Client

```
package server_side;


import java.awt.EventQueue;


import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JButton;
import javax.swing.JTextField;
import javax.swing.border.Border;


import java.awt.Font;
import java.awt.Color;
import java.awt.event.ActionListener;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.Socket;
import java.net.UnknownHostException;
import java.awt.event.ActionEvent;

public class Client implements Runnable {


	static int port=123;
	static String ip="localhost";
	Socket s;
	DataInputStream dis;
	DataOutputStream dos;
```

```
private JFrame frame;
private JTextField textField;
private JTextField textField_1;
JButton btnNewButton;
boolean status=false;
/**
 * Launch the application.
 */
public static void main(String[] args) {
	EventQueue.invokeLater(new Runnable() {
		public void run() {
			try {
				Client window = new Client();
				window.frame.setVisible(true);
			} catch (Exception e) {
				e.printStackTrace();
			}
		}
	});
}

public Client() throws UnknownHostException, IOException {
	initialize();


}
private void initialize() {
	frame = new JFrame();
	frame.getContentPane().setBackground(new Color(103,139,62));
frame.setExtendedState(JFrame.MAXIMIZED_BOTH);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.getContentPane().setLayout(null);

JLabel lblNewLabel = new JLabel("Login Form", JLabel.CENTER);
lblNewLabel.setFont(new Font("Times New Roman", Font.BOLD, 50));
lblNewLabel.setBounds(580,150, 300, 100);
lblNewLabel.setForeground(Color.black);
frame.getContentPane().add(lblNewLabel);

btnNewButton = new JButton("Login");
btnNewButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
                try {
                        createthread();
                } catch (UnknownHostException e1) {
                        // TODO Auto-generated catch block
                        e1.printStackTrace();
                } catch (IOException e1) {
                        // TODO Auto-generated catch block
                        e1.printStackTrace();
                }
        }

});
btnNewButton.setForeground(Color.white);
btnNewButton.setBounds(820,400,80, 30);
btnNewButton.setBackground(new Color(27,110,27));
frame.getContentPane().add(btnNewButton);
```

```
            JLabel lblNewLabel_1 = new JLabel("User Name");
            lblNewLabel_1.setForeground(Color.white);
            lblNewLabel_1.setFont(new Font("Times New Roman", Font.PLAIN, 21));
            lblNewLabel_1.setBounds(570, 280, 100, 50);
            frame.getContentPane().add(lblNewLabel_1);

            JLabel lblNewLabel_2 = new JLabel("Password");
            lblNewLabel_2.setForeground(Color.white);
            lblNewLabel_2.setFont(new Font("Times New Roman", Font.PLAIN, 21));
            lblNewLabel_2.setBounds(570,340, 100, 50);
            frame.getContentPane().add(lblNewLabel_2);

            textField = new JTextField("");
            textField.setBounds(700,290,200,30);
            frame.getContentPane().add(textField);
            textField.setColumns(10);

            textField_1 = new JTextField("");
            textField_1.setBounds(700,350,200,30);
            frame.getContentPane().add(textField_1);
            textField_1.setColumns(10);
    }
    public void createthread() throws UnknownHostException, IOException
    {
       s=new Socket(ip,port);
       Thread t=new Thread(this);
         t.start();
    }
    public void run()
```

```
        {
                String text=textField.getText();
                String text1=textField_1.getText();
                try {
                        dis=new DataInputStream(s.getInputStream());
                  dos=new DataOutputStream(s.getOutputStream());
                } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
            boolean written=false;
                String person="";
                if((text.equals(null) || text.equals(""))  && (text1.equals(null) || 
text1.equals("") ))
                {
                        JOptionPane.showMessageDialog( btnNewButton, "Empty Fields!!", " 
Process Incomplete  ", 0);
                        try {
                                s.close();
                        } catch (IOException e) {
                                e.printStackTrace();
                        }
                }
                else
                {
                        try {
                                dos.writeUTF("login#"+text+"#"+text1);
                                written=true;
                        } catch (IOException e1) {
                                // TODO Auto-generated catch block
        e1.printStackTrace();
    }

}
if(written==true)
{
    try {
        person=new String(dis.readUTF());
    } catch (IOException e1) {
        // TODO Auto-generated catch block
        e1.printStackTrace();
    }
    if(person.equals("admintrue"))
    {
        try{

             Thread.sleep(1000);
            }catch(InterruptedException ex){
                System.out.println(ex);
            }
        frame.dispose();
        Home nw=new Home(s,dis,dos);
        nw.HomeScreen();
    }
    else if(person.equals("votertrue"))
    {
        Home nw=new Home(s,dis,dos);
        status=nw.getstatus();
        if(status==true)
```

```
{
        String partyname="";
        String[] arr=text1.split("@");
        String cnic=arr[1];

        try{

            Thread.sleep(1000);
          }catch(InterruptedException ex){
              System.out.println(ex);
          }
        try {
          dos.writeUTF("voteStatus");
        } catch (Exception e) {
          e.printStackTrace();
        }
        String result = null;
        try{

            Thread.sleep(1200);
          }catch(InterruptedException ex){
              System.out.println(ex);
          }
        try {
          result = new String(dis.readUTF());
        } catch (IOException e) {
          // TODO Auto-generated catch block
          e.printStackTrace();
        }
```

```
                    if(result.equals("nodata"))
                    {
                        JOptionPane.showMessageDialog(null, "No
vote's data!!");
                    }
                    else
                    {
                        String[] array=result.split(",");
                        partyname=(String)array[6];
                        frame.dispose();
                        BalletPaper nww=new
BalletPaper(s,dis,dos,partyname,cnic);
                        nww.ballet();

                    }

                }
                else
                {
                    JOptionPane.showMessageDialog(null, "Voting line is
off!!");
                }
            }
            else if(person.equals("avfalse"))
            {
                JOptionPane.showMessageDialog( btnNewButton, "Login
Failed!!", "   Process Incomplete  ", 0);
                try {
                    s.close();
                } catch (IOException e) {
                    // TODO Auto-generated catch block
```

```
                                                  e.printStackTrace();
                                        }
                              }
                    }
          }
}
```

Home

```
package server_side;


import java.awt.Color;
import java.awt.EventQueue;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JTextField;
import javax.swing.border.LineBorder;
import javax.swing.BorderFactory;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.Socket;
import java.net.UnknownHostException;
import java.util.StringTokenizer;
import java.awt.event.ActionEvent;
import java.awt.Font;
import java.awt.Image;

public class Home implements Runnable{

	private JFrame frame;
	Socket s;
	DataInputStream dis;
	DataOutputStream dos;
	String arrayy[] = {"PML-N","PPP","PTI","JUI-F"};
	String winner="";
	String turnout="";
	String status="";
	String[] array1= null;
	static boolean start;

	/**
	 * Launch the application.
	 */
	public void HomeScreen() {
		EventQueue.invokeLater(new Runnable() {
			public void run() {
				try {
					Home window = new Home(s,dis,dos);
					window.frame.setVisible(true);
				} catch (Exception e) {
					e.printStackTrace();
				}
			}
		});
}

/**
 * Create the application.
 */
public Home(Socket s,DataInputStream dis,DataOutputStream dos) {
    this.s=s;
    this.dos=dos;
    this.dis=dis;
    initialize();
    createthread();
    if(start==true)
    {
        start=true;
    }
    else
        start=false;

}
public boolean getstatus()
{
    return start;
}

private void createthread() {
    // TODO Auto-generated method stub
    Thread t=new Thread(this);
  t.start();
}
```

```
public void run()
{

    try {
        dos.writeUTF("voteStatus");
    } catch (Exception e) {
        e.printStackTrace();
    }
    String result = "";
    try{

          Thread.sleep(3000);
        }catch(InterruptedException ex){
            System.out.println(ex);
        }
    try {
        result = new String(dis.readUTF());
    } catch (IOException e) {
        // TODO Auto-generated catch block
        System.out.println("result unable to get read");
    }
    if(result.equals("nodata"))
    {
        JOptionPane.showMessageDialog(null, "No vote's data!!");
    }
    else
    {
        try{
```

```
                        Thread.sleep(1500);
                    }catch(InterruptedException ex){
                        System.out.println(ex);
                    }
                array1=result.split(",");
                winner=(String)array1[1];
                turnout=(String)array1[3];
                status=(String)array1[5];
                String array=(String)array1[6];
                arrayy=array.split("#");
            }
    }

    /**
     * Initialize the contents of the frame.
     */
    private void initialize() {
        frame = new JFrame();
        frame.getContentPane().setBackground(new Color(103,139,62));
        frame.setExtendedState(JFrame.MAXIMIZED_BOTH);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.getContentPane().setLayout(null);

        JLabel lblHome = new JLabel("Home", JLabel.CENTER);
        lblHome.setFont(new Font("Times New Roman", Font.BOLD, 50));
        lblHome.setForeground(Color.black);
        lblHome.setBounds(580,120, 300, 100);
        frame.getContentPane().add(lblHome);
```

```
JButton btnNewButton = new JButton("Register Candidate");
btnNewButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
                try{

                         Thread.sleep(1000);
                        }catch(InterruptedException ex){
                                System.out.println(ex);
                        }
                frame.dispose();
                CandidateForm1 nw = null;
                try {
                        nw = new CandidateForm1(s,dis,dos,arrayy);
                } catch (IOException e1) {
                        // TODO Auto-generated catch block
                        e1.printStackTrace();
                }
                nw.RegCanScreen();
        }
});
btnNewButton.setBorder(new LineBorder(new Color(27,110,27),5));
btnNewButton.setBackground(Color.white);
btnNewButton.setFont(new Font("Tahoma", Font.BOLD,15));
btnNewButton.setForeground(new Color(27,110,27));
btnNewButton.setBounds(360, 360, 180, 80);
frame.getContentPane().add(btnNewButton);

JButton btnNewButton_1 = new JButton("Register Voter");
btnNewButton_1.setBorder(new LineBorder(new Color(27,110,27),5));
```

```
btnNewButton_1.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
                try{

                           Thread.sleep(1000);
                        }catch(InterruptedException ex){
                                System.out.println(ex);
                        }
                frame.dispose();
                VoterForm vf=new VoterForm(s,dis,dos);
                vf.VoterRegScreen();
        }
});
btnNewButton_1.setBackground(Color.white);
btnNewButton_1.setFont(new Font("Tahoma", Font.BOLD,15));
btnNewButton_1.setForeground(new Color(27,110,27));
btnNewButton_1.setBounds(560,360, 180, 80);
frame.getContentPane().add(btnNewButton_1);

JButton btnNewButton_2 = new JButton("Register Party");
btnNewButton_2.setBorder(new LineBorder(new Color(27,110,27),5));
btnNewButton_2.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
                try{

                           Thread.sleep(2000);
                        }catch(InterruptedException ex){
                                System.out.println(ex);
                        }
                        frame.dispose();
                        RegisterParty rp=new RegisterParty(s,dis,dos);
                        rp.RegPartyScreen();
                }
        });
        btnNewButton_2.setBackground(Color.white);
        btnNewButton_2.setFont(new Font("Tahoma", Font.BOLD,15));
        btnNewButton_2.setForeground(new Color(27,110,27));
        btnNewButton_2.setBounds(760,360, 180, 80);
        frame.getContentPane().add(btnNewButton_2);

        JButton btnNewButton_3 = new JButton("End Voting");
        btnNewButton_3.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {
                        try{

                                  Thread.sleep(500);
                                }catch(InterruptedException ex){
                                        System.out.println(ex);
                                }
                        start=false;
                        JOptionPane.showMessageDialog(null, "Voting has been
Ended");
                        frame.dispose();
                        try {
                                dos.writeUTF("endvote");
                        } catch (IOException e1) {
                                // TODO Auto-generated catch block
                                e1.printStackTrace();
                        }
                        FinalResult fr=new
FinalResult(s,dis,dos,winner,turnout,status);
                        fr.FinalResultScreen();
                }
        });
        btnNewButton_3.setBounds(750,230,180,50);
        btnNewButton_3.setFont(new Font("Tahoma", Font.BOLD,20));
        btnNewButton_3.setForeground(Color.white);
        btnNewButton_3.setBackground(new Color(27,110,27));
        frame.getContentPane().add(btnNewButton_3);

        JButton btnNewButton_6 = new JButton("Start Voting");
        btnNewButton_6.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {
                        start=true;
                        JOptionPane.showMessageDialog(null, "Voting has been
Started");
                }
        });
        btnNewButton_6.setBounds(550,230,180,50);
        btnNewButton_6.setFont(new Font("Tahoma", Font.BOLD,20));
        btnNewButton_6.setForeground(Color.white);
        btnNewButton_6.setBackground(new Color(27,110,27));
        frame.getContentPane().add(btnNewButton_6);

        JButton btnNewButton_4 = new JButton("Vote Status");
        btnNewButton_4.setBorder(new LineBorder(new Color(27,110,27),5));
        btnNewButton_4.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {
                        try{
                                  Thread.sleep(2000);
                                 }catch(InterruptedException ex){
                                         System.out.println(ex);
                                 }
                        frame.dispose();
                        VoterStatus vs=new VoterStatus(s,dis,dos,status);
                        vs.VotesStatusScreen();
                }
        });
        btnNewButton_4.setBackground(Color.white);
        btnNewButton_4.setFont(new Font("Tahoma", Font.BOLD,15));
        btnNewButton_4.setForeground(new Color(27,110,27));
        btnNewButton_4.setBounds(960,360, 180, 80);
        frame.getContentPane().add(btnNewButton_4);

        JButton btnNewButton_5 = new JButton("Log out");
        btnNewButton_5.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {
                        frame.dispose();
                        Client gg;
                        try {
                                 dos.writeUTF("exit");
                                 try{

                                  Thread.sleep(1000);
                                 }catch(InterruptedException ex){
                                         System.out.println(ex);
                                 }
```

```
                    s.close();

                    gg = new Client();

                } catch (UnknownHostException e1) {

                    // TODO Auto-generated catch block

                    e1.printStackTrace();

                } catch (IOException e1) {

                    // TODO Auto-generated catch block

                    e1.printStackTrace();

                }

                Client.main(null);

            }

        });

        btnNewButton_5.setBounds(1035,500,100,50);

        btnNewButton_5.setBorder(BorderFactory.createBevelBorder(0, new Color(27,110,27), Color.white));

        btnNewButton_5.setFont(new Font("Tahoma", Font.BOLD,15));

        btnNewButton_5.setForeground(Color.white);

        btnNewButton_5.setBackground(new Color(27,110,27));

        frame.getContentPane().add(btnNewButton_5);

    }


}
```

Balletpaper

```
package server_side;


import java.awt.EventQueue;


import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JRadioButton;
import java.awt.Font;

import javax.swing.BorderFactory;
import javax.swing.ButtonGroup;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import java.awt.Color;
import java.awt.event.ActionListener;
import java.awt.event.KeyEvent;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.Socket;
import java.net.UnknownHostException;
import java.awt.event.ActionEvent;

public class BalletPaper implements Runnable{
        Socket s;
        DataInputStream dis;
        DataOutputStream dos;
        private JFrame frame;
        ButtonGroup btnGroupH ;
        String partyname;
        JButton btnNewButton;
        String cnic="";
```

```
	/**
	 * Launch the application.
	 */
	public void ballet() {
		EventQueue.invokeLater(new Runnable() {
			public void run() {
				try {
					BalletPaper window = new 
BalletPaper(s,dis,dos,partyname,cnic);
					window.frame.setVisible(true);
				} catch (Exception e) {
					e.printStackTrace();
				}
			}
		});
	}

	/**
	 * Create the application.
	 * @param partyname
	 */
	public BalletPaper(Socket s,DataInputStream dis,DataOutputStream dos, String 
partyname,String cnic) {
		this.s=s;
		this.dos=dos;
		this.dis=dis;
		this.cnic=cnic;
		this.partyname=partyname;
		initialize();
```

```
}

/**
 * Initialize the contents of the frame.
 */
private void initialize() {
        frame = new JFrame();
        frame.setBounds(100, 100, 450, 500);
        frame.getContentPane().setBackground(new Color(103,139,62));
        frame.setExtendedState(JFrame.MAXIMIZED_BOTH);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.getContentPane().setLayout(null);

        JLabel lblNewLabel = new JLabel("E.Ballet Paper");
        lblNewLabel.setFont(new Font("Times New Roman", Font.BOLD, 35));
        lblNewLabel.setBounds(600, 100, 300, 40);
        frame.getContentPane().add(lblNewLabel);

        JPanel panel = new JPanel();
        panel.setFont(new Font("Tahoma", Font.PLAIN, 25));
        panel.setBounds(640, 150, 130, 320);
        panel.setBackground(Color.white);
        frame.getContentPane().add(panel);
        btnGroupH = new ButtonGroup();
        String[] lines = partyname.split("#");
        for (String line : lines) {
           JRadioButton btn = new JRadioButton(line);
           btn.setFont(new Font("Tahoma", Font.BOLD,20));
           btn.setForeground(new Color(27,110,27));
           btn .setActionCommand(line);

           btn.setMnemonic(0);

           btnGroupH.add(btn);

           panel.add(btn);

        }

        frame.add(panel);


        btnNewButton = new JButton("Submit Vote");

        btnNewButton.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                createthread();

            }


        });

        btnNewButton.setForeground(Color.BLUE);

        btnNewButton.setBounds(550, 500, 140, 40);

        btnNewButton.setFont(new Font("Tahoma", Font.BOLD,16));

        btnNewButton.setForeground(Color.white);

        btnNewButton.setBorder(BorderFactory.createBevelBorder(0, new
Color(27,110,27), Color.white));

        btnNewButton.setBackground(new Color(27,110,27));

        frame.getContentPane().add(btnNewButton);


        JButton btnLogOut = new JButton("Log out");

        btnLogOut.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                frame.dispose();

                frame.setVisible(false);


                Client gm;
                try {
                    s.close();
                    gm = new Client();
                } catch (UnknownHostException e1) {
                    // TODO Auto-generated catch block
                    e1.printStackTrace();
                } catch (IOException e1) {
                    // TODO Auto-generated catch block
                    e1.printStackTrace();
                }
                Client.main(null);
            }
        });
        btnLogOut.setBackground(new Color(27,110,27));
        btnLogOut.setForeground(Color.white);
        btnLogOut.setBorder(BorderFactory.createBevelBorder(0, new
Color(27,110,27), Color.white));
        btnLogOut.setFont(new Font("Tahoma", Font.BOLD,15));
        btnLogOut.setBounds(730, 500, 140, 40);
        frame.getContentPane().add(btnLogOut);
    }
    private void createthread() {
        Thread t=new Thread(this);
        t.start();
        // TODO Auto-generated method stub

    }
    public void run()
    {
        String text="";
        boolean written=false;
        if(btnGroupH.isSelected(null))
        {
            JOptionPane.showMessageDialog(btnNewButton, "Please Select the
Party!!","Vote UnSuccessful",0);
        }
        else
        {
            text = btnGroupH.getSelection().getActionCommand().toString();
            try {
                dos.writeUTF("casteVote#"+text+"#"+cnic);
                written=true;
            } catch (IOException e1) {
                // TODO Auto-generated catch block
                e1.printStackTrace();
            }
        }
        try{

             Thread.sleep(1000);
            }catch(InterruptedException ex){
                System.out.println(ex);
            }
        if(written==true)
        {
            String isconfirm=null;
            try {
                isconfirm = new String(dis.readUTF());
            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
            if(isconfirm.equals("svote"))
            {
                JOptionPane.showMessageDialog(btnNewButton, "Vote has
been casted to : "+text,"Vote Successful",0);
                frame.dispose();
                Client gg;
                try {
                    s.close();
                    gg = new Client();
                } catch (UnknownHostException e1) {
                    // TODO Auto-generated catch block
                    e1.printStackTrace();
                } catch (IOException e1) {
                    // TODO Auto-generated catch block
                    e1.printStackTrace();
                }
                Client.main(null);
            }
            else if(isconfirm.equals("uvote")) {
                JOptionPane.showMessageDialog(btnNewButton, "Vote
cannot caste!!","Vote's casting UnSuccessful",0);
            }
            else
            {
                JOptionPane.showMessageDialog(btnNewButton, "Some
problem may have occured..Try Again!!","Vote's casting UnSuccessful",0);
            }
        }
        }
}
```

Voter Register

```
package server_side;


import java.awt.Color;
import java.awt.EventQueue;


import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;


import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.Socket;


import javax.swing.JComboBox;
import javax.swing.JTextField;
import javax.swing.SwingConstants;
import javax.swing.BorderFactory;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JButton;
```

```
public class VoterForm implements Runnable {


	private JFrame frame;
	private JTextField textField;
	private JTextField textField_1;
	private JTextField textField_2;
	private JTextField textField_4;
	private JTextField textField_5;
	private JTextField textField_6;
	static Socket s;
	static DataInputStream dis;
	static DataOutputStream dos;
	JComboBox comboBox_1 ;
	JButton btnBack ;
	JButton btnSave;
	/**
	 * Launch the application.
	 */
	public static void VoterRegScreen() {
		EventQueue.invokeLater(new Runnable() {
			public void run() {
				try {
					VoterForm window=new VoterForm(s,dis,dos);
					window.frame.setVisible(true);
				} catch (Exception e) {
					e.printStackTrace();
				}
			}
		});
	}

	public VoterForm(Socket s, DataInputStream dis, DataOutputStream dos) {
		// TODO Auto-generated constructor stub
		this.s=s;
		this.dos=dos;
		this.dis=dis;
		initialize();
	}



	/**
	 * Initialize the contents of the frame.
	 */
	private void initialize() {
		frame = new JFrame();
		frame.getContentPane().setBackground(new Color(103,139,62));
		frame.setExtendedState(JFrame.MAXIMIZED_BOTH);
		frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		frame.getContentPane().setLayout(null);

		JLabel lblVoterRegistrationForm = new JLabel("Voter Registration Form");
		lblVoterRegistrationForm.setFont(new Font("Times New Roman", Font.BOLD,
30));
		lblVoterRegistrationForm.setForeground(Color.black);
		lblVoterRegistrationForm.setHorizontalAlignment(SwingConstants.CENTER);
		lblVoterRegistrationForm.setBounds(600, 80, 320, 35);
		frame.getContentPane().add(lblVoterRegistrationForm);

		JLabel label = new JLabel("Name");
```

```
label.setFont(new Font("Tahoma", Font.PLAIN, 16));
label.setForeground(Color.white);
label.setBounds(600, 150, 93, 27);
frame.getContentPane().add(label);

JLabel label_1 = new JLabel("Father's Name");
label_1.setFont(new Font("Tahoma", Font.PLAIN, 15));
label_1.setForeground(Color.white);
label_1.setBounds(600, 200,100, 27);
frame.getContentPane().add(label_1);

JLabel label_2 = new JLabel("CNIC");
label_2.setFont(new Font("Tahoma", Font.PLAIN, 16));
label_2.setForeground(Color.white);
label_2.setBounds(600, 250, 80, 27);
frame.getContentPane().add(label_2);

JLabel label_3 = new JLabel("Gender");
label_3.setFont(new Font("Tahoma", Font.PLAIN, 16));
label_3.setForeground(Color.white);
label_3.setBounds(600, 300, 80, 27);
frame.getContentPane().add(label_3);

JLabel label_4 = new JLabel("Date of Birth");
label_4.setFont(new Font("Tahoma", Font.PLAIN, 16));
label_4.setForeground(Color.white);
label_4.setBounds(600, 350, 93, 27);
frame.getContentPane().add(label_4);
```

```
JLabel label_5 = new JLabel("Address");
label_5.setFont(new Font("Tahoma", Font.PLAIN, 16));
label_5.setForeground(Color.white);
label_5.setBounds(600, 400, 80, 27);
frame.getContentPane().add(label_5);

JLabel label_6 = new JLabel("Area");
label_6.setFont(new Font("Tahoma", Font.PLAIN, 16));
label_6.setForeground(Color.white);
label_6.setBounds(600, 450, 80, 27);
frame.getContentPane().add(label_6);



textField = new JTextField();
textField.setFont(new Font("Tahoma", Font.PLAIN, 14));
textField.setColumns(10);
textField.setBounds(720, 150, 154, 20);
frame.getContentPane().add(textField);

textField_1 = new JTextField();
textField_1.setFont(new Font("Tahoma", Font.PLAIN, 14));
textField_1.setColumns(10);
textField_1.setBounds(720, 200, 154, 20);
frame.getContentPane().add(textField_1);

textField_2 = new JTextField();
textField_2.setFont(new Font("Tahoma", Font.PLAIN, 14));
textField_2.setColumns(10);
textField_2.setBounds(720, 250, 154, 20);
```

```
		frame.getContentPane().add(textField_2);


		textField_4 = new JTextField("0000-00-00");
		textField_4.setFont(new Font("Tahoma", Font.PLAIN, 14));
		textField_4.setColumns(10);
		textField_4.setBounds(720, 350, 154, 20);
		frame.getContentPane().add(textField_4);


		textField_5 = new JTextField();
		textField_5.setFont(new Font("Tahoma", Font.PLAIN, 14));
		textField_5.setColumns(10);
		textField_5.setBounds(720, 400, 154, 20);
		frame.getContentPane().add(textField_5);


		textField_6 = new JTextField();
		textField_6.setFont(new Font("Tahoma", Font.PLAIN, 14));
		textField_6.setColumns(10);
		textField_6.setBounds(720, 450, 154, 20);
		frame.getContentPane().add(textField_6);


		btnSave = new JButton("Register");
		btnSave.setBounds(630, 550,95, 40);
		btnSave.setForeground(Color.white);
		btnSave.setBackground(new Color(27,110,27));
		btnSave.setBorder(BorderFactory.createBevelBorder(0, new
Color(27,110,27), Color.white));
		frame.getContentPane().add(btnSave);
		btnSave.addActionListener(new ActionListener() {
			public void actionPerformed(ActionEvent arg0) {
				createthread();
				}
			});

			btnBack = new JButton("Back");
			btnBack.addActionListener(new ActionListener() {
				public void actionPerformed(ActionEvent arg0) {
					frame.dispose();
					Home nw=new Home(s, dis, dos);
					nw.HomeScreen();
				}

			});
			btnBack.setBounds(760,550,95,40);
			btnBack.setForeground(Color.white);
			btnBack.setBorder(BorderFactory.createBevelBorder(0, new
Color(27,110,27), Color.white));
			btnBack.setBackground(new Color(27,110,27));
			frame.getContentPane().add(btnBack);


			comboBox_1 = new JComboBox();
			comboBox_1.setModel(new DefaultComboBoxModel(new String[] {"Male",
"Female", "Other"}));
			comboBox_1.setToolTipText("Select party");
			comboBox_1.setFont(new Font("Tahoma", Font.PLAIN, 14));
			comboBox_1.setBounds(720, 300, 154, 20);
			frame.getContentPane().add(comboBox_1);
		}
		private void createthread() {
			// TODO Auto-generated method stub
```

```
                    Thread t=new Thread(this);
                    t.start();
            }
            public void run()
            {
                    String text=textField.getText();
                    String text1=textField_1.getText();
                    String text2=textField_2.getText();
                    String text3 = (String)comboBox_1.getSelectedItem();
                    String text4 = textField_4.getText();
                    String text5=textField_5.getText();
                    String text6=textField_6.getText();
                    boolean written=false;
                    if(text.equals(null) || text.equals("") || text1.equals(null) || text1.equals("")
|| text2.equals(null) || text2.equals("") || text3.equals(null) || text3.equals("") ||
text4.equals(null) || text4.equals("") || text5.equals(null)                          ||
text5.equals("") || text6.equals(null) || text6.equals("")  )
                    {
                            JOptionPane.showMessageDialog(btnSave, "Some Fields are
Empty!!","Verification Failed",0);
                    }
                    else
                    {
                            try {

            dos.writeUTF("registerVoter#"+text+"#"+text1+"#"+text2+"#"+text4+"#"+text3+"#"+
text5+"#"+text6);
                                    written=true;
                            } catch (IOException e1) {
                                    // TODO Auto-generated catch block
                                    e1.printStackTrace();
```

```
                }
            }
            try{


                  Thread.sleep(1000);
                }catch(InterruptedException ex){
                    System.out.println(ex);
                }
            if(written==true)
            {
                String isconfirm=null;
                try {
                    isconfirm = new String(dis.readUTF());
                } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
                if(isconfirm.equals("uvregister"))
                {
                    JOptionPane.showMessageDialog(btnSave, "Voter has no
NADRA record!!","Registration Failed",0);
                }
                else if(isconfirm.equals("uacregister"))
                {
                    JOptionPane.showMessageDialog(btnSave, "Voter's age is less
than 18!!","Registration Failed",0);
                }
                else
                {
```

```
                                        JOptionPane.showMessageDialog(btnSave, "Voter's username is : Vote \n Password is :"+ isconfirm,"Registration Successfull !!",0);
                                        frame.dispose();
                                        Home nw=new Home(s, dis, dos);
                                        nw.HomeScreen();
                              }
                    }
          }
}
```

Candidate Register

```
package server_side;


import java.awt.Color;
import java.awt.Component;
import java.awt.EventQueue;
import java.util.Date;
import java.util.StringTokenizer;

import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JSpinner.DateEditor;

import java.awt.Font;
import java.awt.Window;

import javax.swing.JTextField;
import javax.swing.SwingConstants;
import javax.swing.JComboBox;

import javax.swing.BorderFactory;

import javax.swing.ComboBoxModel;

import javax.swing.DefaultComboBoxModel;

import javax.swing.JButton;

import java.awt.event.ActionListener;

import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.io.IOException;

import java.net.Socket;

import java.net.UnknownHostException;

import java.sql.Array;

import java.text.SimpleDateFormat;

import java.awt.event.ActionEvent;


public class CandidateForm1 implements Runnable {


        private JFrame frame;

        private JTextField textField;

        private JTextField textField_1;

        private JTextField textField_2;

        private JTextField textField_3;

        private JTextField textField_5;

        private JTextField textField_6;

        Socket s;

        DataInputStream dis;

        DataOutputStream dos;

        String[] array;

        JComboBox comboBox_1;
```

```
    JComboBox comboBox;

    JButton btnSave;

    JButton btnBack ;


    /**

     * Launch the application.

     */

    public  void RegCanScreen() {

        EventQueue.invokeLater(new Runnable() {

            public void run() {

                try {

                    CandidateForm1 window = new CandidateForm1(s, dis,
dos,array);

                    window.frame.setVisible(true);


                } catch (Exception e) {

                    e.printStackTrace();

                }

            }

        });

    }




    public CandidateForm1(Socket s, DataInputStream dis, DataOutputStream
dos,String[] array2) throws IOException {

        // TODO Auto-generated constructor stub

        this.s=s;

        this.dos=dos;

        this.dis=dis;

        this.array=array2;
```

```
		initialize();
	}



	/**
	 * Initialize the contents of the frame.
	 */
	private void initialize() {
		frame = new JFrame();
		frame.getContentPane().setBackground(new Color(103,139,62));
		frame.setExtendedState(JFrame.MAXIMIZED_BOTH);
		frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		frame.getContentPane().setLayout(null);

		JLabel lblCandidateForm = new JLabel("Candidate Form");
		lblCandidateForm.setFont(new Font("Times New Roman", Font.BOLD, 30));
		lblCandidateForm.setForeground(Color.black);
		lblCandidateForm.setHorizontalAlignment(SwingConstants.CENTER);
		lblCandidateForm.setBounds(600, 80, 300, 35);
		frame.getContentPane().add(lblCandidateForm);

		JLabel lblFathersName = new JLabel("Father's Name");
		lblFathersName.setFont(new Font("Tahoma", Font.PLAIN, 16));
		lblFathersName.setForeground(Color.white);
		lblFathersName.setBounds(600, 200,100, 27);
		frame.getContentPane().add(lblFathersName);

		JLabel lblName = new JLabel("Name");
```

```
lblName.setFont(new Font("Tahoma", Font.PLAIN, 16));
lblName.setForeground(Color.white);
lblName.setBounds(600, 150, 93, 27);
frame.getContentPane().add(lblName);

JLabel lblCnic = new JLabel("CNIC");
lblCnic.setFont(new Font("Tahoma", Font.PLAIN, 16));
lblCnic.setForeground(Color.white);
lblCnic.setBounds(600, 250, 80, 27);
frame.getContentPane().add(lblCnic);

JLabel lblGender = new JLabel("Gender");
lblGender.setFont(new Font("Tahoma", Font.PLAIN, 16));
lblGender.setForeground(Color.white);
lblGender.setBounds(600, 300, 80, 27);
frame.getContentPane().add(lblGender);

JLabel lblDateOfBirth = new JLabel("Date of Birth");
lblDateOfBirth.setFont(new Font("Tahoma", Font.PLAIN, 16));
lblDateOfBirth.setForeground(Color.white);
lblDateOfBirth.setBounds(600, 350, 93, 27);
frame.getContentPane().add(lblDateOfBirth);

JLabel lblAddress = new JLabel("Address");
lblAddress.setFont(new Font("Tahoma", Font.PLAIN, 16));
lblAddress.setForeground(Color.white);
lblAddress.setBounds(600, 400, 80, 27);
frame.getContentPane().add(lblAddress);
```

```
JLabel lblArea = new JLabel("Area");
lblArea.setFont(new Font("Tahoma", Font.PLAIN, 16));
lblArea.setForeground(Color.white);
lblArea.setBounds(600, 450, 80, 27);
frame.getContentPane().add(lblArea);

JLabel lblParty = new JLabel("Party");
lblParty.setFont(new Font("Tahoma", Font.PLAIN, 16));
lblParty.setForeground(Color.white);
lblParty.setBounds(600, 500, 80, 27);
frame.getContentPane().add(lblParty);

textField = new JTextField();
textField.setFont(new Font("Tahoma", Font.PLAIN, 14));
textField.setBounds(720, 150, 160, 20);
frame.getContentPane().add(textField);
textField.setColumns(10);

textField_1 = new JTextField();
textField_1.setFont(new Font("Tahoma", Font.PLAIN, 14));
textField_1.setColumns(10);
textField_1.setBounds(720, 200, 160, 20);
frame.getContentPane().add(textField_1);

textField_2 = new JTextField();
textField_2.setFont(new Font("Tahoma", Font.PLAIN, 14));
textField_2.setColumns(10);
textField_2.setBounds(720, 250, 160, 20);
frame.getContentPane().add(textField_2);
```

```
		textField_3 = new JTextField("0000-00-00");
		textField_3.setFont(new Font("Tahoma", Font.PLAIN, 14));
		textField_3.setColumns(10);

		textField_3.setBounds(720, 300, 160, 20);
		frame.getContentPane().add(textField_3);

		textField_5 = new JTextField();
		textField_5.setFont(new Font("Tahoma", Font.PLAIN, 14));
		textField_5.setColumns(10);
		textField_5.setBounds(720, 400, 160, 20);
		frame.getContentPane().add(textField_5);

		textField_6 = new JTextField();
		textField_6.setFont(new Font("Tahoma", Font.PLAIN, 14));
		textField_6.setColumns(10);
		textField_6.setBounds(720, 450, 160, 20);
		frame.getContentPane().add(textField_6);

		comboBox = new JComboBox(array);
		comboBox.setToolTipText("Select party");
		comboBox.setFont(new Font("Tahoma", Font.PLAIN, 14));
		comboBox.setBounds(720, 500, 160, 20);
		frame.getContentPane().add(comboBox);

		comboBox_1 = new JComboBox();
		comboBox_1.setModel(new DefaultComboBoxModel(new String[] {"Male", 
"Female", "Other"}));
		comboBox_1.setToolTipText("Select party");
		comboBox_1.setFont(new Font("Tahoma", Font.PLAIN, 14));
		comboBox_1.setBounds(720, 350, 160, 20);
		frame.getContentPane().add(comboBox_1);




		btnSave = new JButton("Register");
		btnSave.setBounds(630, 550,95, 40);
		btnSave.setForeground(Color.white);
		btnSave.setBackground(new Color(27,110,27));
		btnSave.setBorder(BorderFactory.createBevelBorder(0, new
Color(27,110,27), Color.white));
		frame.getContentPane().add(btnSave);
		btnSave.addActionListener(new ActionListener() {
			public void actionPerformed(ActionEvent arg0) {
				createthread();



			}




		});

		btnBack = new JButton("Back");
		btnBack.addActionListener(new ActionListener() {
			public void actionPerformed(ActionEvent arg0) {
				frame.dispose();
				Home nw=new Home(s, dis, dos);
				nw.HomeScreen();
```

```
            }


        });

        btnBack.setBounds(760,550,95,40);

        btnBack.setForeground(Color.white);

        btnBack.setBorder(BorderFactory.createBevelBorder(0, new
Color(27,110,27), Color.white));

        btnBack.setBackground(new Color(27,110,27));

        frame.getContentPane().add(btnBack);

    }

    private void createthread() {

        // TODO Auto-generated method stub

        Thread t=new Thread(this);

        t.start();

    }

    public void run()

    {

        String text=textField.getText();

        String text1=textField_1.getText();

        String text2=textField_2.getText();

        String text3 = (String)comboBox_1.getSelectedItem();

        String text4 = textField_3.getText();

        String text5=textField_5.getText();

        String text6=textField_6.getText();

        String text7 = (String)comboBox.getSelectedItem();

        boolean written=false;

        if(text.equals(null) || text.equals("") || text1.equals(null) || text1.equals("")
|| text2.equals(null) || text2.equals("") || text3.equals(null) || text3.equals("") ||
text4.equals(null) || text4.equals("") || text5.equals(null)                    ||
text5.equals("") || text6.equals(null) || text6.equals("") || text7.equals(null) ||
text7.equals("")  )
```

```
			{
				JOptionPane.showMessageDialog(btnSave, "Some Fields are 
Empty!!","Verification Failed",0);
			}
			else
			{
				try {
	
	dos.writeUTF("registerCandidate#"+text+"#"+text1+"#"+text2+"#"+text4+"#"+text3+
"#"+text5+"#"+text6+"#"+text7);
					written=true;
				} catch (IOException e1) {
					// TODO Auto-generated catch block
					e1.printStackTrace();
				}
			}
			try{
				
				  Thread.sleep(3000);
				}catch(InterruptedException ex){
					System.out.println(ex);
				}
			if(written==true)
			{
				String isconfirm=null;
				try {
					isconfirm = new String(dis.readUTF());
				} catch (IOException e) {
					// TODO Auto-generated catch block
					e.printStackTrace();
```

```
            }
            if(isconfirm.equals("scregister"))
            {
                JOptionPane.showMessageDialog(btnSave, "Candidate registered successfully!!","Registration Successfull",0);
                frame.dispose();
                Home nw=new Home(s, dis, dos);
                nw.HomeScreen();


            }
            else if(isconfirm.equals("ucregister"))
            {
                JOptionPane.showMessageDialog(btnSave, "Candidate is not registered voter!!","Registration Failed",0);
            }
            else if(isconfirm.equals("uacregister"))
            {
                JOptionPane.showMessageDialog(btnSave, "Candidate age is less than 35!!","Registration Failed",0);
            }
            else
                JOptionPane.showMessageDialog(btnSave, "Candidate's registration is unsuccessful!!","Registration Failed",0);
        }
    }


}
```

Party Register

```
package server_side;
```

```
import java.awt.EventQueue;


import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;


import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Font;
import javax.swing.SwingConstants;
import javax.swing.JTextField;
import javax.swing.BorderFactory;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.Socket;
import java.awt.event.ActionEvent;

public class RegisterParty implements Runnable{


        private JFrame frame;
        private JTextField textField;
        static Socket s;
        static DataInputStream dis;
        static DataOutputStream dos;
        JButton btnRegister;
```

```
/**
 * Launch the application.
 */
public static void RegPartyScreen() {
	EventQueue.invokeLater(new Runnable() {
		public void run() {
			try {
				RegisterParty window = new RegisterParty(s,dis,dos);
				window.frame.setVisible(true);
			} catch (Exception e) {
				e.printStackTrace();
			}
		}
	});
}

public RegisterParty(Socket s, DataInputStream dis, DataOutputStream dos) {
	// TODO Auto-generated constructor stub
	this.s=s;
	this.dos=dos;
	this.dis=dis;
	initialize();
}

/**
 * Initialize the contents of the frame.
 */
private void initialize() {
	frame = new JFrame();
```

```
        frame.getContentPane().setBackground(new Color(103,139,62));
        frame.setExtendedState(JFrame.MAXIMIZED_BOTH);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.getContentPane().setLayout(null);

        JLabel lblRegisterParty = new JLabel("Register Party");
        lblRegisterParty.setHorizontalAlignment(SwingConstants.CENTER);
        lblRegisterParty.setFont(new Font("Times New Roman", Font.BOLD, 35));
        lblRegisterParty.setForeground(Color.black);
        lblRegisterParty.setBounds(620,220, 230, 40);
        frame.getContentPane().add(lblRegisterParty);

        JLabel lblPartyName = new JLabel("Party Name:");
        lblPartyName.setFont(new Font("Tahoma", Font.PLAIN, 20));
        lblPartyName.setBounds(590, 300, 120, 30);
        lblPartyName.setForeground(Color.white);
        frame.getContentPane().add(lblPartyName);

        textField = new JTextField();
        textField.setBounds(720, 305, 170, 30);
        frame.getContentPane().add(textField);
        textField.setColumns(10);

        btnRegister = new JButton("Register");
        btnRegister.setBounds(630,380, 100, 50);
        btnRegister.setBorder(BorderFactory.createBevelBorder(0, new
Color(27,110,27), Color.white));
        btnRegister.setFont(new Font("Tahoma", Font.BOLD,15));
        btnRegister.setForeground(Color.white);
        btnRegister.setBackground(new Color(27,110,27));
        frame.getContentPane().add(btnRegister);
        btnRegister.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent arg0) {
                createthread();


            }


        });


        JButton btnBack = new JButton("Back");
        btnBack.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                try{


                  Thread.sleep(1000);
                }catch(InterruptedException ex){
                    System.out.println(ex);
                }
                frame.dispose();
                Home nw=new Home(s, dis, dos);
                nw.HomeScreen();
            }
        });
        btnBack.setBounds(760, 380, 100, 50);
        btnBack.setBorder(BorderFactory.createBevelBorder(0, new
Color(27,110,27), Color.white));
        btnBack.setFont(new Font("Tahoma", Font.BOLD,15));
        btnBack.setForeground(Color.white);
        btnBack.setBackground(new Color(27,110,27));
        frame.getContentPane().add(btnBack);
```

```
    }
    private void createthread() {
        // TODO Auto-generated method stub
        Thread t=new Thread(this);
        t.start();
    }
    public void run()
    {
        String text=textField.getText();
        boolean written=false;
        if(text.equals(null) || text.equals(""))
        {
            JOptionPane.showMessageDialog(btnRegister, "Some Fields are
Empty!!","Verification Failed",0);
        }
        else
        {
            try {
                dos.writeUTF("addparty#"+text);
                written=true;
            } catch (IOException e1) {
                // TODO Auto-generated catch block
                e1.printStackTrace();
            }
        }
        try{

             Thread.sleep(1000);
            }catch(InterruptedException ex){
                System.out.println(ex);
            }
        if(written==true)
        {
            String isconfirm=null;
            try {
                isconfirm = new String(dis.readUTF());
            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
            if(isconfirm.equals("spregister"))
            {
                JOptionPane.showMessageDialog(btnRegister, "Party has been Registered!!","Registration Successful",0);
                frame.dispose();
                Home nw=new Home(s, dis, dos);
                nw.HomeScreen();
            }
            else if(isconfirm.equals("upregister")) {
                JOptionPane.showMessageDialog(btnRegister, "Party is already Registered!!","Registration UnSuccessful",0);
            }
            else
            {
                JOptionPane.showMessageDialog(btnRegister, "Some problem may have occured..Try Again!!","Registration UnSuccessful",0);
            }
        }
    }
```

```
}
```

Vote Status

```
package server_side;


import java.awt.Color;
import java.awt.EventQueue;


import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;


import java.awt.Font;
import java.awt.GridLayout;


import javax.swing.SwingConstants;
import javax.swing.JTextField;
import javax.swing.BorderFactory;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.net.Socket;
import java.awt.event.ActionEvent;

public class VoterStatus {


	private JFrame frame;
	private JTextField textField;
```

```
    private JTextField textField_1;
    private JTextField textField_2;
    private JTextField textField_3;
    static Socket s;
    static DataInputStream dis;
    static DataOutputStream dos;
    static String status;

    /**
     * Launch the application.
     */
    public static void VotesStatusScreen() {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    VoterStatus window = new VoterStatus(s,dis,dos,
status);
                    window.frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    public VoterStatus(Socket s, DataInputStream dis, DataOutputStream dos, String
status) {
        // TODO Auto-generated constructor stub
        this.s=s;
        this.dos=dos;
```

```
            this.dis=dis;
            this.status=status;
            initialize();
    }

    /**
     * Initialize the contents of the frame.
     */
    private void initialize() {
            frame = new JFrame();
            frame.getContentPane().setBackground(new Color(103,139,62));
            frame.setExtendedState(JFrame.MAXIMIZED_BOTH);
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            frame.getContentPane().setLayout(null);

            JLabel lblVotesStatus = new JLabel("Votes Status");
            lblVotesStatus.setHorizontalAlignment(SwingConstants.CENTER);
            lblVotesStatus.setFont(new Font("Times New Roman", Font.BOLD, 35));
            lblVotesStatus.setBounds(600, 90, 200, 38);
            frame.getContentPane().add(lblVotesStatus);



            JPanel panel = new JPanel();
            panel.setFont(new Font("Tahoma", Font.PLAIN, 14));
            panel.setBounds(560, 160, 300, 300);
            panel.setBackground(Color.white);
            frame.getContentPane().add(panel);
            int i=170;
            String[] lines = status.split("#");
```

```
                    for (String line : lines) {
                       JLabel label = new JLabel();
                       label.setFont(new Font("Tahoma", Font.BOLD,20));
                       label.setText(line);
                       label.setBounds(100, i, 500, 200);
                       panel.add(label);
                       i=i+20;
                    }

                    JButton btnBack = new JButton("Back");
                    btnBack.addActionListener(new ActionListener() {
                              public void actionPerformed(ActionEvent e) {
                                        try{

                                                   Thread.sleep(1000);
                                                  }catch(InterruptedException ex){
                                                            System.out.println(ex);
                                                  }
                                        frame.dispose();
                                        Home nw=new Home(s, dis, dos);
                                        nw.HomeScreen();
                              }
                    });
                    btnBack.setBounds(670, 500, 95, 50);
                    btnBack.setFont(new Font("Tahoma", Font.BOLD,16));
                    btnBack.setForeground(Color.white);
                    btnBack.setBorder(BorderFactory.createBevelBorder(0, new
Color(27,110,27), Color.white));
                    btnBack.setBackground(new Color(27,110,27));
                    frame.getContentPane().add(btnBack);
```

```
        }

}
```

Final Result

```
package server_side;


import java.awt.Color;

import java.awt.EventQueue;


import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.SwingConstants;

import java.awt.Font;

import javax.swing.JTextField;

import javax.swing.BorderFactory;

import javax.swing.JButton;

import java.awt.event.ActionListener;

import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.io.IOException;

import java.net.Socket;

import java.awt.event.ActionEvent;


public class FinalResult {


        private JFrame frame;

        static Socket s;
```

```
    static DataInputStream dis;

    static DataOutputStream dos;

    static String winner;

    static String turnout;

    static String status;


    /**

     * Launch the application.

     */

    public static void FinalResultScreen() {

        EventQueue.invokeLater(new Runnable() {

            public void run() {

                try {

                    FinalResult window = new
FinalResult(s,dis,dos,winner,turnout,status);

                    window.frame.setVisible(true);

                } catch (Exception e) {

                    e.printStackTrace();

                }

            }

        });

    }



    public FinalResult(Socket s, DataInputStream dis, DataOutputStream dos, String
winner, String turnout,String status) {

        // TODO Auto-generated constructor stub

        this.s=s;

        this.dos=dos;

        this.dis=dis;
```

```
		this.winner=winner;
		this.status=status;
		this.turnout=turnout;
		initialize();
	}


	/**
	 * Initialize the contents of the frame.
	 */
	private void initialize() {
		frame = new JFrame();
		frame.getContentPane().setBackground(new Color(103,139,62));
		frame.setExtendedState(JFrame.MAXIMIZED_BOTH);
		frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		frame.getContentPane().setLayout(null);

		JLabel lblFinalltResu = new JLabel("Final Result");
		lblFinalltResu.setFont(new Font("Times New Roman", Font.BOLD, 35));
		lblFinalltResu.setHorizontalAlignment(SwingConstants.CENTER);
		lblFinalltResu.setForeground(Color.black);
		lblFinalltResu.setBounds(600, 150, 260, 49);
		frame.getContentPane().add(lblFinalltResu);

		JLabel lblWinnerIs = new JLabel("Winner :");
		lblWinnerIs.setFont(new Font("Tahoma", Font.PLAIN, 17));
		lblWinnerIs.setForeground(Color.white);
		lblWinnerIs.setBounds(600,230, 97, 26);
		frame.getContentPane().add(lblWinnerIs);
```

```
JLabel win = new JLabel();
win.setText(winner);
win.setFont(new Font("Tahoma", Font.BOLD, 16));
win.setBounds(720,230, 97, 26);
frame.getContentPane().add(win);

JLabel lblTotalVotes = new JLabel("Toltal Votes :");
lblTotalVotes.setFont(new Font("Tahoma", Font.PLAIN, 17));
lblTotalVotes.setBounds(600,300, 100, 26);
lblTotalVotes.setForeground(Color.white);
frame.getContentPane().add(lblTotalVotes);

JLabel turn = new JLabel();
turn.setText(turnout);
turn.setFont(new Font("Tahoma", Font.BOLD, 14));
turn.setBounds(720,300, 100, 26);
frame.getContentPane().add(turn);

JButton btnPublish = new JButton("Publish");
btnPublish.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        frame.dispose();
        Publish ps=new Publish(s,dis,dos,winner,turnout,status);
        ps.PublishScreen();
    }
});
btnPublish.setBounds(600,400, 95, 45);
btnPublish.setBorder(BorderFactory.createBevelBorder(0, new
Color(27,110,27), Color.white));
```

```
                btnPublish.setForeground(Color.white);
                btnPublish.setBackground(new Color(27,110,27));
                frame.getContentPane().add(btnPublish);

                JButton btnBack = new JButton("Back");
                btnBack.addActionListener(new ActionListener() {
                        public void actionPerformed(ActionEvent e) {
                                try{

                                  Thread.sleep(1000);
                                }catch(InterruptedException ex){
                                        System.out.println(ex);
                                }
                                frame.dispose();
                                Home nw=new Home(s, dis, dos);
                                nw.HomeScreen();
                        }
                });
                btnBack.setBounds(750, 400, 95, 45);
                btnBack.setForeground(Color.white);
                btnBack.setBorder(BorderFactory.createBevelBorder(0, new
Color(27,110,27), Color.white));
                btnBack.setBackground(new Color(27,110,27));
                frame.getContentPane().add(btnBack);
        }

}
```

Publish

```
package server_side;


import java.awt.Color;
import java.awt.EventQueue;


import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.SwingConstants;
import java.awt.Font;


import javax.swing.BorderFactory;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.net.Socket;
import java.awt.event.ActionEvent;

public class Publish {


        private JFrame frame;
        static Socket s;
        static DataInputStream dis;
        static DataOutputStream dos;
        static String status;
        static String winner;
        static String turnout;
```

```
	/**
	 * Launch the application.
	 */
	public void PublishScreen() {
		EventQueue.invokeLater(new Runnable() {
			public void run() {
				try {
					Publish window = new 
Publish(s,dis,dos,winner,turnout,status);
					window.frame.setVisible(true);
				} catch (Exception e) {
					e.printStackTrace();
				}
			}
		});
	}



	public Publish(Socket s, DataInputStream dis, DataOutputStream dos, String winner, 
String turnout, String status) {
		this.s=s;
		this.dos=dos;
		this.dis=dis;
		this.winner=winner;
		this.status=status;
		this.turnout=turnout;
		initialize();
	}
```

```
/**
 * Initialize the contents of the frame.
 */
private void initialize() {
        frame = new JFrame();
        frame.getContentPane().setBackground(new Color(103,139,62));
        frame.setExtendedState(JFrame.MAXIMIZED_BOTH);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.getContentPane().setLayout(null);

        JLabel lblResultHasBeen = new JLabel("Result has been published !");
        lblResultHasBeen.setFont(new Font("Times New Roman", Font.BOLD,30));
        lblResultHasBeen.setHorizontalAlignment(SwingConstants.CENTER);
        lblResultHasBeen.setBounds(600, 100, 350, 52);
        frame.getContentPane().add(lblResultHasBeen);



        JLabel lblWinnerIs = new JLabel("Winner :");
        lblWinnerIs.setFont(new Font("Tahoma", Font.PLAIN, 17));
        lblWinnerIs.setForeground(Color.white);
        lblWinnerIs.setBounds(620,180, 97, 26);
        frame.getContentPane().add(lblWinnerIs);

        JLabel win = new JLabel();
        win.setText(winner);
        win.setFont(new Font("Tahoma", Font.BOLD, 16));
        win.setBounds(740,180, 97, 26);
        frame.getContentPane().add(win);
```

```
JLabel lblTotalVotes = new JLabel("Toltal Votes :");
lblTotalVotes.setFont(new Font("Tahoma", Font.PLAIN, 17));
lblTotalVotes.setBounds(620, 200, 100, 26);
lblTotalVotes.setForeground(Color.white);
frame.getContentPane().add(lblTotalVotes);

JLabel turn = new JLabel();
turn.setText(turnout);
turn.setFont(new Font("Tahoma", Font.BOLD, 14));
turn.setBounds(740,200, 100, 26);
frame.getContentPane().add(turn);

JPanel panel = new JPanel();
panel.setFont(new Font("Tahoma", Font.PLAIN, 14));
panel.setBounds(550, 250, 330, 300);
panel.setBackground(Color.white);
frame.getContentPane().add(panel);
int i=170;
String[] lines = status.split("#");
for (String line : lines) {
   JLabel label = new JLabel();
   label.setFont(new Font("Tahoma", Font.BOLD,20));
   label.setText(line);
   label.setBounds(100, i, 500, 200);
   panel.add(label);
   i=i+20;
}

JButton btnBack = new JButton("Back");
```

```
                    btnBack.addActionListener(new ActionListener() {
                              public void actionPerformed(ActionEvent e) {
                                        try{


                                                   Thread.sleep(1000);
                                                  }catch(InterruptedException ex){
                                                            System.out.println(ex);
                                                  }
                                        frame.dispose();
                                        Home nw=new Home(s, dis, dos);
                                        nw.HomeScreen();
                              }
                    });
                    btnBack.setBounds(700, 600, 95, 30);
                    btnBack.setFont(new Font("Tahoma", Font.BOLD,15));
                    btnBack.setForeground(Color.white);
                    btnBack.setBorder(BorderFactory.createBevelBorder(0, new
Color(27,110,27), Color.white));
                    btnBack.setBackground(new Color(27,110,27));
                    frame.getContentPane().add(btnBack);
          }


}
```

Server

```
package server_side;



import java.io.*;
import java.net.*;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;


import server_side.Server;


import java.awt.EventQueue;

import javax.swing.JFrame;

import javax.swing.JTextField;

import javax.swing.text.JTextComponent;

import javax.swing.JLabel;

import javax.swing.JButton;

import java.awt.Font;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.IOException;

import java.awt.Color;

import javax.swing.JTextArea;

import javax.swing.JScrollBar;

import javax.swing.JScrollPane;

import javax.swing.JInternalFrame;

import java.awt.event.*;

import java.awt.*;

import javax.swing.*;

public class Server implements Runnable , ActionListener

{
```

```
public JFrame frame;
public JTextField textField;
private JTextField textField_1;
JButton btnStart;
JButton btnStop;
static String port;
static String ip;
static String stringg="";
JScrollPane scrollpane;
static JTextArea textArea;
static JTextArea textArea_1;
JPanel p;
String alldata="";
ServerSocket ss;
int counter=0;
boolean voting_time=false;
static int turnout=1;
String result="";
private JScrollPane scrollpane1;
static Server server;
  public Server() {
        initialize();
        }
        /**
        * Initialize the contents of the frame.
        * @param s3
        */
        public void initialize() {
                frame = new JFrame();
```

```
frame.setBounds(100, 100, 450, 436);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.getContentPane().setLayout(null);
textField = new JTextField();
textField.setBounds(203, 60, 86, 20);
frame.getContentPane().add(textField);
textField.setColumns(10);
JLabel lblNewLabel = new JLabel("IP Address");
lblNewLabel.setFont(new Font("Tahoma", Font.PLAIN, 13));
lblNewLabel.setBounds(106, 56, 78, 26);
frame.getContentPane().add(lblNewLabel);
JLabel lblPort = new JLabel("Port");
lblPort.setFont(new Font("Tahoma", Font.PLAIN, 13));
lblPort.setBounds(106, 93, 46, 14);
frame.getContentPane().add(lblPort);
textField_1 = new JTextField();
textField_1.setBounds(203, 91, 86, 20);
frame.getContentPane().add(textField_1);
textField_1.setColumns(10);
btnStart = new JButton("Start");
btnStart.addActionListener(this);
btnStart.setBounds(73, 130, 89, 23);
frame.getContentPane().add(btnStart);
JButton btnStop = new JButton("Stop");
btnStop.setBounds(230, 130, 89, 23);
btnStop.addActionListener(this);
frame.getContentPane().add(btnStop);
btnStop.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
```

```
                                try {
                                                          ss.close();
                                                          textArea.append("\n\n<---Server Stopped ---
>");
                                                } catch (IOException e1) {
                                                          // TODO Auto-generated catch block
                                                          textArea.setText("Server not started..");
                                                }
                               }
                              } );
                    textArea = new JTextArea();
                    textArea.setBackground(Color.WHITE);
                    textArea.setBounds(26, 164, 178, 204);
                    textArea.setLineWrap(true);
                    textArea.setWrapStyleWord(true);
                    textArea.setText(alldata);
                    scrollpane=new
JScrollPane(textArea,JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED,JScrollPane.HORIZONT
AL_SCROLLBAR_NEVER);
                    scrollpane.setBounds(27, 164, 170, 204);
                    frame.getContentPane().add(scrollpane);
                    String turnoutt="";
                    textArea_1 = new JTextArea();
          textArea_1.setBounds(223, 164, 183, 204);
          textArea_1.setLineWrap(true);
          textArea_1.setWrapStyleWord(true);
          textArea_1.setText(turnoutt);
          scrollpane1=new
JScrollPane(textArea_1,JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED,JScrollPane.HORIZO
NTAL_SCROLLBAR_NEVER);
          scrollpane1.setBounds(223, 164, 170, 204);
```

```
frame.getContentPane().add(scrollpane1);
}



@Override
public void actionPerformed(ActionEvent ae)
{

        if( ae.getSource()==btnStart)
        {
                if (textField.getText().isEmpty() && textField_1.getText().isEmpty())
                {
                        textArea.setText("Enter PORT");
                System.out.println("Enter IP and PORT");
                }
        else if(ae.getSource()!=btnStop)
        {
                port=textField_1.getText();
                ip=textField.getText();
                int port1 = Integer.parseInt(port);
        serverstart(port1,ip);
        }
}


}
public void run()
{
    try
    {
```

```
                    while(true){
                counter++;
                Socket s=ss.accept();  //server accept the client connection request
                System.out.println(" >> " + "Client No:" + counter + " started!");
                DataInputStream dis = new DataInputStream(s.getInputStream());
             DataOutputStream dos = new DataOutputStream(s.getOutputStream());
                ClientHandler sct = new ClientHandler(s,dis,dos,counter,server); //send  the
request to a separate thread
                sct.start();
               String turnoutt="\n\n    < ---- Voting turnout : ---- >";
                System.out.print(stringg);
                alldata="\n\n ----------------------\n\n"+" >> " + "Client No:" + counter + "
requested!";


              textArea_1.setText(turnoutt);
                textArea.append(alldata);
              }
             }catch(Exception e){
                             System.out.println("Server closed");
             }


}
          public void serverstart(int port1, String ip1) {
                             // TODO Auto-generated method stub


                    try{




              ss=new ServerSocket(port1);
```

```
    textArea.setText("Server Started...");
    Thread t=new Thread(this);
    t.start();


        }
catch(Exception e){
        textArea.setText("Port not Available...");
   }
        }
public static void execution(String ex)
{
        textArea.append(ex);
}
public static void voteturnout()
{
        textArea_1.setText("\n\n    < ---- Voting turnout : "+(turnout++)+"---- >");
}
public static void endResult() throws SQLException
{
        textArea.append("\n\n<---Voting has Ended--->");
        Database db=new Database();
        String turn=(turnout--)+"";
        String turnoutt=db.select("result", turn);
        String[] res=turnoutt.split(",");
        String end =turnoutt.replace(res[6],"" );
        String endresult=end.replaceAll("[#,]", "");
                textArea_1.setText(endresult);
}
        public static void main(String[] args) throws Exception
    {
        EventQueue.invokeLater(new Runnable()
        {
        public void run()
        {
        try
        {
                        Server window = new Server();
                        window.frame.setVisible(true);
                        window.frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                }
                catch (Exception e)
                {
                e.printStackTrace();
                }
        }
        });

    }
}
```

ClientHANDLER

```
package server_side;


import java.awt.Color;
import java.awt.Font;
import java.io.*;


import java.net.Socket;
import java.net.SocketException;
import java.sql.SQLException;

import java.sql.Statement;


import java.util.StringTokenizer;


import javax.swing.JOptionPane;




class ClientHandler extends Thread {


    final DataInputStream dis;

    final DataOutputStream dos;

    int clientNo;

    Socket s;

    public static int turnout;

    public static boolean Aloggedin=false;

    public static int Vloggedin;

    public static String  voteended="false";

    public Server servers;



    // constructor

    public ClientHandler(Socket s,   DataInputStream dis, DataOutputStream dos,int counter, 
Server servers) {

            super();

        this.dis= dis;

        this.dos = dos;

        this.s = s;

        this.servers=servers;
        this.clientNo=counter;
      }
          @Override
          public void run() {
                    try{
                        DataInputStream dis = new DataInputStream(s.getInputStream());
                        DataOutputStream dos = new DataOutputStream(s.getOutputStream());
                        String clientMessage="";
                        while(!clientMessage.equals("exit")){
                               Database db=new Database();
                               clientMessage=dis.readUTF();
                               if(clientMessage.equals("endvote"))
                               {
                                         Server.endResult();
                               }
                                  System.out.println(clientMessage+"clientmeassage");
                                   StringTokenizer st = new StringTokenizer(clientMessage, "#");
                      String method = st.nextToken();
                      if(method.equals("login") )
                                   {
                                       String v1 = st.nextToken();
                             String v2 = st.nextToken();
                                       login(v1,v2,db);
                                   }
                      else if(method.equals("registerVoter") )
                                   {
                                       String v1 = st.nextToken();
                             String v2 = st.nextToken();
                             String v3 = st.nextToken();
                             String v4 = st.nextToken();
                             String v5 = st.nextToken();
                             String v6 = st.nextToken();
                             String v7 = st.nextToken();
                             registerVoter(v1, v2,v3,v4,v5,v6,v7,db);
                                   }
                      else if(method.equals("registerCandidate") )
                                   {
                                       String v1 = st.nextToken();
                             String v2 = st.nextToken();
                             String v3 = st.nextToken();
                             String v4 = st.nextToken();
                             String v5 = st.nextToken();
                             String v6 = st.nextToken();
                             String v7 = st.nextToken();
                             String v8 = st.nextToken();
                             registerCandidate(v1, v2,v3,v4,v5,v6,v7,v8,db);
                                   }
                      else if(method.equals("voteStatus") )
                                   {
                             voteStatus(db);
                                   }
                      else if(method.equals("casteVote") )
                                   {
                                       String v1 = st.nextToken();
                             String v2 = st.nextToken();
                                       casteVote(v1,v2,db);
                                   }
                      else if(method.equals("addparty") )
```

```
                                   {
                                      String v1 = st.nextToken();
                                      addparty(v1,db);
                                   }
                          else if(method.equals("party") )
                                   {
                                      party(db);
                                   }
                                   db.closed();
                            }
                            Server.execution("\n*Admin loggedOff");
                          System.out.println("Connection closed");
                          dis.close();
                            dos.close();
                            s.close();
                           }catch(Exception ex){
                            System.out.println(ex);
                           }finally{
                            System.out.println("Client : exit!! ");
                           }
             }
             public void login(String arg1,String arg2,Database db) throws IOException,
        SQLException
          {
                        String user=arg1;
                        String password=arg2;
                        if(user.equals("Admin"))
                        {
                              if(password.equals("admin12345"))
                              {
```

```
                Server.execution("\n*Admin LoggedIn Successful");
                dos.writeUTF("admintrue");
                Aloggedin=true;
      Aloggedin=false;
        }
        else
        {
                dos.writeUTF("avfalse");
                Server.execution("\n*Admin LoggedIn Failed");
        }
}
else if(user.equals("Vote"))
{
        String ret=db.select("Vlogin", arg2);
        if (ret.equals("false"))
        {
                dos.writeUTF("avfalse");
                Server.execution("\n*Voter LoggedIn UnSuccessful");
         }
        else {
                dos.writeUTF("votertrue");
                Vloggedin++;
                Server.execution("\n*Voter LoggedIn Successful");
        }
}
else
{
        dos.writeUTF("avfalse");
        error();
```

```
                }
    }
            public void registerVoter(String rv1,String rv2,String rv3,String rv4,String rv5,String
rv6,String rv7,Database db) throws IOException, SQLException
            {
                int age=0;
                boolean cnicfound=false;
                String dob;
                String pass="";
                String ret=db.select("Vregister", rv3);
                StringTokenizer st = new StringTokenizer(ret, "#");
        String returnn = st.nextToken();
                if(ret.equals("false"))
                {
                    dos.writeUTF("uvregister");
                    Server.execution("\n*No match Found from nadra Database");
                }
                if(returnn.equals("true"))
                {
                    dob = st.nextToken();
                    pass = st.nextToken();
                    String then = dob.substring(0, dob.indexOf("-"));
                    long millis=System.currentTimeMillis();
                    java.sql.Date date=new java.sql.Date(millis);
                    String strDate=date.toString();
                    String now=strDate.substring(0, strDate.indexOf("-"));
                    int then1 = Integer.parseInt(then);
                    int now1 = Integer.parseInt(now);
                    age=now1-then1;
                    cnicfound=true;
```

```
		}
		if(cnicfound==true)
		{
			if(age>18)
			{
				dos.writeUTF(pass);
				Server.execution("\n*Voter Registered Successfully");
				db.insert("Vdata",rv1,rv2,rv3,rv4,rv5,rv6,rv7,pass);
			}
			else
			{
				dos.writeUTF("uavregister");
				error();
			}
		}
		else
		{
			error();
		}


	}
	public void registerCandidate(String rv1,String rv2,String rv3,String rv4,String
rv5,String rv6,String rv7,String rv8,Database db) throws SQLException, IOException
	{
		int age=0;
		boolean cnicfound=false;
		String dob;
		String cnic = "";
		String area="";
```

```
        String ret=db.select("Cregister", rv3);
        StringTokenizer st = new StringTokenizer(ret, "#");
String returnn = st.nextToken();
        if(ret.equals("false"))
        {
            dos.writeUTF("ucregister");
            Server.execution("\n*No match Found from voter Database");
        }
        else if(returnn.equals("true"))
        {
            dob=rv4;
            String then = dob.substring(0, dob.indexOf("-"));
            long millis=System.currentTimeMillis();
            java.sql.Date date=new java.sql.Date(millis);
            String strDate=date.toString();
            String now=strDate.substring(0, strDate.indexOf("-"));
            int then1 = Integer.parseInt(then);
            int now1 = Integer.parseInt(now);
            age=now1-then1;
            cnicfound=true;
        }
        if(cnicfound==true)
        {
            if(age>32)
            {
                dos.writeUTF("scregister");
                Server.execution("\n*Candidate Registered Successfully");
                db.insert("Rdata",rv1,rv2,rv3,rv4,rv5,rv6,rv7,rv8);
```

```
                    }
                    else
                    {
                              dos.writeUTF("uacregister");
                              error();
                    }
          }
          else
          {
                    error();
          }
}
public void voteStatus(Database db) throws SQLException, IOException
{
                    String ret=db.select("status", "");
                    String fin=db.select("result", "");
                    if (ret.equals( "false"))
                    {
                              dos.writeUTF("nodata");
                              Server.execution("\n*No vote's data to show");
                    }
                    else {
                              try{

                      Thread.sleep(500);
                    }catch(InterruptedException ex){
                              System.out.println(ex);
                    }
                              dos.writeUTF(fin);
                    }
            }
            public void casteVote(String p,String id,Database db) throws SQLException,
    IOException
            {
                    String party="";
                    int vote=0;
                    String ret=db.select("vote", p);
                    if (ret .equals("false"))
                    {
                            dos.writeUTF("uvote");
                            Server.execution("\n*No party registered");
                            }
                    else {
                                    turnout++;
                                    StringTokenizer st = new StringTokenizer(ret, "#");
                         party = st.nextToken();
                         vote = Integer.parseInt(st.nextToken());
                            }
                    db.update(party, vote);
            //          db.delete(id);
                        dos.writeUTF("svote");
                        Server.voteturnout();
                        Server.execution("\n*Vote Submitted Successfuly");
                        Server.execution("\n*Voter loggedOut!!");
                            Vloggedin--;
            }
            public void addparty(String party,Database db) throws SQLException, IOException
            {
                    String check=db.insertparty(party,0);
```

```
		if(check.equals("false"))
		{
			dos.writeUTF("upregister");
			Server.execution("\n*NParty's registration is unsussessful");
		}
		else
		{
			dos.writeUTF("spregister");
			Server.execution("\n*New Party is registered successfuly");
		}
	}
	public void party(Database db) throws SQLException, IOException
	{
		String ret=db.select("party", "");
		if (ret.equals( "false"))
		{
			dos.writeUTF("nodata");
			Server.execution("\n*No party's data to show");
			}
		else {
			dos.writeUTF(ret);
		}
	}
	public int getTurnout() {
		return turnout;
	}
	public int getVLoggedin() {
		return Vloggedin;
	}
```

```
        public void error()
        {
                Server.execution("\n*Process Failed");
        }
        public boolean getAloggedin()
        {
                return Aloggedin;
        }
}
```

Database

```
package server_side;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class Database {
        Connection con;
        Statement stat;
        String partyn="";
        public Database() throws SQLException
        {
                 try {
                        Class.forName("com.mysql.jdbc.Driver").newInstance();
```

```
    con=DriverManager.getConnection("jdbc:mysql://localhost:3306/server_client","root","12345");
            stat=con.createStatement();
        } catch (InstantiationException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (IllegalAccessException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (ClassNotFoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
    public String select(String check,String tocheck) throws SQLException
    {

        if(check.equals("Vlogin"))
        {
            String sql="SELECT * FROM voter where voterpass='"+tocheck+"'";
            ResultSet rs=stat.executeQuery(sql);
            if (rs.next() == false)
            {
                return "false"; }
            else {
                return "true";
            }
        }
        if(check.equals("Vregister"))
```

```
{
    String area="";
    String cnic="";
    String sql="SELECT * FROM nadra where cnic='"+tocheck+"'";
    ResultSet rs=stat.executeQuery(sql);
    if (rs.next() == false)
    {
        return "false"; }
    else {
        do
        {
        String dob = rs.getString("dob");
        area=rs.getString("area");
        cnic=rs.getString("cnic");
        String pass=area+"@"+cnic;
        return "true#"+dob+"#"+pass;
    }
    while (rs.next());
    }
}
else if(check.equals("Cregister"))
{
    String area="";
    String cnic="";
    String sql="SELECT * FROM voter where cnic='"+tocheck+"'";
    ResultSet rs=stat.executeQuery(sql);
    if (rs.next() == false)
    {
        return "false"; }
        else {
                do
                {
                String dob = rs.getString("dob");
                return "true#"+dob;
        }
        while (rs.next());
        }
}
else if(check.equals("status"))
{
        String sql="SELECT * FROM party ";
        ResultSet rs=stat.executeQuery(sql);
        if (rs.next() == false)
        {
                return "false"; }
        else {
                String executed="*Voting Result:";
                do {
                        String party = rs.getString("partyName");
                        String vote=rs.getString("votes");
              String data=party+" "+vote;
              executed=executed+"\n"+ data;
                }
                while (rs.next());
        return executed;
        }
}
else if(check.equals("vote"))
```

```
{
    String party="";
    int vote=0;
    String sql="SELECT * FROM party ";
    ResultSet rs=stat.executeQuery(sql);
    if (rs.next() == false)
    {
        return "false";
        }
    else {
        do {
            party = rs.getString("partyName");
            vote=rs.getInt("votes");
            if(party.equals(tocheck))
            {
                vote++;
                break;
            }
        }
        while (rs.next() );
        return party+"#"+vote;}
}
else if(check.equals("party"))
{
    String par="";
    String sql="SELECT partyName FROM party ";
    ResultSet rs=stat.executeQuery(sql);
    if (rs.next() == false)
    {
```

```
                    return "false";
                    }
            else {
                    do {
                            par = par+rs.getString("partyName")+"#";
                    }
                    while (rs.next() );
                    return par;}
    }
    else if(check.equals("result"))
    {
            int prev=0;
     String party="";
            int vote=0;
            String winner="";
            String result="";
            String sql="SELECT * FROM party ";
            ResultSet rs=stat.executeQuery(sql);
            if (rs.next() == false)
            {
                    winner="\n*No Votes has been given";
                    }
            else {
                    result="<-----Vote's Result------>#\n";
                    do {
                            party = rs.getString("partyName");
                            vote=rs.getInt("votes");
                            if(vote>prev)
                            {
```

```
                                        prev=vote;
                                        winner=party;
                                }
                                result=result+"\n*     "+party+" - "+vote+"     *#\n";
                        }
                        while (rs.next() ); }

                String partyn=partynn();
                String turnoutt="  <<Voters TurnOut : "+tocheck+" >>"+"\n << Winner
: ,"+winner+",- ,"+prev+", >>"+"\n\n,"+result+","+partyn;
                return turnoutt;
        }
        return "";

}
public String partynn() throws SQLException
{
        String sql1="SELECT partyName FROM party ";
        ResultSet rs1=stat.executeQuery(sql1);
        if (rs1.next() == false)
        {
                }
        else {
                do {
                        partyn = partyn+rs1.getString("partyName")+"#";
                }
                while (rs1.next() );}
        return partyn;
}
public void update(String party,int vote) throws SQLException
```

```
        {
                String query="update party set votes = ? where partyName = ?";
                PreparedStatement preparedStmt = con.prepareStatement(query);
            preparedStmt.setInt   (1, vote);
            preparedStmt.setString(2, party);
            preparedStmt.execute();
        }
        public void delete(String voterid) throws SQLException
        {
                String qu = "delete from voter where cnic = ?";
            PreparedStatement ps = con.prepareStatement(qu);
                   ps.setString(1, voterid);
                   ps.execute();
        }
        public void insert(String in,String rv1,String rv2,String rv3,String rv4,String rv5,String
rv6,String rv7,String voterpassword) throws SQLException
        {
                if(in.equals("Vdata"))
                {
                        String query="INSERT INTO voter
(name,fathername,dob,gender,address,area,cnic,voterpass) VALUES ('"+rv1+"',
'"+rv2+"','"+rv4+"','"+rv5+"','"+rv6+"','"+rv7+"','"+rv3+"','"+voterpassword+"')";
                        stat.execute(query);
                }
                else if(in.equals("Rdata"))
                {
                        String query="INSERT INTO candidates
(name,fathername,dob,gender,address,area,cnic,party) VALUES ('"+rv1+"',
'"+rv2+"','"+rv4+"','"+rv5+"','"+rv6+"','"+rv7+"','"+rv3+"','"+voterpassword+"')";
                        stat.execute(query);
                }
```

```
    }
    public String insertparty(String party,int vote) throws SQLException
    {


            String sql="SELECT * FROM party where partyName='"+party+"'" ;
            ResultSet rs=stat.executeQuery(sql);
            if (rs.next() == false)
            {
                String query="INSERT INTO party (partyName,votes) VALUES
('"+party+"','"+vote+"')";
                stat.execute(query);
                return "true";
                }
            else {
                return "false";
                }
    }
    public void closed() throws SQLException {
        // TODO Auto-generated method stub
        con.close();
    }


}
```