Advance Java Group act 2

SEPTEMBER 5

COMPANY NAME

Authored by: Albert Michael ludick

Logo Name

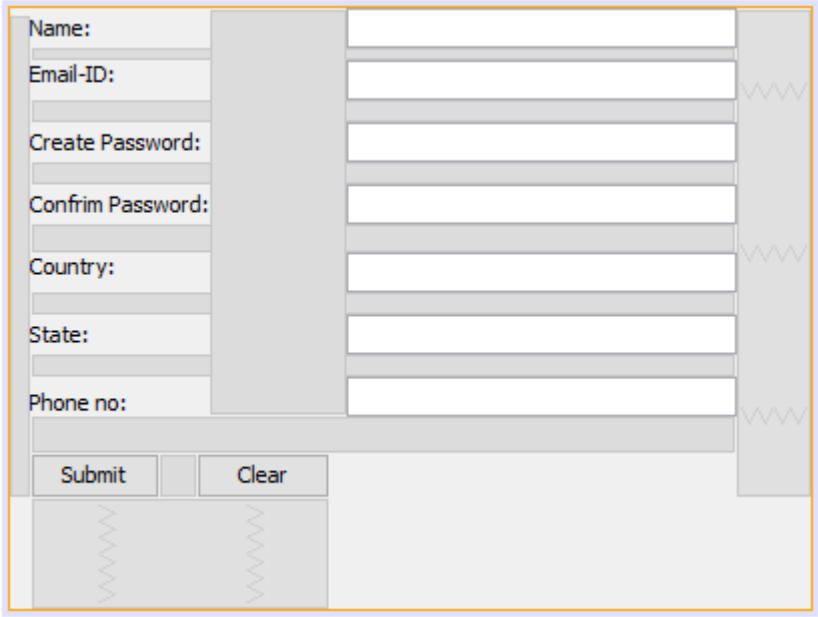
Title Heading

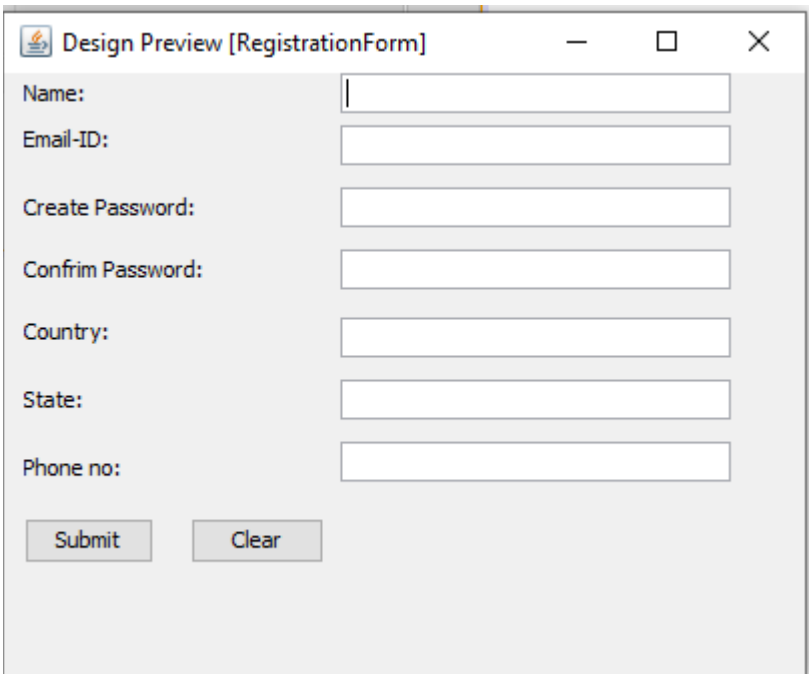
Subtitle Text Here

To get started right away, just tap any placeholder text (such as this) and start typing to replace it with your own.

Want to insert a picture from your files or add a shape, text box, or table? You got it! On the Insert tab of the ribbon, just tap the option you need.

> ***"Find even more easy-to-use tools on the Insert tab, such as to add a hyperlink or insert a comment"***


Name:
Email-ID:
Create Password:
Confrim Password:
Country:
State:
Phone no:
Submit
Clear


Design Preview [RegistrationForm]
Name:
Email-ID:
Create Password:
Confrim Password:
Country:
State:
Phone no:
Submit
Clear

\

Source Code

```
private void btnClearActionPerformed(java.awt.event.ActionEvent evt) {
    //set text fields to empty
    txtName.setText("");
    txtEmail.setText("");
    pfCreate.setText("");
    pfConfrim.setText("");
    txtCountry.setText("");
    txtstate.setText("");
    txtPhone.setText("");
}
```

```
private void btnSubmitActionPerformed(java.awt.event.ActionEvent evt) {
    //textfield validation and messagebox spawn
     if(txtName.getText()!=null
            ||txtEmail.getText()!=null
            ||txtCountry.getText()!=null
            ||txtstate.getText()!=null
            ||txtPhone.getText()!=null){
        if(CheckPassword()&&CheckName()
                &&CheckEmail()&&CheckPhoneNumber()
                &&CheckState()&&CheckCountry()){//&&next
         JOptionPane.showMessageDialog(null,"Data Saved Successfully");
    }else{
            JOptionPane.showMessageDialog(null,"was not "
                + "Saved successfully");
         }
    }else{
        JOptionPane.showMessageDialog(null,"One or More field is empty");
    }
}
```

```java
/***
 * checks if the password fields match
 * @return if the password is correct or not
 */
 private boolean CheckPassword(){
     char[] password=pfCreate.getPassword();
     char[] passConfrim=pfConfrim.getPassword();
     String sPassword="";
     String sPasswordConfrim="";
     if(password.length!=8||password.length < 20){
         if(passConfrim.length!=8||passConfrim.length < 20){
             for(int i =0;i<password.length;i++){
       sPassword += password[i];
     }
      for(int k =0;k<passConfrim.length;k++){
          sPasswordConfrim +=passConfrim[k];
       }
       if(PasswordHasNumbers(sPassword)){
           if(PasswordHasNumbers(sPasswordConfrim)){
               if(sPassword.equals(sPasswordConfrim)){
               correctPassword=true;
           }else{
               JOptionPane.showMessageDialog(null,"the Password do not match");
               correctPassword=false;
           }
         }
         else{
             correctPassword=false;
            JOptionPane.showMessageDialog(null,"the Password is "
                + "not long enough");
         }
     } else{
           correctPassword=false;
           JOptionPane.showMessageDialog(null,"the Password "
               + "is not long enough");
         }
           }else{
           correctPassword=false;
         }
       }else{
           correctPassword=false;
         }
     return correctPassword;
 }
```

```java
/**
 * checks if password contains a number
 * @param password
 * @return if the password contains a single char of numbers
 */
private boolean PasswordHasNumbers(String password){
    Pattern pattern = Pattern.compile("(?=.*[0-9])$");
  Matcher match = pattern.matcher(password);
  if(!match.matches()){
    JOptionPane.showMessageDialog(null,"the password field does not contain"
            + "a single digit");
    PasswordHasLowerCaseLetter(password);
  }
  return match.matches();
} /**
 * checks if password contains a LowerCaseLetter
 * @param password
 * @return if the password contains a single char of LowerCaseLetter
 */
private boolean PasswordHasLowerCaseLetter(String password){
    Pattern pattern = Pattern.compile("(?=.*[a-z])$");
  Matcher match = pattern.matcher(password);
  if(!match.matches()){
    JOptionPane.showMessageDialog(null,"the password field does not contain"
            + "a single lower case");
    PasswordHasUpperCaseLetter(password);
  }
  return match.matches();
} /**
 * checks if password contains a UpperCaseLetter
 * @param password
 * @return if the password contains a single char of UpperCaseLetter
 */
```

```
    return match.matches();
  } /**
   * checks if password contains a LowerCaseLetter
   * @param password
   * @return if the password contains a single char of LowerCaseLetter
   */
  private boolean PasswordHasLowerCaseLetter(String password){
      Pattern pattern = Pattern.compile("(?=.*[a-z])$");
    Matcher match = pattern.matcher(password);
    if(!match.matches()){
      JOptionPane.showMessageDialog(null,"the password field does not contain"
            + "a single lower case");
      PasswordHasUpperCaseLetter(password);
    }
    return match.matches();
  } /**
   * checks if password contains a UpperCaseLetter
   * @param password
   * @return if the password contains a single char of UpperCaseLetter
   */
  private boolean PasswordHasUpperCaseLetter(String password){
      Pattern pattern = Pattern.compile("(?=.*[A-Z])$");
    Matcher match = pattern.matcher(password);
    if(!match.matches()){
      JOptionPane.showMessageDialog(null,"the password field does not contain"
            + "a single upper case");
      PasswordHasSpecialCharLetter(password);
    }
    return match.matches();
  } /**
   * checks if password contains a special char
   * @param password
   * @return if the password contains a single char of special char
   */
   private boolean PasswordHasSpecialCharLetter(String password){
      Pattern pattern = Pattern.compile("(?=.*[@#$%^&-+=()])$");
    Matcher match = pattern.matcher(password);
    if(!match.matches()){
      JOptionPane.showMessageDialog(null,"the password field does not contain"
            + "a single special character");
    }
    return match.matches();
  }
```

```java
}
 /***
 * checks a name field if the input is valid
 * @return if the field is correct or not
 */
private boolean CheckName(){//regex for the name component
    Pattern pattern = Pattern.compile("(.*)(\\d+)(.*)");
    Matcher match = pattern.matcher(txtName.getText());
    if(!match.matches()){
        JOptionPane.showMessageDialog(null,"the name field is "
                                        + "incorrectly validated");
    }
    return match.matches();
}
```

```java
/***
 * checks a email field if the input is valid
 * @return if the field is correct or not
 */
private boolean CheckEmail(){
    Pattern pattern = Pattern.compile("/^(([^<>()\\[\\]\\.,;:\\s@\\\"]"
            + "+(\\.[^<>()\\[\\]\\.,;:\\s@\\\"]+)*)|(\\\".+\\\"))"
            + "@(([^<>()[\\]\\.,;:\\s@\\\"]+\\.)+[^<>()"
            + "[\\]\\.,;:\\s@\\\"]{2,})$/i");
    Matcher match = pattern.matcher(txtEmail.getText());
    if(!match.matches()){
        JOptionPane.showMessageDialog(null,"the name field is "
                                        + "incorrectly validated");
    }
    return match.matches();
}
```

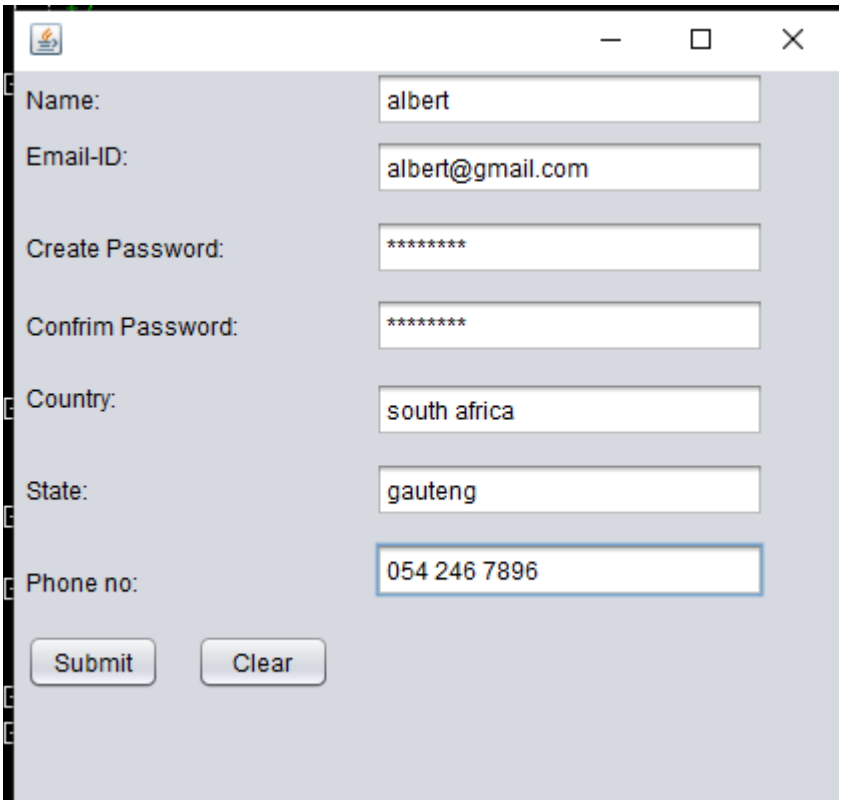
```java
}
/***
 *checks a PhoneNumber field if the input is valid
 * @return if the field is correct or not
 */
private boolean CheckPhoneNumber(){
   Pattern pattern = Pattern.compile("^[+]*[(]{0,1}[0-9]{1,4}[)]"
            + "{0,1}[-\\s\\./0-9]*$");
    Matcher match = pattern.matcher(txtPhone.getText());
    if(!match.matches()){
        JOptionPane.showMessageDialog(null,"the name field is "
                                        + "incorrectly validated");
    }
    return match.matches();
}
/***
```

```java
}
/***
 * checks a State field if the input is valid
 * @return if the field is correct or not
 */
private boolean CheckState(){//regex for the name component
    Pattern pattern = Pattern.compile("(.*)(\\d+)(.*)");
    Matcher match = pattern.matcher(txtstate.getText());
    if(!match.matches()){
        JOptionPane.showMessageDialog(null,"the name field is "
                                                    + "incorrectly validated");
    }
    return match.matches();
}
```

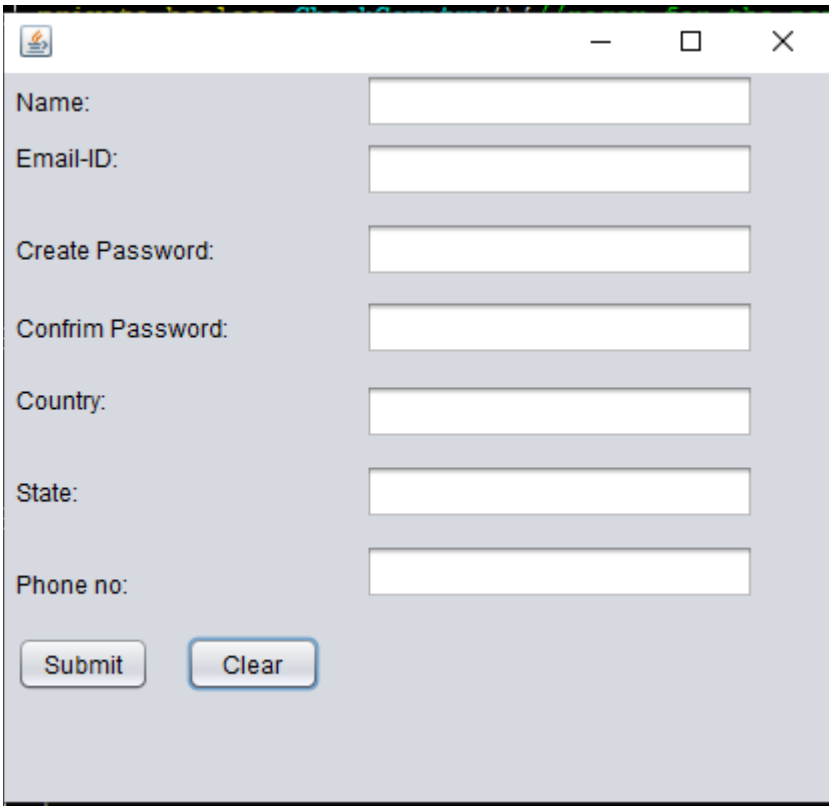
```java
/***
 * checks a Country field if the input is valid
 * @return if the field is correct or not
 */
private boolean CheckCountry(){//regex for the name component
    Pattern pattern = Pattern.compile("(.*)(\\d+)(.*)");
    Matcher match = pattern.matcher(txtCountry.getText());
    if(!match.matches()){
        JOptionPane.showMessageDialog(null,"the name field is "
                                                    + "incorrectly validated");
    }
    return match.matches();
}
/***
```

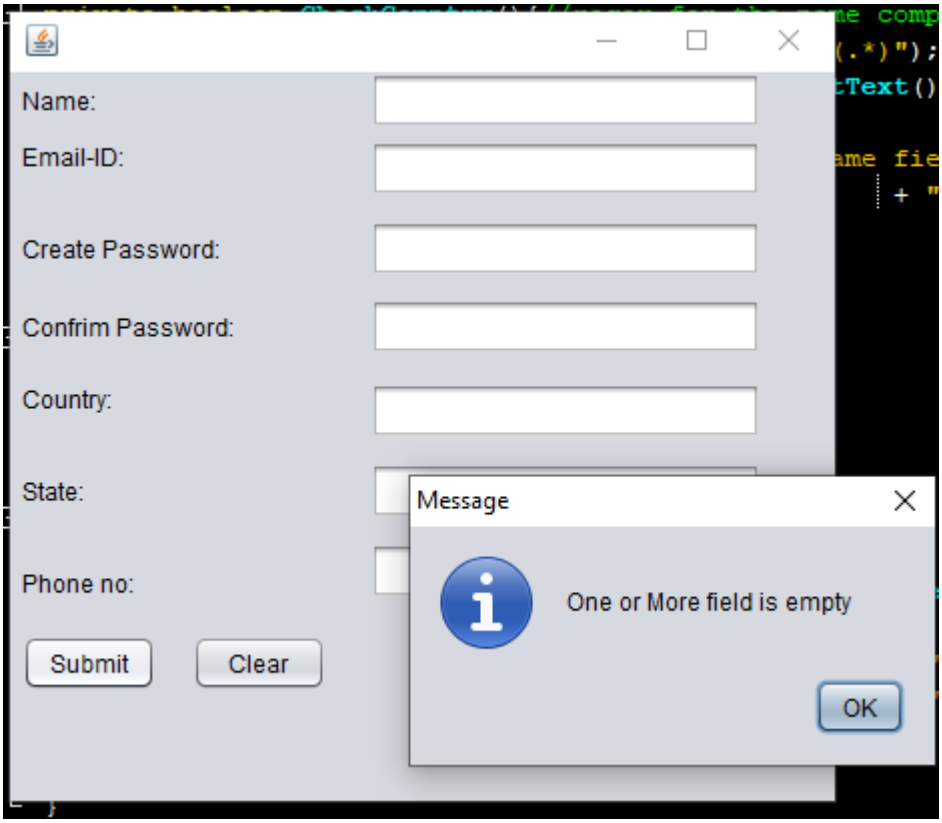
```java
/***
 * checks a random field if the input is valid
 * @return if the field is correct or not
 */
private boolean CheckField(String pattern,String type
        ,javax.swing.JTextField Control){
    Pattern patternToMacth = Pattern.compile(pattern);
    Matcher match = patternToMacth.matcher(Control.getText());
    if(!match.matches()){
        JOptionPane.showMessageDialog(null,"the "+type+" field is "
                                                    + "incorrectly validated");
    }
    return match.matches();
}
```

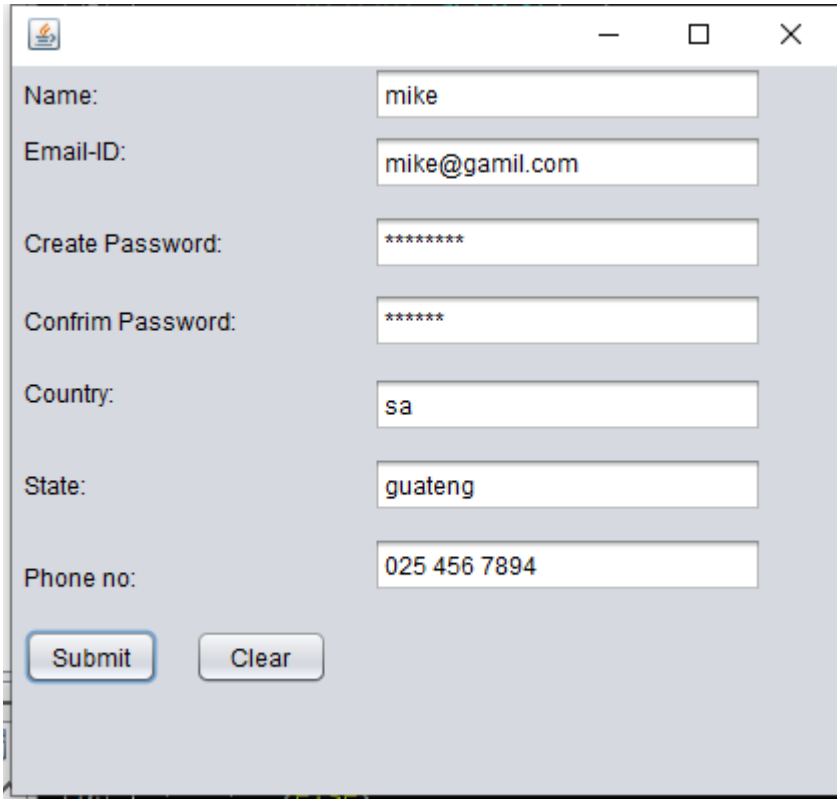
Runnig application

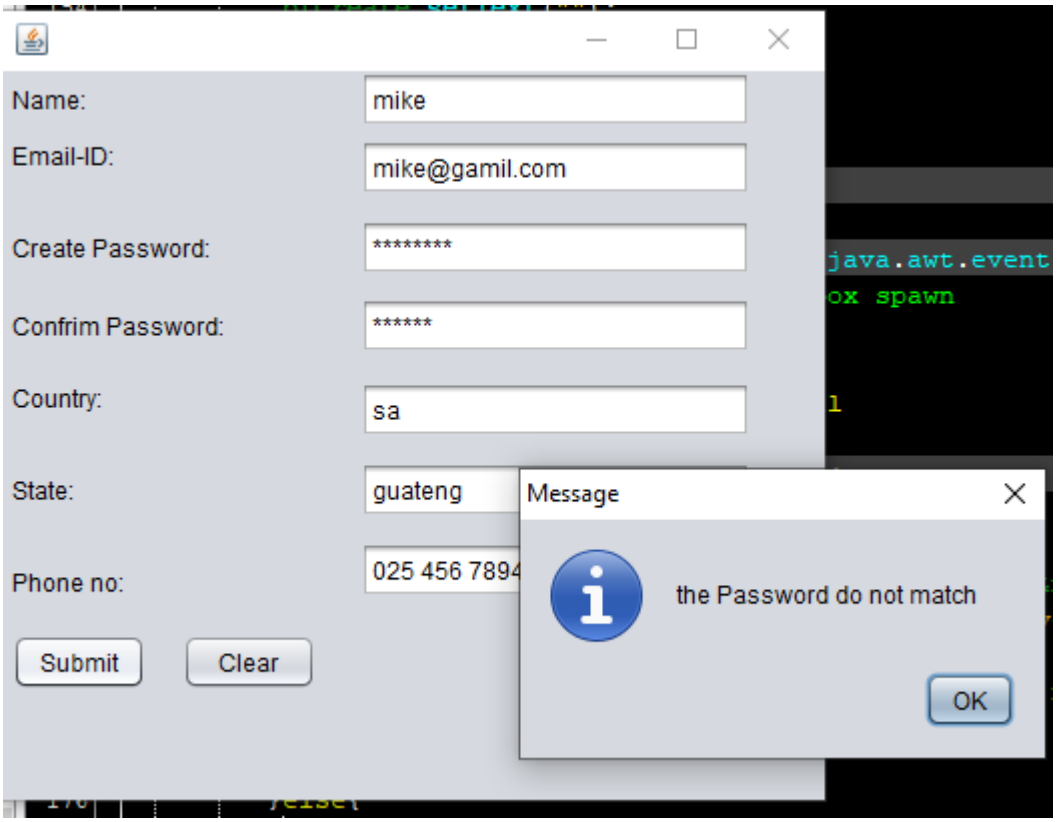


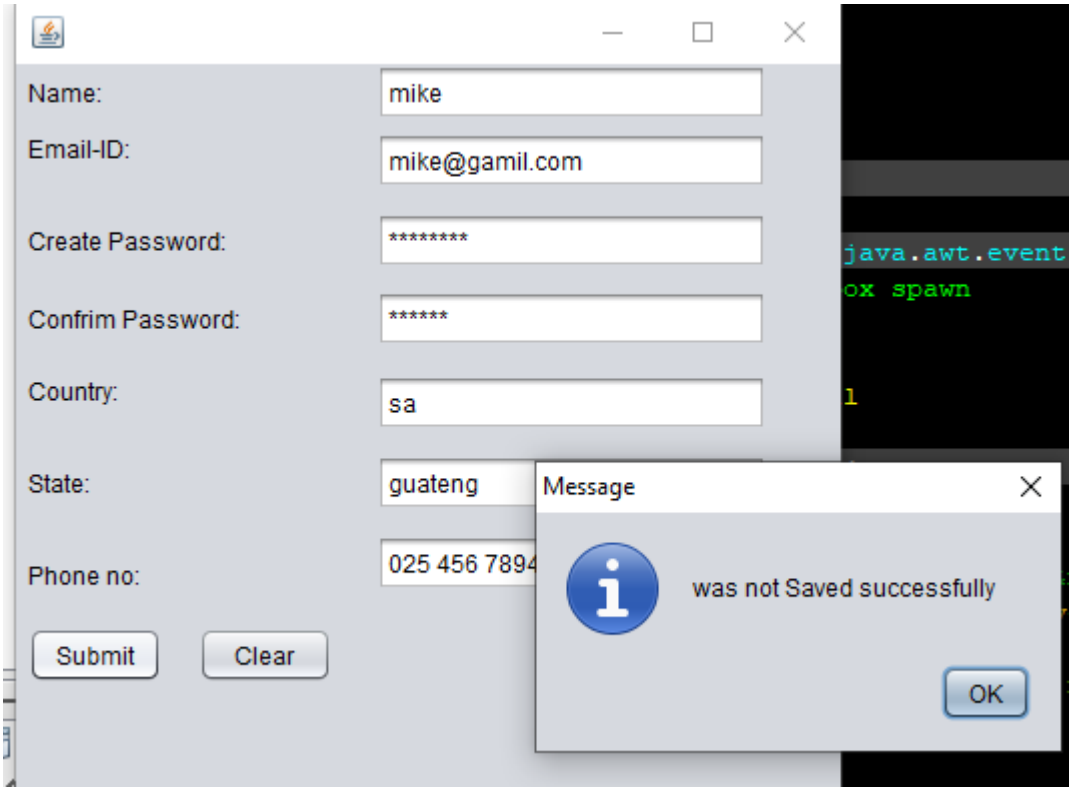
When I click on the clear button

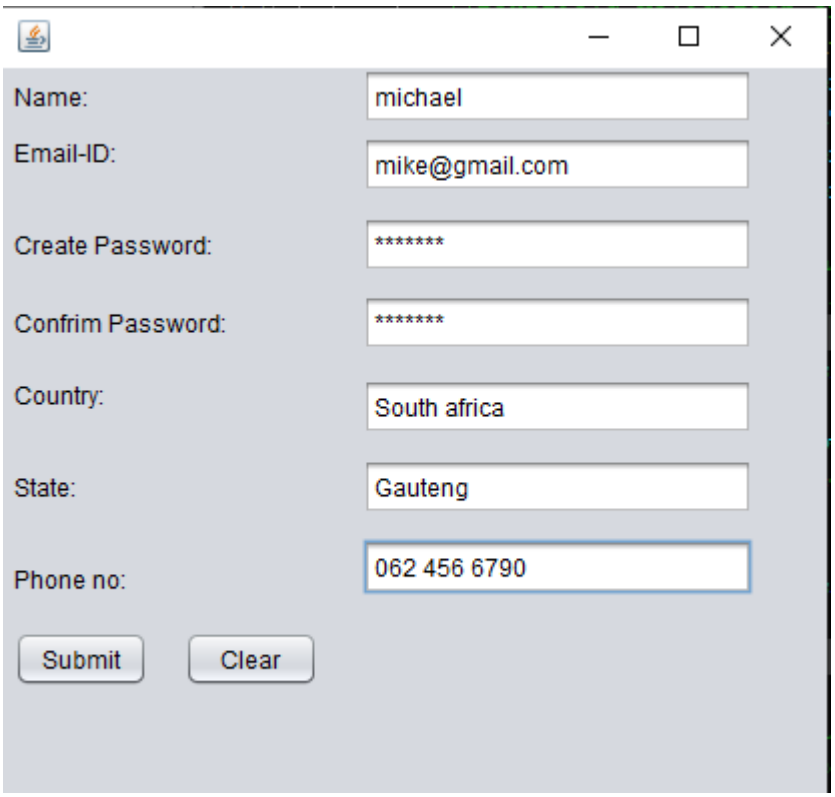


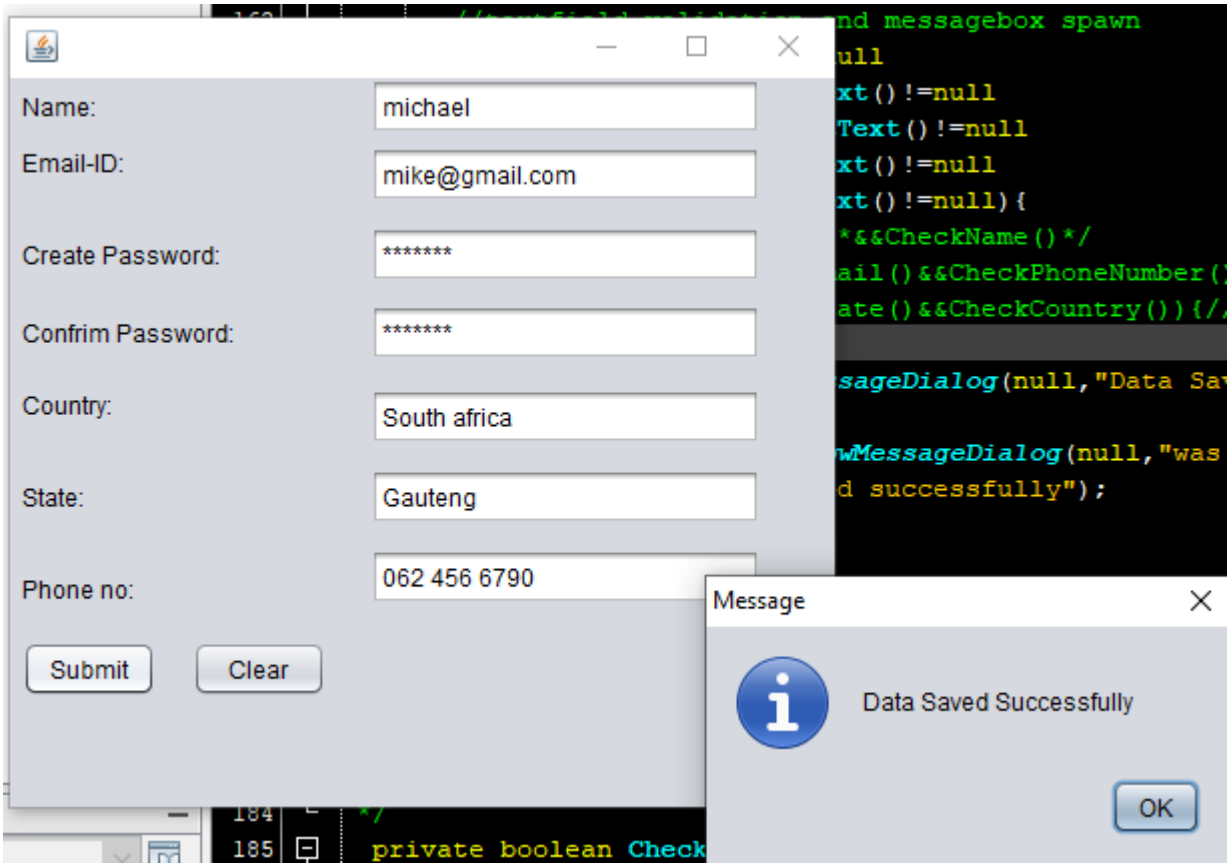



Name:
mike
Email-ID:
mike@gamil.com
Create Password:
****
Confrim Password:
******
Country:
sa
State:
guateng
Phone no:
025 456 7894
Submit
Clear


Name:
mike
Email-ID:
mike@gamil.com
Create Password:
****
Confrim Password:
******
Country:
sa
State:
guateng
Phone no:
025 456 7894
Submit
Clear
Message
the Password do not match
OK


Name:
mike
Email-ID:
mike@gamil.com
Create Password:
****
Confrim Password:
******
Country:
sa
State:
guateng
Phone no:
025 456 7894
Submit
Clear
Message
was not Saved successfully
OK
java.awt.event
ox spawn
1


Name:
michael
Email-ID:
mike@gmail.com
Create Password:
****
Confrim Password:
****
Country:
South africa
State:
Gauteng
Phone no:
062 456 6790
Submit
Clear


Name:
michael
Email-ID:
mike@gmail.com
Create Password:
****
Confrim Password:
****
Country:
South africa
State:
Gauteng
Phone no:
062 456 6790
Submit
Clear
Message
Data Saved Successfully
OK