CHAPTER V

IMPLEMENTATION AND TESTING

5.1 Implementation

This phonebook application can input new data, edit exsist data, delete exsist data and search exsist data. Main program of this application is RHSProject.java. After compile the application, the first program do when running is read file txt.

RHSProject.java

```
ReadFileTXT dataTree = new ReadFileTXT();
```

ReadFileTXT.java

```
BufferedReader bacaBarisTXT = new BufferedReader(new
FileReader("contact.txt"));
while(bacaBarisTXT.readLine()!=null)
{
      i++;
}
bacaBarisTXT.close();
```

First count data from file text.

```
      BufferedReader bacaTXT = new BufferedReader(new
FileReader("contact.txt"));
                  String [] item = new String[i];
                  for(int j=0;j<i;j++)
                  {
                        item[j]=bacaTXT.readLine();
                  }
                  bacaTXT.close();
```

Then do looping until las data in file text.

```
DataNode root = new DataNode();
subItem[0]=item[0].split(";");
root.setFirstName(subItem[0][0]);
root.setLastName(subItem[0][1]);
root.setAddress1(subItem[0][2]);
root.setAddress2(subItem[0][3]);
root.setPhoneNumber(subItem[0][4]);
root.setBirthDay(subItem[0][5]);
root.setGender(subItem[0][6]);
root.setAtas(null);
```

Then set first data as root.

```
      int j=1;
      TreeString test = new TreeString();
      while(j<i)
            {
                  subItem[j]=item[j].split(";");
            test.Place(root.getFirstName(),subItem[j]
```

```
[0],root,0,subItem[j][1],subItem[j][2],subItem[j][3],subItem[j]
[4],subItem[j][5],subItem[j][6]);
j++;
}
```

Then send root and other data to make a tree.

After have a tree then make a GUI(Graphical User Interface) to get list name. in this application have three main menu(Insert,Display and Search).

```
DisplayListGui gui2 = new DisplayListGui(root,jumDat);
mainFinalPanel = new JPanel();
mainFinalPanel.add(gui2.getPanel());
glue.add(mainFinalPanel);
makeMenu();
```

For default menu have a DisplayListGui. In this menu is displayed table fill all data from tree sort ascending.

DisplayListGui.java

```
if(urut!=null)
      {
            makeTable(urut.kiri);
            dataList[re][0]=urut.getFirstName();
            dataList[re][1]=urut.getLastName();
            important[re]=urut;
            re++;
            makeTable(urut.kanan);
      }
```

This program to sort ascending from tree.

```
tableList = new JTable(dataList,kepalaList);
tableList.setSelectionMode(ListSelectionModel.SINGLE_INTERVAL_SELE
CTION);
terpilih=tableList.getSelectionModel();
terpilih.addListSelectionListener(new ChoiceListener());
```

Then make a table and fill with sorter data. if one name is clicked then small gui will appear.

```
int first = e.getFirstIndex();
                  int last = e.getLastIndex();
                  int number = terpilih.getAnchorSelectionIndex();
                  if(e.getValueIsAdjusting())
                  {
                        for(int i=first;i<=last;i++)
                        {
                              if(terpilih.isSelectedIndex(i))
                              {
                              DisplaySmallList coba = new
                              DisplaySmallList(important[i]);
                              coba.setVisible(true);
                              }
                        }
                  }
```

In small GUI we can edit exsist data and delete exsist data by press a button.

If edit button is clicked then small GUI will diappear and form GUI will appear.

```
public void actionPerformed(ActionEvent r)
```

```
                {
                        EditGui edit = new EditGui(ini);
                        edit.setVisible(true);
                        setVisible(false);
                }
```

```
public void actionPerformed(ActionEvent e)
            {
                  mainFinalPanel.removeAll();
                  ApPhone gui1 = new ApPhone(jumDat);
                  gui1.setRoot(root);
                  mainFinalPanel.add(gui1.getPanel());
                  mainFinalPanel.updateUI();
            }
```

This code show if choose menu display. In this class shown an empty field. After all field is fiiled and click check button.

```
DataNode newData = new DataNode();
newData.setFirstName(firstName);
newData.setLastName(lastName);
newData.setAddress1(address1);
newData.setAddress2(address2);
newData.setPhoneNumber(phoneNumber);
newData.setBirthDay(birthDay);
newData.setGender(gender);
```

Inserted data will be saved in node.

```
DataNode newCek = new DataNode();
SpellLama cekString1 = new SpellLama();
cekString1.eja(firstName);
newCek.setFirstName(cekString1.getStringBaru());
SpellLama cekString2 = new SpellLama();
cekString2.eja(lastName);
newCek.setLastName(cekString2.getStringBaru());
SpellLama cekString3 = new SpellLama();
cekString3.eja(address1);
newCek.setAddress1(cekString3.getStringBaru());
SpellLama cekString4 = new SpellLama();
cekString4.eja(address2);
newCek.setAddress2(cekString4.getStringBaru());
newCek.setPhoneNumber(phoneNumber);
newCek.setBirthDay(birthDay);
newCek.setGender(gender);
```

Change the inserted data with SpellLama.java to get the new spell and saved in node.

```
CatchDataHitung hitungan = new CatchDataHitung();
hitungan.CatchDataHitung(root,newCek,0,0);
ubah=hitungan.getRootValue();
```

Send node fill new spell and root to calculate the similarity wisth edit distance algorithm.

CheckerList.java

```
while(j<intBaru)
{
	for(i=0;i<intNode;i++)
	{
		if(i==0 && j==0)
		{
			checker[i][j]=0;
		}
		else if(j==0)
		{
			checker[i][j]=i;
		}
		else if(i==0)
		{
			checker[i][j]=j;
		}
		else
		{
			int checkNode=(int) nilaiNode[i];
			int checkBaru=(int) nilaiBaru[j];
			if(checkNode==checkBaru)
			{
				a=checker[i-1][j-1];
			}
			else
			{
				a=checker[i-1][j-1]+1;
			}
			b=checker[i-1][j]+1;
			c=checker[i][j-1]+1;
			if(a<b && a<c)
			{
				checker[i][j]=a;
			}
			else if(a<c && a==b)
			{
				checker[i][j]=a;
			}
			else if(a<b && a==c)
			{
				checker[i][j]=a;
			}
			else if(b<a && b<c)
			{
				checker[i][j]=b;
			}
			else if(b<a && b==c)
```

```
                        {
                              checker[i][j]=b;
                        }
                        else if(b<c && b==a)
                        {
                              checker[i][j]=b;
                        }
                        else if(c<a && c<b)
                        {
                              checker[i][j]=c;
                        }
                        else if(c<b && c==a)
                        {
                              checker[i][j]=c;
                        }
                        else if(c<a && c==b)
                        {
                              checker[i][j]=c;
                        }
                        else if(a==b && b==c)
                        {
                              checker[i][j]=a;
                        }
                  }

            j++;
      }
```

This code show calculation of edit distance algorithm.

5.2 Testing

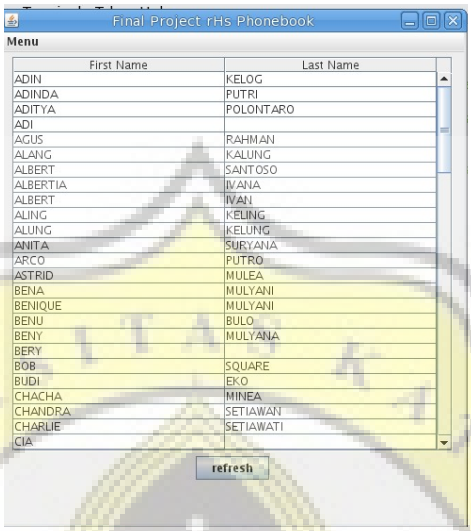

Figure 5.2.1 List Gui

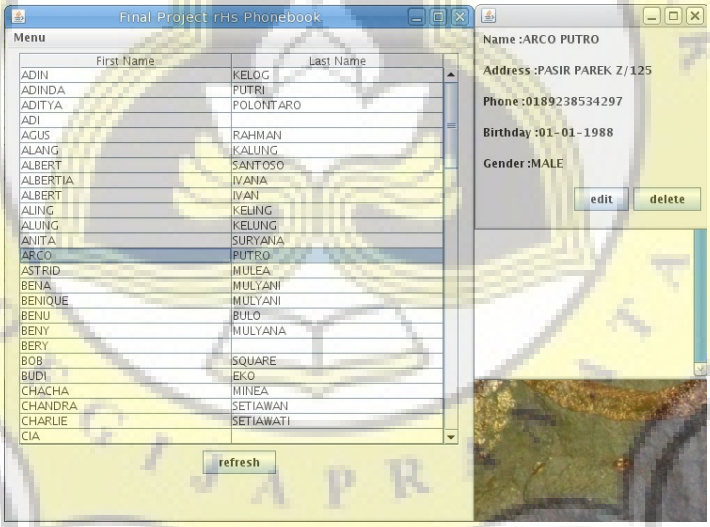

Figure 5.2.2 List Gui with small gui

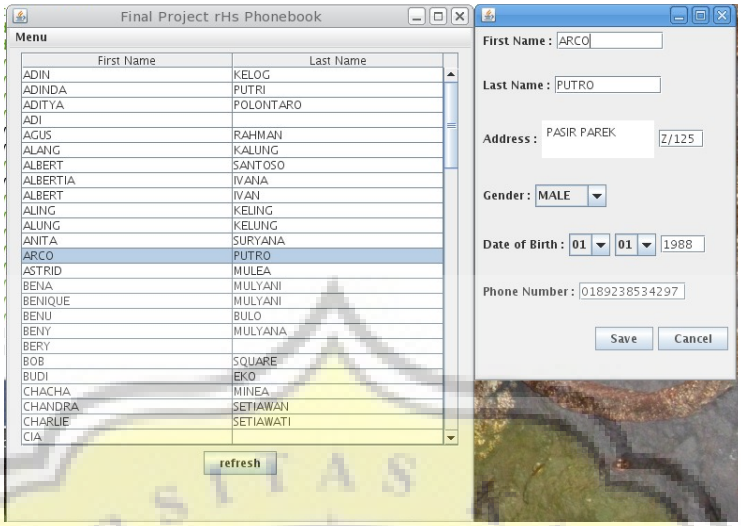

Figure 5.2.3 List Gui with edi gui

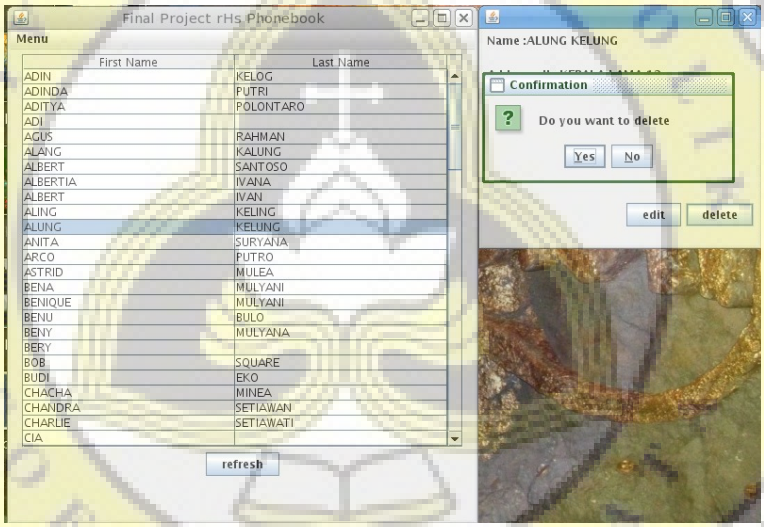

Figure 5.2.4 List Gui if delete button pressed

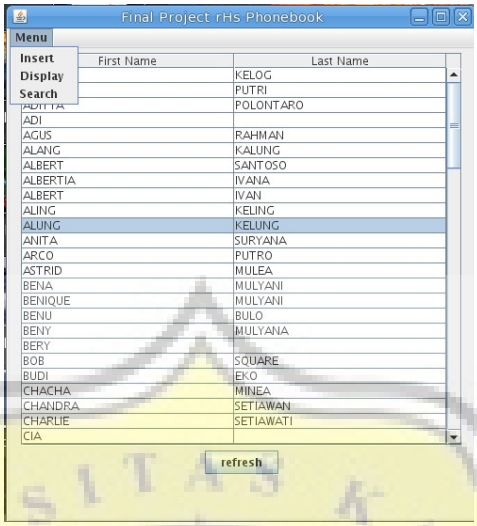

Figure 5.2.5 Menu

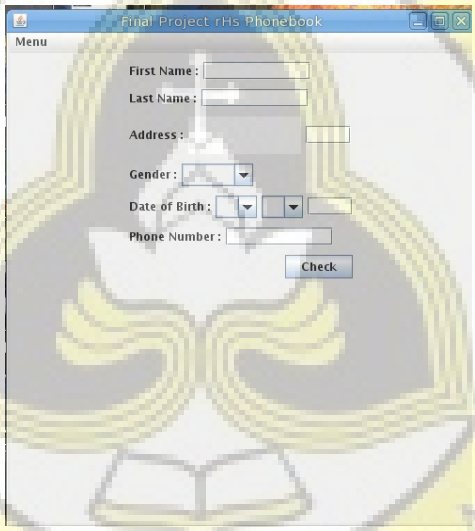

Figure 5.2.6 Insert GUI

Figure 5.2.7 clicked button and data maybe exsist

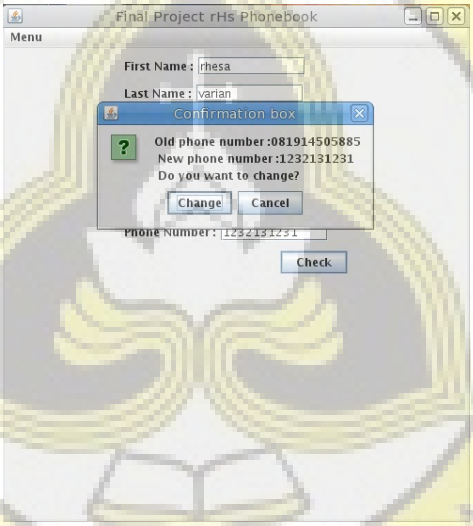

Figure 5.2.8 data inserted exsist and have a new value

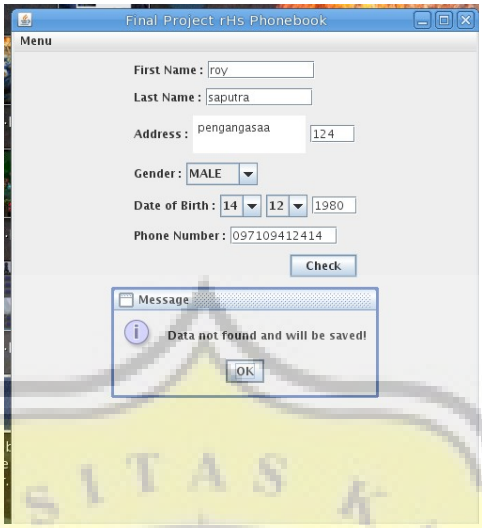

Figure 5.2.9 data inserted is new

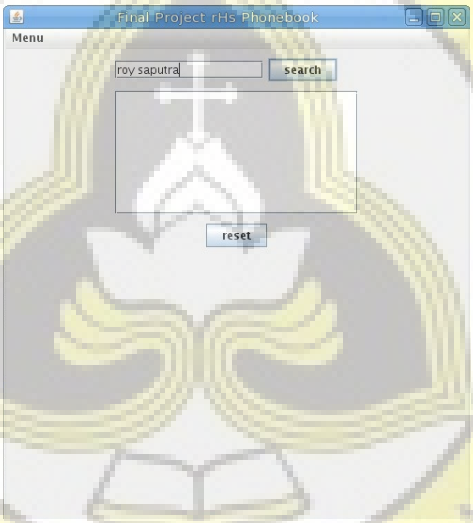

Figure 5.2.10 searching menu

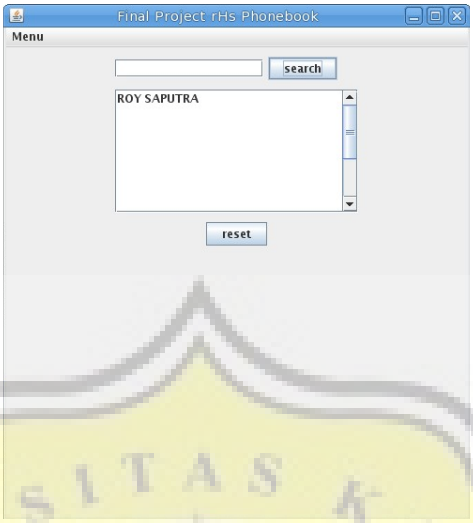

Figure 5.2.11 result searching

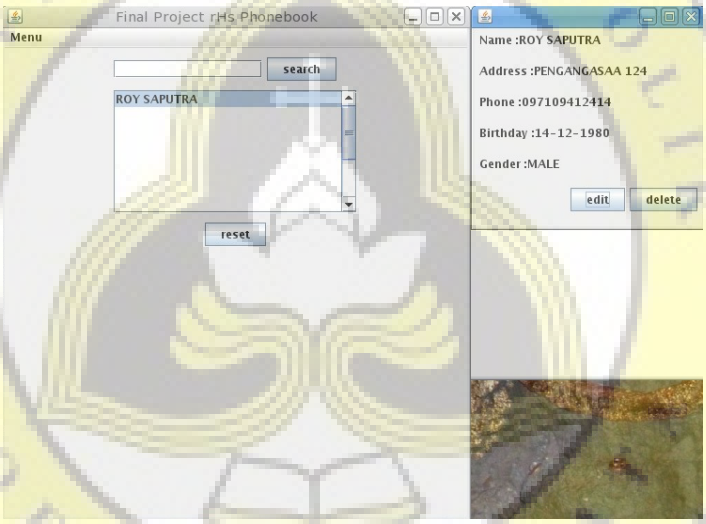

Figure 5.2.12 if result searching is choose