativeTo(null);
    }
```

```
  private void CancelBtn2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        DelIndex.setVisible(false);
    }

  private void DelConBtn1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        try {
            int index = Integer.parseInt(txtCandelete.getText()) - 1;
            if (index >= 0 && index < list.size()) {
                Candidate c = list.get(index);
                int candidateIndex = list.indexOf(c);
                setDelCandidateDetails(c, candidateIndex);
                adminGUI.closeAdmin();
                DelIndex.setVisible(false);
                CandDel.setVisible(true);
                CandDel.setSize(740, 430);
                CandDel.setLocationRelativeTo(null);

            } else {
                Object[] options = {"OK"};
                JOptionPane.showOptionDialog(this, "Candidate not found,
Please enter again!", "Error",
                        JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE,
                        null, options, options[0]);

                txtCandelete.setText("");
            }
        } catch (NumberFormatException e) {

            Object[] options = {"OK"};
            JOptionPane.showOptionDialog(this, "Please enter a valid
candidate number!", "Error",
                    JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE,
                    null, options, options[0]);

            txtCandelete.setText("");
        }
    }

private void canBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        CandDel.setVisible(false);
        new adminGUI().setVisible(true);
    }
```

```
private void DelCfmBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        list.remove(candidateIndex);

        FileWriter writer;
        try {
            writer = new FileWriter("Candidate.txt");
            for (Candidate c : list) {
                String candidateInfo = c.getName() + "," + c.getAge() +
"," + c.getJob() + "," + c.getExperience() + "," + c.getAchievements() +
"\n";
                writer.write(candidateInfo);
            }
            writer.close();
            Object[] options = {"OK"};
            JOptionPane.showOptionDialog(this, "The Candidate Information
has been deleted!", "Information Delete",
                    JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE,
                    null, options, options[0]);
            CandDel.setVisible(false);
            new adminGUI().setVisible(true);

        } catch (IOException ex) {
Logger.getLogger(CandidateReg.class.getName()).log(Level.SEVERE, null,
ex);
        }
    }

private void exportBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        JFileChooser chooser = new
JFileChooser("C:/Users/vince/OneDrive/Desktop");
        chooser.setDialogType(JFileChooser.SAVE_DIALOG);
        chooser.setSelectedFile(new File("MyFile.txt"));
        int result = chooser.showSaveDialog(null);
        if (result == JFileChooser.APPROVE_OPTION) {
            File selectedFile = chooser.getSelectedFile();
            // Save the file at the selected location
            try {
                FileWriter writer = new FileWriter(selectedFile);
                writer.write(txtOutput.getText());
                writer.close();
                Object[] options = {"OK"};
                JOptionPane.showOptionDialog(null, "File saved
successfully!", "Success", JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE, null, options, options[0]);
```

```
                txtOutput.setText("");
            } catch (IOException ex) {
                JOptionPane.showOptionDialog(null, "Error saving file!",
"Error", JOptionPane.DEFAULT_OPTION, JOptionPane.ERROR_MESSAGE, null, new
Object[]{}, null);
            }
        }
    }

private void displayResultsBtnActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
        // Create an array to store the vote counts for each candidate
        int[] voteCounts = new int[list.size()];

        // Iterate through the voter list and count the votes for each
candidate
        for (Voter v : voterlist) {
            for (int i = 0; i < list.size(); i++) {
                if (v.getVoted().equals(list.get(i).getName())) {
                    voteCounts[i]++;
                }
            }
        }
        // Display the results in the txtOutput text area
        txtOutput.setText("Results:\n=======================\n");
        for (int i = 0; i < list.size(); i++) {
            txtOutput.append(list.get(i).getName() + ": " + voteCounts[i]
+ " votes\n");
        }
        txtOutput.setCaretPosition(0);
    }

private void resetBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        int choice = JOptionPane.showConfirmDialog(null, "Do you want to
reset all the information?", "Reset", JOptionPane.YES_NO_OPTION);
        if (choice == JOptionPane.YES_OPTION) {
            try {
                list.clear();
                FileWriter fw = new FileWriter("Candidate.txt", false);
                fw.write("");
                fw.close();
                FileWriter vw = new FileWriter("voter.txt", false);
                vw.write("");
                vw.close();
                voterlist.clear();
                txtOutput.setText("");

                JOptionPane.showMessageDialog(null, "Information reset
successfully");
            } catch (IOException e) {
                e.printStackTrace();
            }
        } else if (choice == JOptionPane.NO_OPTION) {
            JOptionPane.showMessageDialog(null, "Reset canceled");
        }
    }

private void ageUpdateKeyPressed(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        char c = evt.getKeyChar();
        if (Character.isDigit(c) || evt.getKeyCode() ==
KeyEvent.VK_BACK_SPACE || evt.getKeyCode() == KeyEvent.VK_DELETE) {
            ageUpdate.setEditable(true);
            ageLabel1.setText("");
        } else {
            ageUpdate.setEditable(false);
            ageLabel1.setText("Please Enter Numbers Only!");
        }
    }

private void expUpdateKeyPressed(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        char ch = evt.getKeyChar();
        if (Character.isDigit(ch) || evt.getKeyCode() ==
KeyEvent.VK_BACK_SPACE || evt.getKeyCode() == KeyEvent.VK_DELETE) {
            expUpdate.setEditable(true);
            expLbl1.setText("");
        } else {
            expUpdate.setEditable(false);
            expLbl1.setText("Please Enter Numbers Only!");
        }
    }

public static void closeAdmin() {
        adminInstance.setVisible(false);
    }

    private void insertCandidate() {
        jComboBox1.addItem("All Candidates");
        for (Candidate c : list) {
            jComboBox1.addItem(c.getName());
        }    }
```

```
private void txtCandeleteKeyPressed(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        char ch = evt.getKeyChar();
        if (Character.isDigit(ch) || evt.getKeyCode() ==
KeyEvent.VK_BACK_SPACE || evt.getKeyCode() == KeyEvent.VK_DELETE) {
            txtCandelete.setEditable(true);
            dltLbl.setText("");
        } else {
            txtCandelete.setEditable(false);
            dltLbl.setText("Please Enter Numbers Only!");
        }
    }

public void setCandidateDetails(Candidate c, int candidateIndex) {
        // set the details of the candidate object to the textfields
        nameUpdate.setText(c.getName());
        ageUpdate.setText(Integer.toString(c.getAge()));
        jobUpdate.setText(c.getJob());
        expUpdate.setText(Integer.toString(c.getExperience()));
        bestUpdate.setText(c.getAchievements());
        this.candidateIndex = candidateIndex;
    }

    public void setDelCandidateDetails(Candidate c, int candidateIndex) {
        // set the details of the candidate object to the textfields
        nameLabel.setText(c.getName());
        ageLabel.setText(Integer.toString(c.getAge()));
        occLabel.setText(c.getJob());
        expLabel.setText(Integer.toString(c.getExperience()));
        bestLabel.setText(c.getAchievements());
        this.candidateIndex = candidateIndex;
    }

    public void checkUpDelTime() {
        LocalDateTime now = LocalDateTime.now();
        if (Timeline.isRegistrationPeriodActive()) {
            updateBtn.setEnabled(true);
            deleteBtn.setEnabled(true);

        } else if (Timeline.isVotingPeriodActive()) {
            updateBtn.setEnabled(false);
            deleteBtn.setEnabled(false);
        }

    }
```

```
public void setImage() {
        Image image =
Toolkit.getDefaultToolkit().getImage(getClass().getResource("sekolah-tingg
i-klang-logo-DBDC58156C-seeklogo.com.png"));
        ImageIcon icon = new ImageIcon(image);
        setIconImage(icon.getImage());
        CandUpdate.setIconImage(icon.getImage());
        UpdateIndex.setIconImage(icon.getImage());
        CandDel.setIconImage(icon.getImage());
        DelIndex.setIconImage(icon.getImage());
    }
```

VoteGUI

```
import java.awt.Image;
import java.awt.Toolkit;
import java.awt.event.WindowEvent;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import javax.swing.ImageIcon;
import javax.swing.JOptionPane;
import my.PibgVoteApp.MainMenuGUI.CandidateListHolder;
import static my.PibgVoteApp.MainMenuGUI.VoterListHolder.voterlist;

public class VoteGUI extends javax.swing.JFrame {

    private String name;
    private String icNo;
    private static VoteGUI voteInstance;
    ArrayList<Candidate> list = CandidateListHolder.list;

public VoteGUI(String name, String icNo) {
        this.name = name;
        this.icNo = icNo;
        initComponents();
        setImage();
        voteInstance = this;
        display();
        insertCandidate();
    }

    @Override
    protected void processWindowEvent(WindowEvent e) {
        if (e.getID() == WindowEvent.WINDOW_CLOSING) {
            // prevent the window from being closed
            Object[] options = {"OK"};
            JOptionPane.showOptionDialog(this, "You are not allowed to
exit!", "NO EXIT",
                    JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE,
                    null, options, options[0]);
            return;
        }
        super.processWindowEvent(e);
    }

private void voteBackActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
```

```
        //back
        VoteGUI.closeVoteGUI();
        new MainMenuGUI().setVisible(true);
    }

public static void closeVoteGUI() {
        voteInstance.setVisible(false);
    }



private void voteSaveActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        int response = JOptionPane.showConfirmDialog(this, "Do you want to
proceed with your vote?", "Confirm Vote", JOptionPane.YES_NO_OPTION,
JOptionPane.INFORMATION_MESSAGE);

        if (response == JOptionPane.YES_OPTION) {

            String selectedName = (String) jComboBox1.getSelectedItem();
            for (Voter v : voterlist) {
                if
(v.getName().toLowerCase().trim().equals(name.toLowerCase().trim()) &&
v.getIcNo().toLowerCase().trim().equals(icNo.toLowerCase().trim())) {
                    // Update the "voted" attribute of the voter object
                    v.setVoted(selectedName);

                    try ( BufferedWriter bw = new BufferedWriter(new
FileWriter("voter.txt"))) {
                        for (Voter voter : voterlist) {
                            bw.write(voter.getName() + "," +
voter.getIcNo() + "," + voter.getVoted());
                            bw.newLine();
                        }
                    } catch (IOException e) {
                        e.printStackTrace();
                    }
                }
            }
            Object[] options = {"OK"};
            JOptionPane.showOptionDialog(this, "You vote has been
recorded!", "Vote Successful",
                    JOptionPane.DEFAULT_OPTION,
JOptionPane.INFORMATION_MESSAGE,
                    null, options, options[0]);

            VoteGUI.closeVoteGUI();
            new MainMenuGUI().setVisible(true);

        } else if (response == JOptionPane.NO_OPTION) {
            JOptionPane.getRootFrame().dispose();
        }
    }

private void insertCandidate() {
        for (Candidate c : list) {
            jComboBox1.addItem(c.getName());
        }}
private void display() {
        // Clear the text area
        jTextArea1.setText("");

        // Print the table header
        String header = String.format("%-5s %-15s %-5s %-25s %-5s %s\n",
"No.", "Name", "Age", "Job", "Exp.", "Achievements");
        jTextArea1.append(header);

jTextArea1.append("-----------------------------------------------------------
-------------------------------------\n");

        // Print each object in the list, including the candidate number
        int candidateNum = 1;
        for (Candidate c : list) {
            String row = String.format("%-5d %-15s %-5d %-25s %-5d %s\n",
candidateNum, c.getName(), c.getAge(), c.getJob(), c.getExperience(),
c.getAchievements());
            jTextArea1.append(row);
            candidateNum++;
        }
    }

public void setImage() {
        Image image =
Toolkit.getDefaultToolkit().getImage(getClass().getResource("sekolah-tingg
i-klang-logo-DBDC58156C-seeklogo.com.png"));
        ImageIcon icon = new ImageIcon(image);
        setIconImage(icon.getImage());

    }
```

Sample run

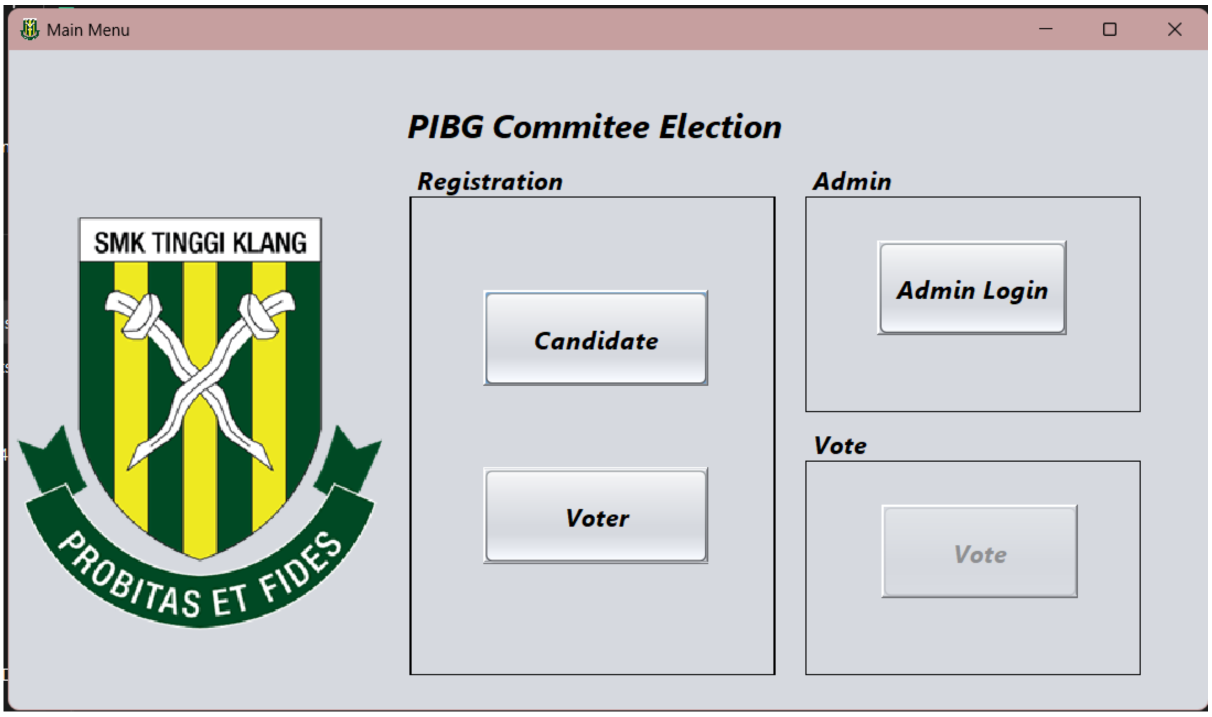
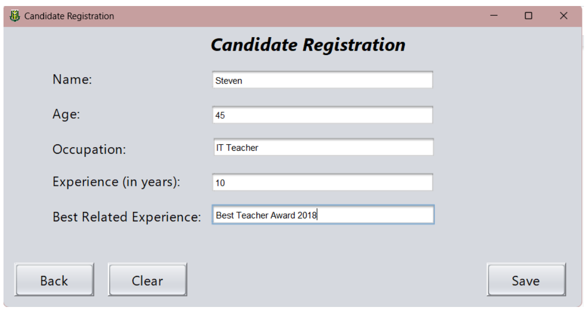


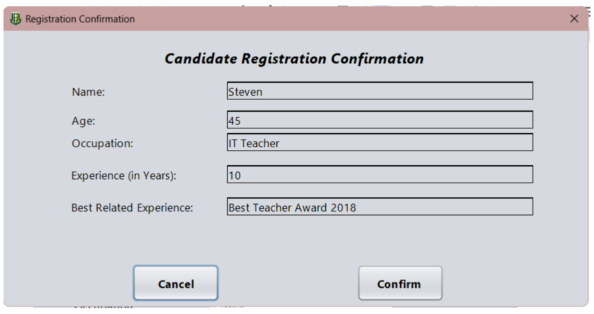
Image 1: Main Menu in Register Day

This image 1 shows the main menu for the PIBG SMK Tinggi Klang for election. .First of all, there are 4 buttons in the image which are Candidate, Voter, Admin Login, and Vote button. So , assume that the day is the registration day; the candidate can register themselves by clicking the Candidate button to register themselves as one of the candidates for the election. The next button is the voter, the voter can click the Voter button to register themselves as a voter for them to vote. Moreover, the Admin Login button is only for the admin to login only which requires a password. Last but not least, which is the Vote button so when it is register day the voter can vote for them as they need to wait until the vote day has come.



Candidate Registration

Name:	Steven
Age:	45
Occupation:	IT Teacher
Experience (in years):	10
Best Related Experience:	Best Teacher Award 2018



Candidate Registration Confirmation

Name:	Steven
Age:	45
Occupation:	IT Teacher
Experience (in Years):	10
Best Related Experience:	Best Teacher Award 2018

Image 2: Candidate Registration form Image 3: Candidate Confirmation dialog

From above these 2 images, when the candidate wants to register themselves they will need to click the Candidate button and it will bring them to this page which is the picture on the left. After that, the candidate can register by put their information in, and after they put all the information they can press the Save button and it will pop out a dialog which is the confirmation from image 3 once again for the candidate to see their information again and if they confirm then can press the confirm button; and will bring the user to comeback to Main Menu.

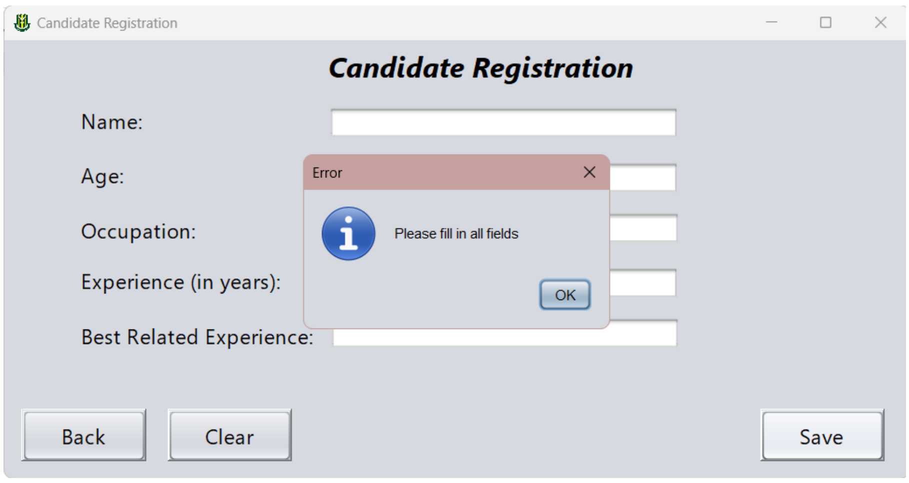
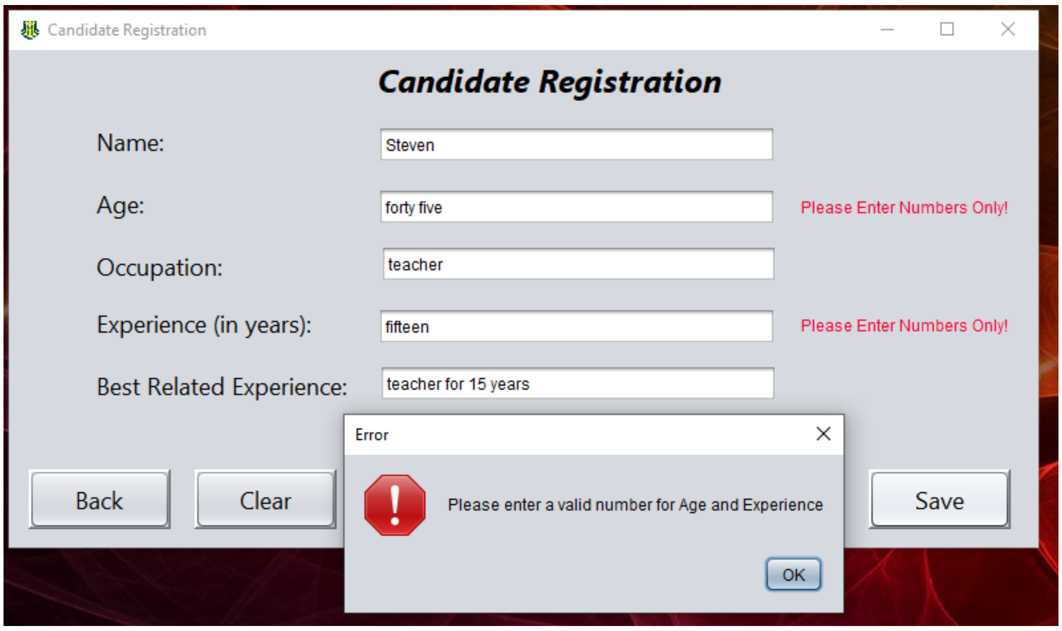


Image 4: Error (Fill in all the fields)



Name: Steven

Age: forty five — Please Enter Numbers Only!

Occupation: teacher

Experience (in years): fifteen — Please Enter Numbers Only!

Best Related Experience: teacher for 15 years

Please enter a valid number for Age and Experience

Image 5: Showing the label and displaying error if invalid entry.

Above the two images showing when the candidate wants to proceed when their information is not fully input will pop out the message showing that the candidate needs to fill in all the fields which can be seen in Image 4. Moreover, image 5 showing that if the candidate uses a character or symbol in the age and experience(in years) text field will show the label which is "Please Enter Numbers Only!" to tell the candidate to key in only the numbers. If the user proceeds without heeding to the warning, the error pop-up will show, the age and experience textfields will be cleared after the ok button is clicked.

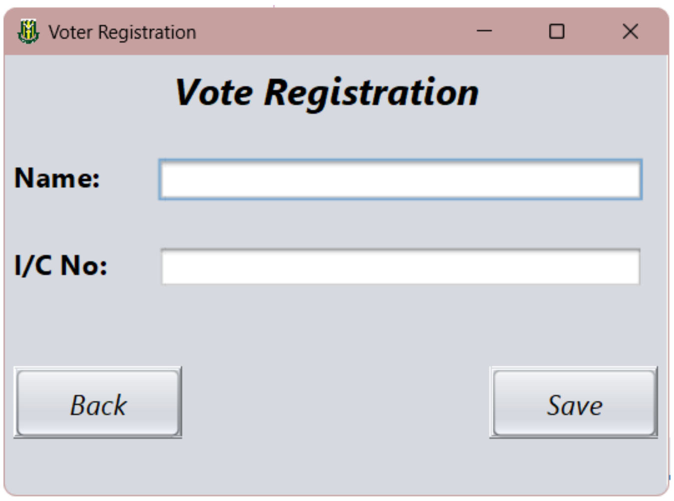
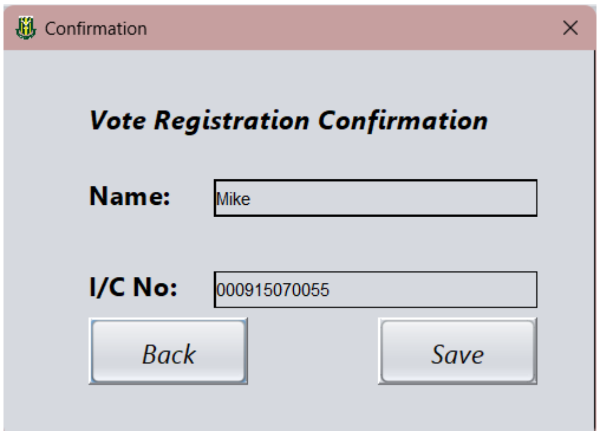
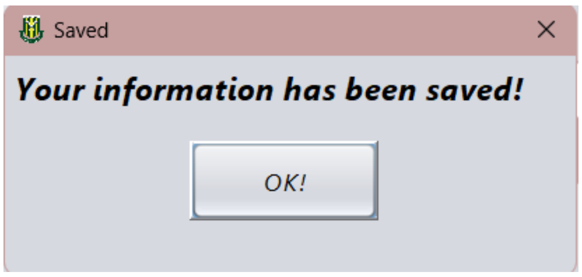


Image 6: Voter register dialog



Name: Mike

I/C No: 000915070055

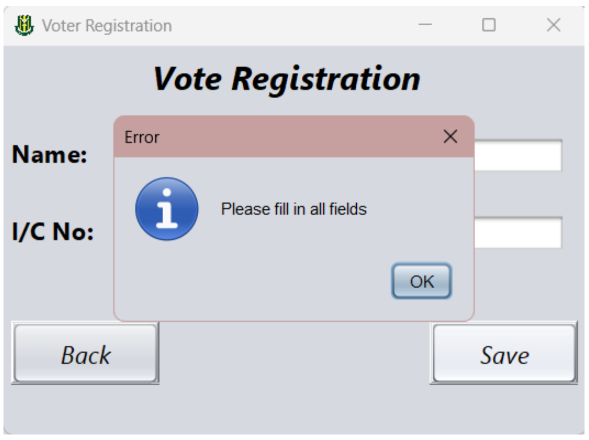


Your information has been saved!

Image 7: Voter Confirmation Dialog

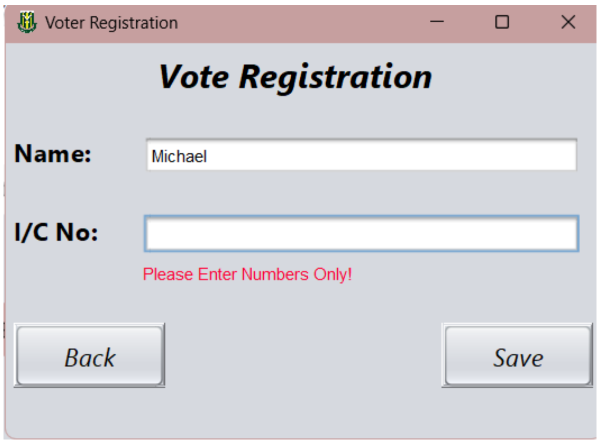
Image 8: Jdialog"Information saved"

Above these all are the flow of the voter registration, when voter want to register their name to take part in election they will need to press the Voter button and will pop out a jdialog which is the Voter register dialog to the voter to key in their name and Ic number to register. After they key in all the information, they need to press the Save button which will pop out the dialog which is looks like the image 7 which showing the confirmation to the voter to check their information and click the save button once they are confirm with their information and will pop out the dialog which is the message showing that the voter information has been saved and when click the "OK!" button will bring them to the Main Menu.



Please fill in all fields

Image 9: Error Message



Michael

Please Enter Numbers Only!

Image10: Showing Label

When the voter didn't pull all the information in the text fields and press the save button, will pop out an option pane which showing the message "Please fill in all the fields" to ensure the voter to key in all the information. The image 10 showing the label which will need the voter to make sure they only can key in numbers even if they want to key in character or symbol will pop out the label which tells the voter to enter numbers only.

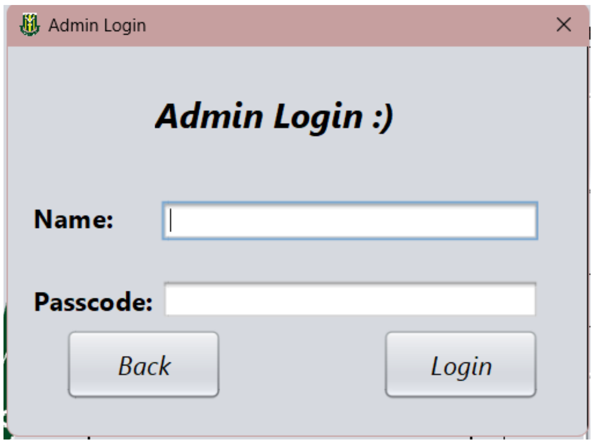
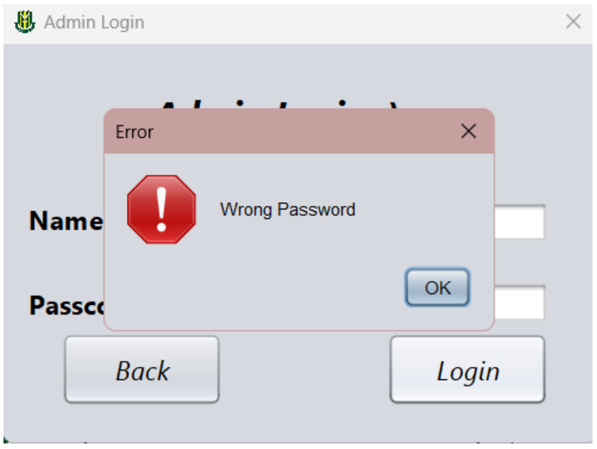


Image 11: Admin Login Dialog



Wrong Password

Image 12: Option Pane(Wrong Password)

When an admin wants to login, first they need to press the admin button which can be seen in Image 1 and if they click the button will pop out a dialog that looks alike in image 11, which requires the admin to key in the name and the password. If the admin key in the wrong password will show an option pane which is showing the message "Wrong Password" and require to key in again.

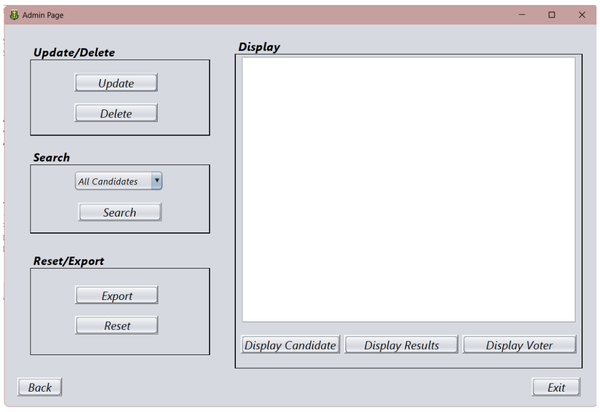
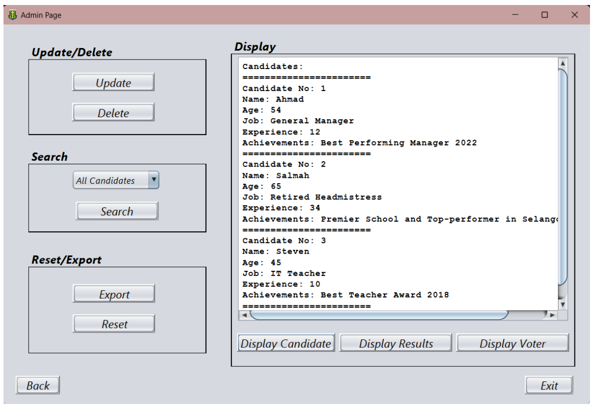


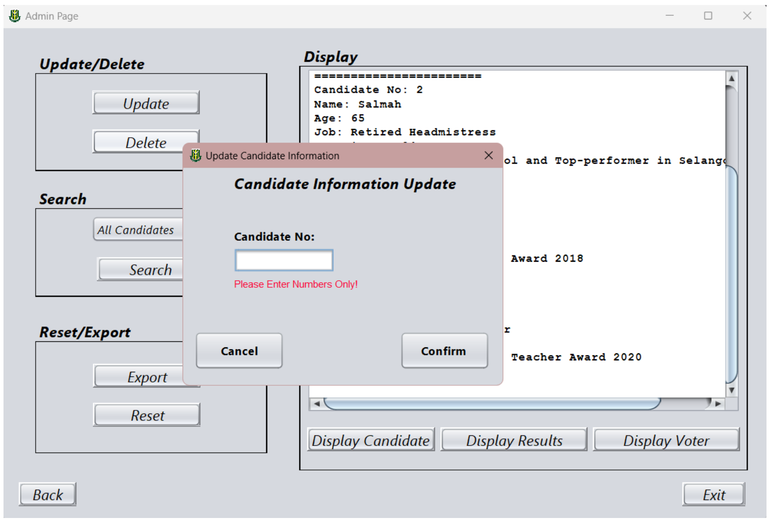
Image 13: Admin Page



```
Candidates:
========================
Candidate No: 1
Name: Ahmad
Age: 54
Job: General Manager
Experience: 12
Achievements: Best Performing Manager 2022
========================
Candidate No: 2
Name: Salmah
Age: 65
Job: Retired Headmistress
Experience: 34
Achievements: Premier School and Top-performer in Selango
========================
Candidate No: 3
Name: Steven
Age: 45
Job: IT Teacher
Experience: 10
Achievements: Best Teacher Award 2018
========================
```

Image 14: Press Display Candidate

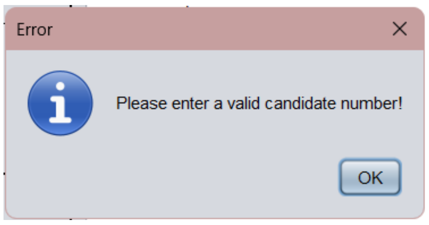
When the admin successfully login to the admin page will show the Jform like in Image 13, when the admin wants to see the candidate that had registered they can click the Display Candidate button to show the candidate information that will look like in the image 14.



```
========================
Candidate No: 2
Name: Salmah
Age: 65
Job: Retired Headmistress
ol and Top-performer in Selang
Award 2018
r
Teacher Award 2020
```

Candidate Information Update

Please Enter Numbers Only!

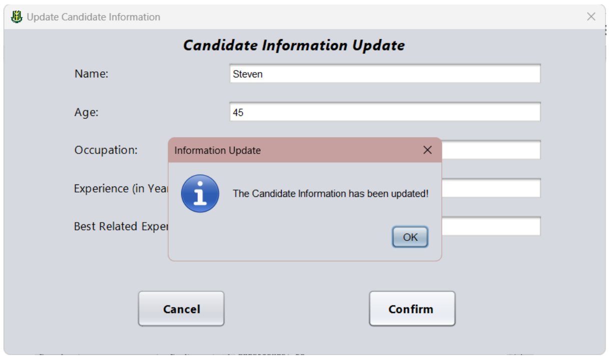


Please enter a valid candidate number!

Image 15: Update Dialog

Image 16: Option Pane

When the candidate wants to update their information, they are required to ask the admin to login to the admin page and click the update button to help them to update the information. When the admin clicks the update button, it will pop out the dialog in the image 15 which requires the admin to key in the candidate number so that they can update the information. If the admin key in the characters and symbol in the text field will show the label that warns the admin to key in the number only; or if the admin key in the candidate number that is not in the list will pop out an option pane which showing the message that saying “Please enter a valid candidate number”, and require them to key in again.


Update Candidate Information

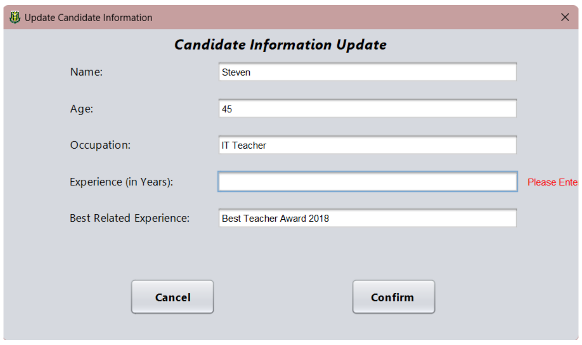
Candidate Information Update

Name:	Steven
Age:	45
Occupation:	
Experience (in Yea	
Best Related Exper	

Information Update

The Candidate Information has been updated!

Image 17: Option Pane


Update Candidate Information

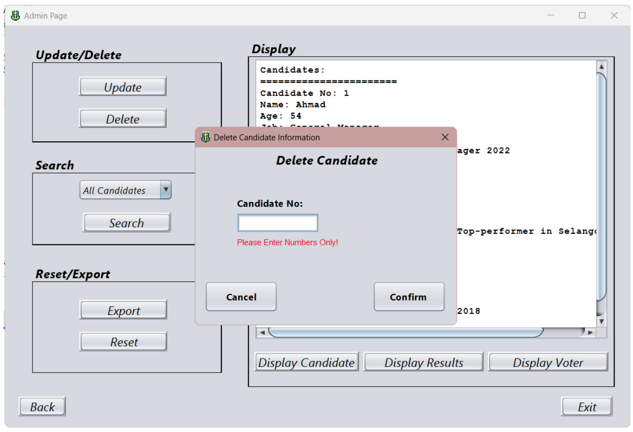
Candidate Information Update

Name:	Steven
Age:	45
Occupation:	IT Teacher
Experience (in Years):	
Best Related Experience:	Best Teacher Award 2018

Please Ente

Image 18: Update Form

Above Image 18 shows when the admin clicks into the number that in the candidate list can bring them into this form for them to update their information. If the candidate wants to update they can change the information here. When they are finally done change the information and click confirm will pop out and option pane which shows the candidate that the information had been updated.


Admin Page

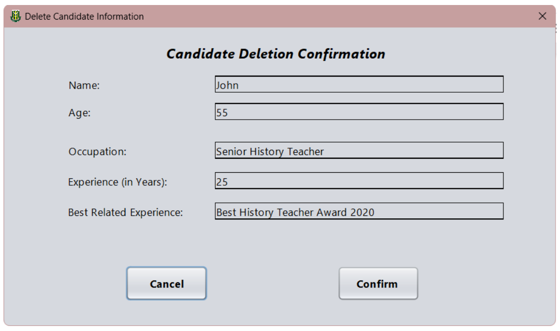
```
Candidates:
========================
Candidate No: 1
Name: Ahmad
Age: 54
ager 2022
Top-performer in Selang
2018
```

Delete Candidate Information

Delete Candidate

Candidate No:

Please Enter Numbers Only!

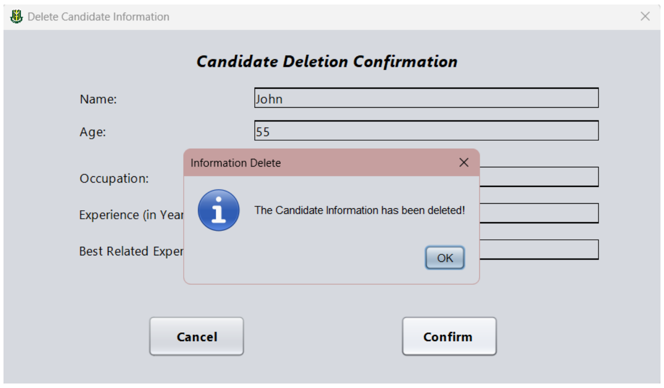

Delete Candidate Information

Candidate Deletion Confirmation

Name:	John
Age:	55
Occupation:	Senior History Teacher
Experience (in Years):	25
Best Related Experience:	Best History Teacher Award 2020

Image 19: Delete Dialog

Image 20: Delete Confirmation


Delete Candidate Information

Candidate Deletion Confirmation

Name:	John
Age:	55
Occupation:	
Experience (in Year	
Best Related Exper	

Information Delete

The Candidate Information has been deleted!

Image 21: Option Pane

Above these 3 images showing when the candidate want to pull out from the election, when admin click the delete button will pop out a dialog like in the image 19 which requires a key in the candidate number that wants to pull out from the PIBG election. When the key in the number correctly will pop out and show the confirmation like in image 20, is the admin key in the wrong number will pop out an option pane just like in the image 16. After the candidate confirms that he/ she wants to pull out from the election will need to press the confirm button and will pop out an option pane just like in image 21 that shows the message "The candidate information has been deleted" which means that the candidate successfully pulled out from the PIBG election.

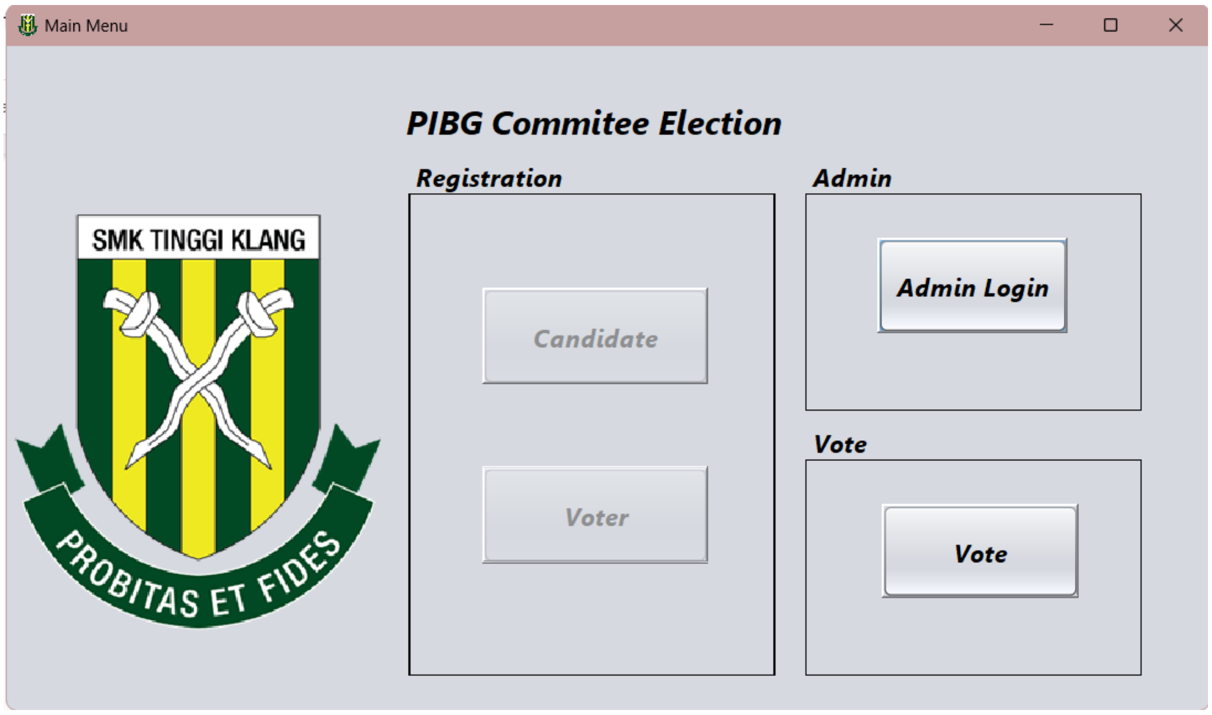
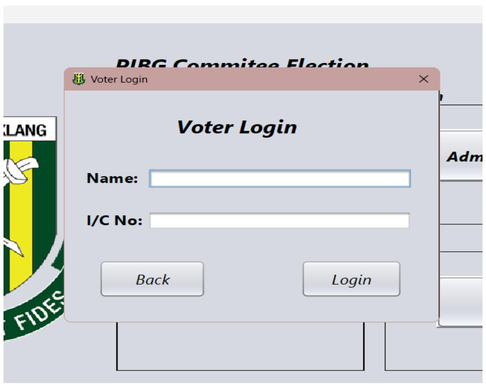
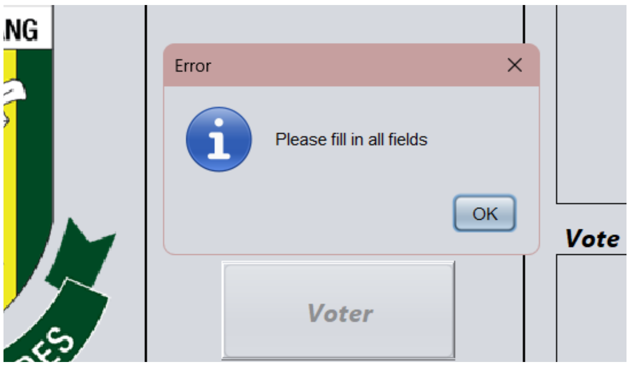


Image 22: Main menu in Voting Day

Above image 22 showing the main menu when it is voting day, the button on the register side will be disabled and the vote section will be enabled and will allow the voter to vote for their desired candidate.





Please fill in all fields

Image 23: Voter login

Image 24: Option pane

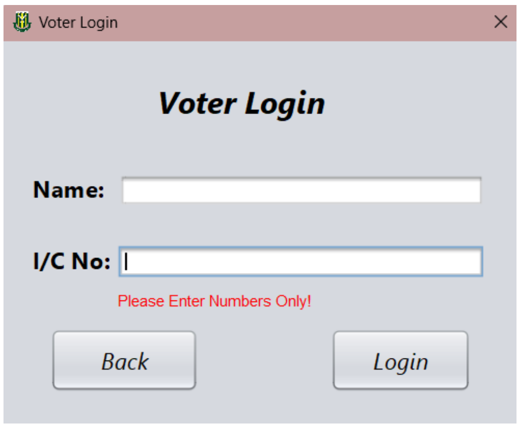
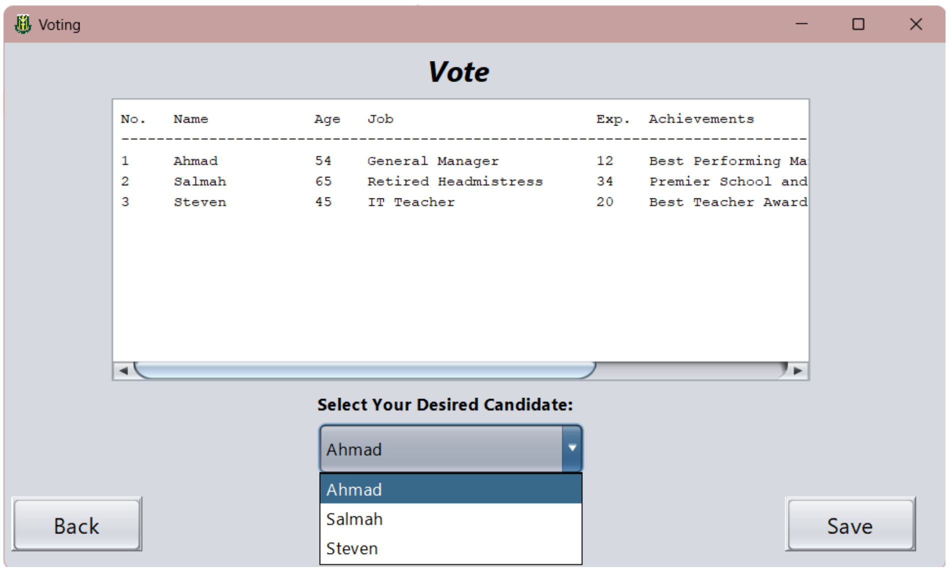
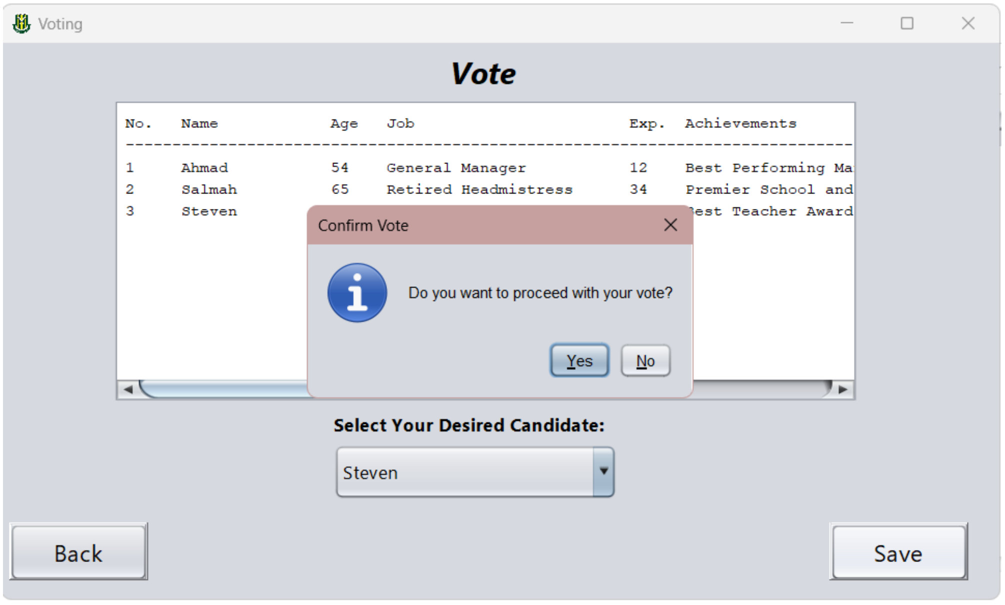


Image 25: Showing the label



```
No.   Name             Age   Job                        Exp.  Achievements
------------------------------------------------------------------------------
1     Ahmad            54    General Manager            12    Best Performing Ma
2     Salmah           65    Retired Headmistress       34    Premier School and
3     Steven           45    IT Teacher                 20    Best Teacher Award
```

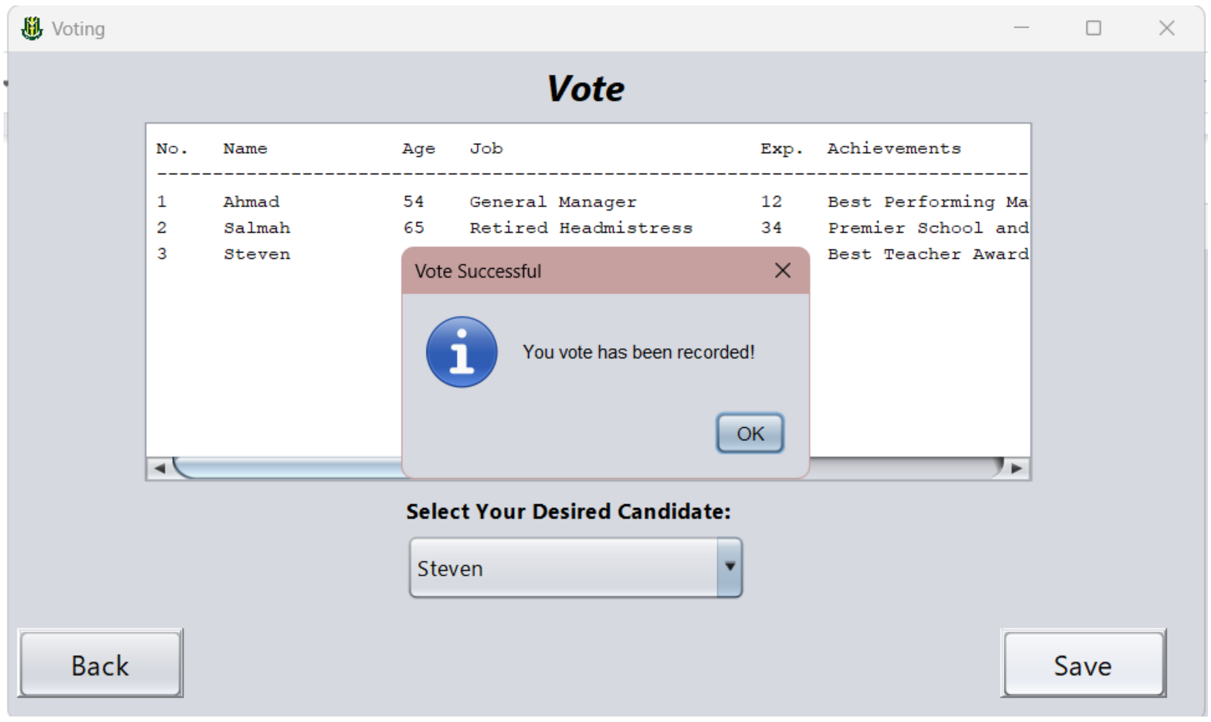
Image 26: Vote Section



```
No.   Name             Age   Job                        Exp.  Achievements
------------------------------------------------------------------------------
1     Ahmad            54    General Manager            12    Best Performing Ma
2     Salmah           65    Retired Headmistress       34    Premier School and
3     Steven                                                  Best Teacher Award
```

Do you want to proceed with your vote?

Image 27: Confirmation Vote


Voting

Vote

```
No.   Name          Age   Job                     Exp.  Achievements
--------------------------------------------------------------------------
1     Ahmad         54    General Manager         12    Best Performing Ma
2     Salmah        65    Retired Headmistress    34    Premier School and
3     Steven                                            Best Teacher Award
```

Vote Successful

You vote has been recorded!

Image 28: Showing the vote has been successful

On the voting day, the voter can use the button “Vote”, to start to vote for the candidate. When the voter clicks the button will show the dialog which requires the voter to login by key in their name and IC number which is like in image 23. If the voter did not fill up all the text field the option pane will pop out like in image 24 which will show the message that need voter to fill up all the text field to login; also if the voter try to fill up the ic number text with a character or symbol will pop out a label tell that voter need to fill up the with number only. Next, when voter successfully login will come out a new frame which is vote section in image 26 which shows the combo box to select desired candidate, back button, save button and text field showing the candidate information. When a voter decides to choose a candidate they can just click the combo box and the combo box will show the name of the candidate to click. The voter after choosing the desired candidate can press the save button and will pop out an option pane which will confirm to the voter to confirm their selection and after they confirm will show another option pane message which is “Your vote has been confirmed” just like in the Image 28.

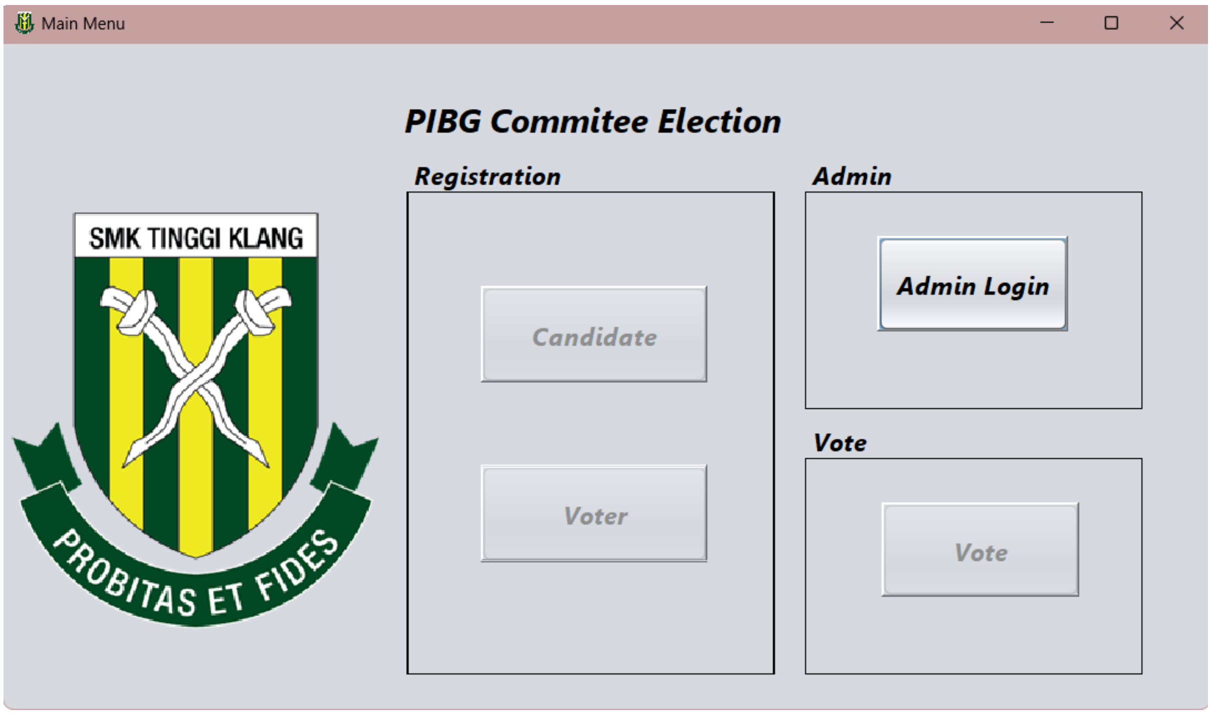
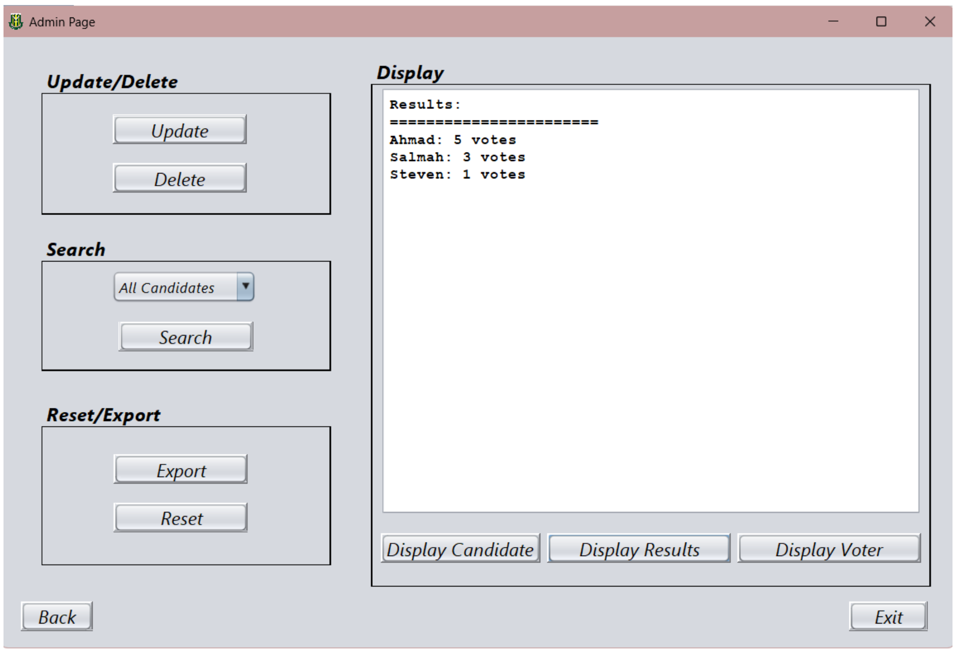
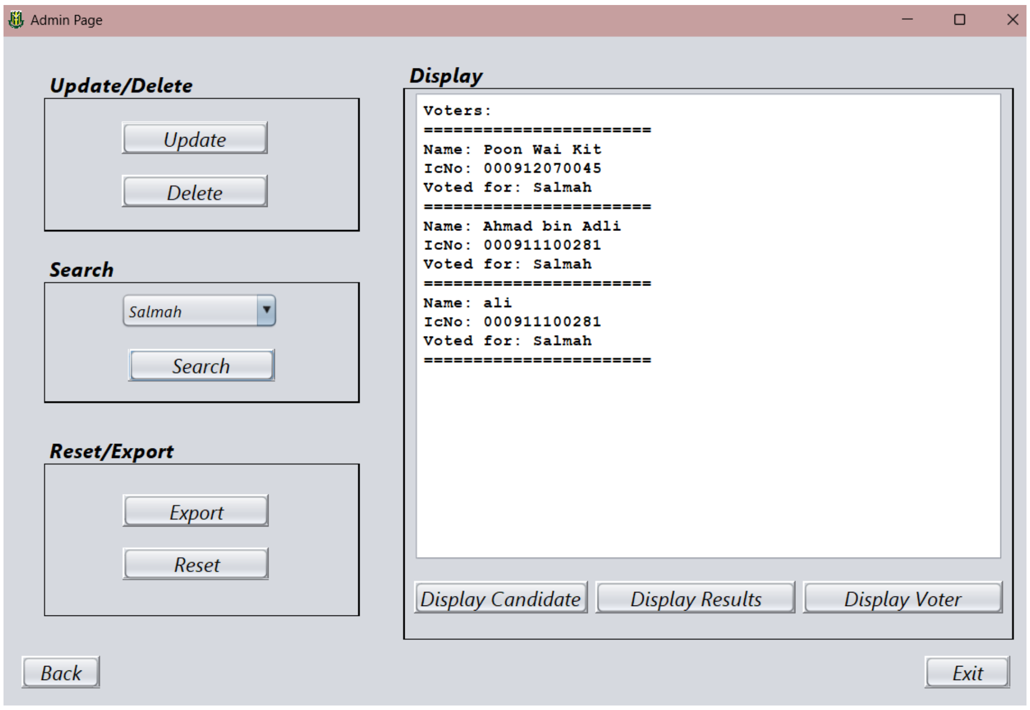


Image 29: Main Menu after the voting day end



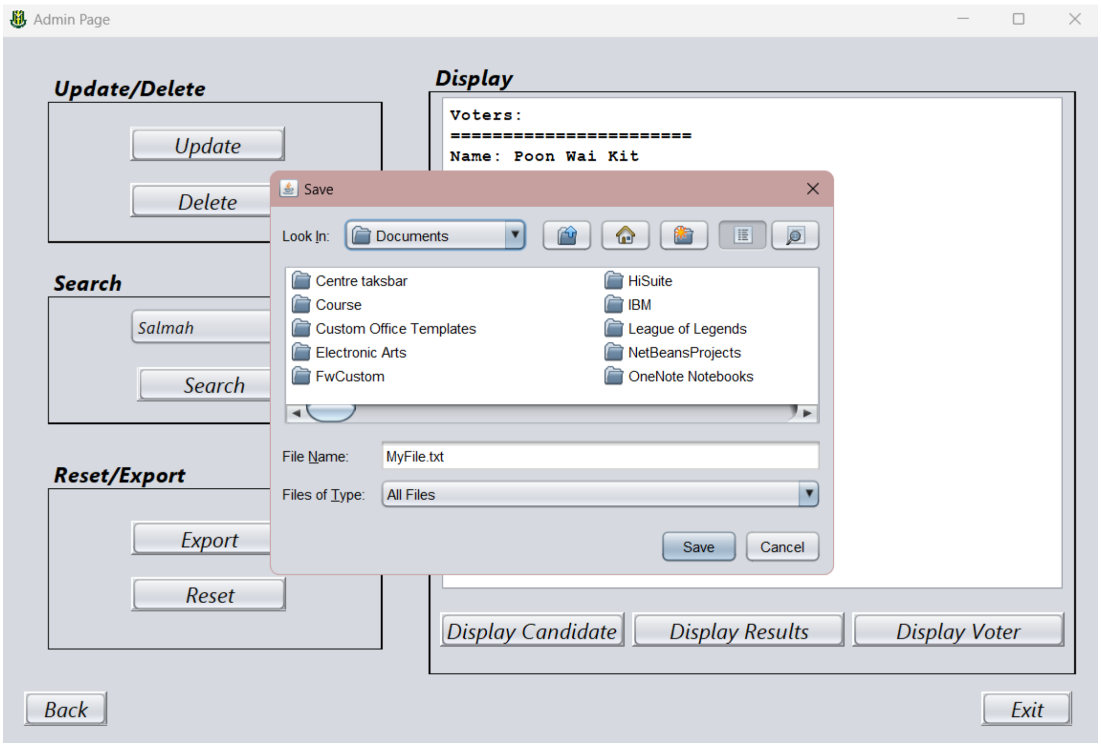
```
Results:
========================
Ahmad:  5 votes
Salmah:  3 votes
Steven:  1 votes
```

Image 30: Display result



```
Voters:
======================
Name: Poon Wai Kit
IcNo: 000912070045
Voted for: Salmah
======================
Name: Ahmad bin Adli
IcNo: 000911100281
Voted for: Salmah
======================
Name: ali
IcNo: 000911100281
Voted for: Salmah
======================
```

Image 31: Display Voter

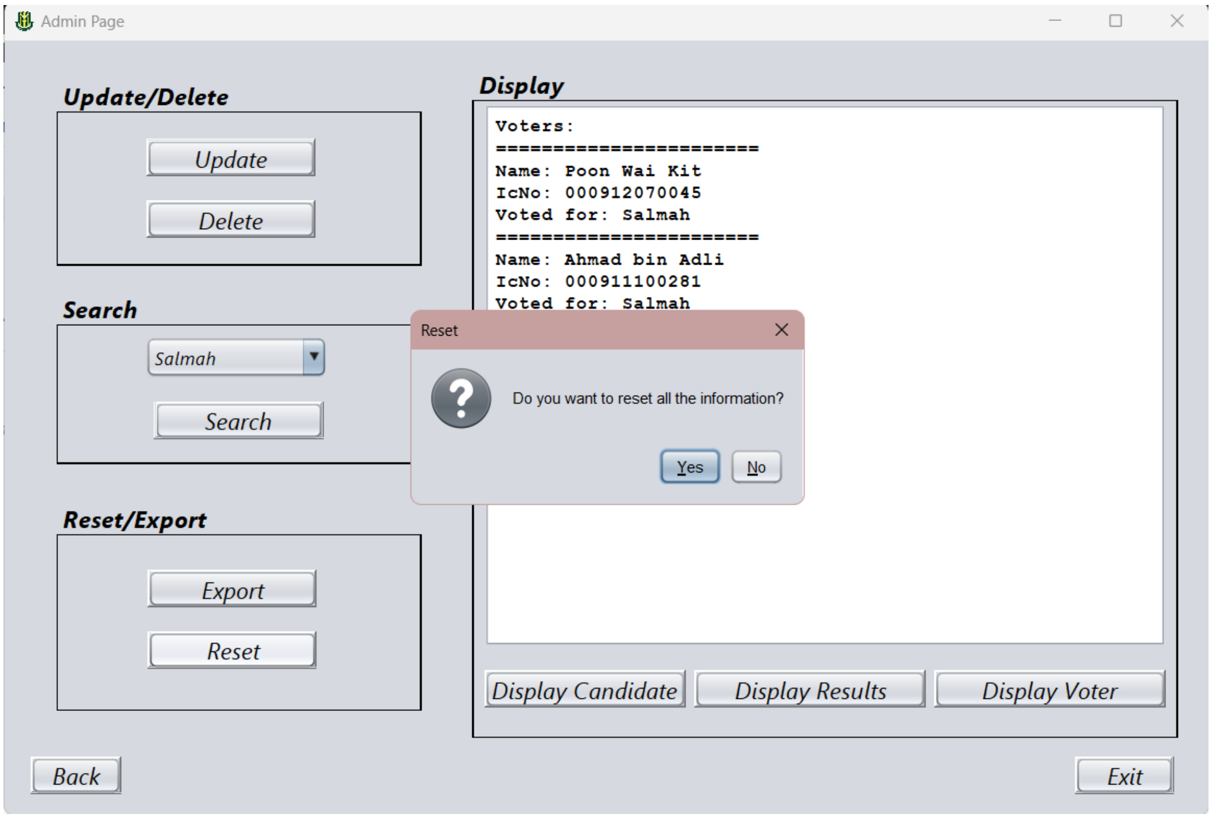


```
Voters:
======================
Name: Poon Wai Kit
```

File Name: MyFile.txt

Files of Type: All Files

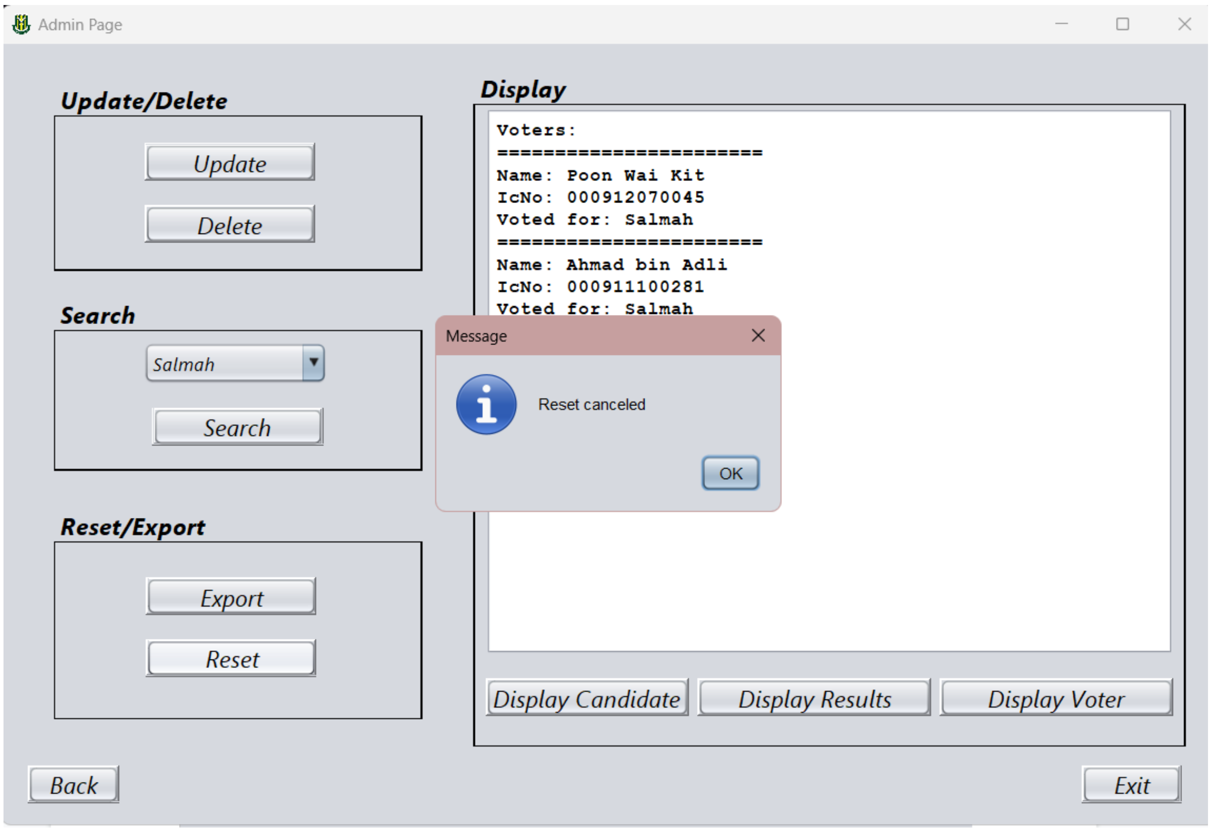
Image 32: Export button



```
Voters:
======================
Name: Poon Wai Kit
IcNo: 000912070045
Voted for: Salmah
======================
Name: Ahmad bin Adli
IcNo: 000911100281
Voted for: Salmah
```

Do you want to reset all the information?

Image 33: Reset



```
Voters:
======================
Name: Poon Wai Kit
IcNo: 000912070045
Voted for: Salmah
======================
Name: Ahmad bin Adli
IcNo: 000911100281
Voted for: Salmah
```

Reset canceled

Image 34: Reset cancel

After the voting day, the button in voter section will be disabled like in Image 29, after that admin can login to the admin page to check the result by click the Display Result button will show the result like in Image 30, if the admin want to check the voter who vote for who the can check it through by clicking display voter button like in the Image 31. After that, if the admin wants to export the PIBG election result they can click the export button and they can save their file to their decided file location like in Image 32. After they export the result, they can reset the whole file which is the candidate info, result and voter information. Admin can do it like in the image 33 by clicking the reset button and clicking confirm if cancel, then the reset will be cancelled. The reset function is for the next election if they want to use it in the future.